On Reinforcement Learning with Adversarial Corruption and Its Application to Block MDP

Tianhao Wu [*1,2] **Yunchang Yang** [*3] **Simon S. Du** [4] **Liwei Wang** [3,5]

Abstract

We study reinforcement learning (RL) in episodic tabular MDPs with adversarial corruptions, where some episodes can be adversarially corrupted. When the total number of corrupted episodes is known, we propose an algorithm, Corruption Robust Monotonic Value Propagation (CR-MVP), which achieves a regret bound of $\tilde{O}\left(\left(\sqrt{SAK}+S^2A+CSA\right)\mathsf{polylog}(H)\right)$, where S is the number of states, A is the number of actions, H is the planning horizon, K is the number of episodes, and C is the known corruption level. We also provide a novel lower bound, which indicates that our upper bound is nearly tight. Finally, as an application, we study RL with rich observations in the block MDP model. We provide the first algorithm that achieves a $\sqrt{K}$-type regret in this setting and is oracle efficient.

1. Introduction

Reinforcement Learning (RL) has become a ubiquitous paradigm for decision-making in multi-stage environments. In a reinforcement learning problem, an agent is trained to interact with an environment in order to maximize its cumulative rewards through time. We model the environment as a Markov decision process (MDP) whose transition dynamics are unknown. As the agent interacts with the environment it observes the states, actions and rewards generated by the system dynamics. This has seen widespread applications in many areas including robotics (Kober et al., 2013), computer gaming (Mnih et al., 2015; Silver et al., 2017) and stock market (Yang et al., 2020a).

However, in many real world applications, a trained agent is vulnerable to corrupted data stemming from malicious entities (Huang et al., 2017; Ma et al., 2019), non-malicious yet non-stationary behavior, or simply errors in the system. Data corruption may lead the agent to exhibit inefficient and often unsafe behavior. In order to guarantee the safety and robustness of the agent against data corruption, it is of great importance to design efficient algorithms that are robust to data corruption in RL.

Previous works handling corruption mainly focus on the multi-armed bandit setting (Lykouris et al., 2018; Gupta et al., 2019), which is not directly applicable to the RL setting. Recently, Lykouris et al. (2019) initiates the study of episodic reinforcement learning under adversarial corruptions. They assume that both the reward and the transition of the underlying system can be corrupted. In this case they present an algorithm that achieves $\tilde{O}(C\sqrt{SAHK}+CS^2A+C^2SA)$ regret bound,[1] where S is the number of episodes, A is the number of actions, H is the planning horizon, K is the number of episodes, and C is the known number of corrupted episodes (also called corruption level). Unfortunately, when C is large, say $C=\Omega\left(\sqrt{K}\right)$, their bound becomes vacuous. Therefore, a better bound with respect to C is needed. Furthermore, they assume that the adversary decides whether to corrupt one episode after seeing the agent's policy, but *before* the agent takes its action at each step. However, in many real-world applications, the adversary is allowed to make such decision *after* the agent takes an action. For example, in the stock market, the opponent may choose whether to manipulate the price of the stock after seeing your operation. This kind of adversary is typically stronger as it receives more information before it decides whether to corrupt the environment. In this paper, we take a step towards answering the following question:

Can we achieve a tight regret bound in RL with stronger adversarial corruptions?

[*]Equal contribution [1]Peking University [2]Pazhou Lab, Guangzhou, 510330, China [3]Center for Data Science, Peking University [4]University of Washington [5]Key Laboratory of Machine Perception, MOE, School of EECS, Peking University. Correspondence to: Liwei Wang <wanglw@cis.pku.edu.cn>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

[1] $\widetilde{O}(\cdot)$ hides logarithmic factors. Sometime we write $\mathsf{polylog}(H)$ to highlight the dependency on H.

1.1. Our Contributions

We study the stronger type adversary which can decide whether to corrupt the environment after it observes the current state and the action of the agent, and we assume that the adversary only changes the agent's observation while not changing the underlying state. Note the adversary can corrupt consecutive time steps to replace the state and reward according to another MDP, hence we can stimulate the transition corruption. Different from (Lykouris et al., 2019), we assume that an upper bound of the corruption level C is known in prior. Although Lykouris et al. (2019) also discusses the known case, they assume that the adversary changes the underlying state, which is different from our setting. While this is a stronger assumption than the case where the corruption level is unknown, in many applications, an upper bound of C is known, for which we will discuss more in Section 6. We summarize our contributions below.

- We present a conceptually simple and computationally efficient algorithm. Our algorithm, Corruption Robust Monotonic Value Propagation (CR-MVP), is similar to the MVP algorithm in (Zhang et al., 2020a), with a delicate alteration to the form of the "exploration bonus" in order to make it robust against corruptions.
- Theoretically, we show that our algorithm achieves a high probability regret bound $\tilde{O}\left(\left(\sqrt{SAK}+S^2A+CSA\right)\text{polylog}(H)\right)$. Our bound is much tighter than that in (Lykouris et al., 2019), which is $\tilde{O}(C\sqrt{SAHK}+CS^2A+C^2SA)$. More importantly, ignoring the CSA term, our upper bound matches the best known upper bound of tabular MDP without corruption, $\tilde{O}\left(\left(\sqrt{SAK}+S^2A\right)\text{polylog}(H)\right)$ (Zhang et al., 2020a).
- We further study what is the optimal dependency on the corruption level C. We establish a novel lower bound, which states that any algorithm must incur an $\Omega(CSA)$ regret when as long as $K \geq \Omega(CSA)$. This lower bound, together with previous lower bounds contextual bandits and tabular reinforcement learning (Osband & Van Roy, 2016; Jaksch et al., 2010; Bubeck & Cesa-Bianchi, 2012), shows that our upper bound is nearly-tight up to an additive S^2A factor.
- As an application, we consider block MDP, which has studied extensively in many previous works (Du et al., 2019; Misra et al., 2020; Agarwal et al., 2020), and has widespread applications in image and text tasks. We provide an algorithm, ID-MVP, based on CR-MVP, which is a statistically and computationally efficient algorithm (with access to a least square oracle) with an $\sqrt{K}$-type regret bound.

2. Related Work

Multi-armed Bandit and Reinforcement Learning with Corruption. Many prior works studied Multi-armed Bandit and Reinforcement Learning problems with adversarial or stochastic corruptions. The model of adversarial corruption was first introduced by (Lykouris et al., 2018). They consider stochastic multi-armed bandits with corrupted rewards where the corruption level C is unknown, and provide an algorithm that achieves $\mathcal{O}\left(AC\sum_{a\neq a^\star}\frac{(\log(AT/\delta))^2}{\Delta(a)}\right)$[2] regret with probability at least $1-\delta$. They also provide a $\Omega(C)$ lower bound. Gupta et al. (2019) provides a better algorithm in the bandit setting, which improves the upper bound to $O\left(AC+\sum_{i\neq i^*}\frac{\log T}{\Delta_i}\log\left(\frac{A}{\delta}\log T\right)\right)$. They also claim a bound of $\tilde{O}\left(C+\sum_{i\neq i^*}1/\Delta_i\right)$ when C is known. Zimmert & Seldin (2019) further optimize the dependence on the number of actions via an elegant modification of mirror descent, for the weaker notion of pseudoregret and assuming that the best action is unique. Jin et al. (2019) considers the bandit-feedback setting in MDPs, where rewards are adversarial, and are only observed for state-action pairs visited. They provide regret bounds under the assumption that the adversary can only corrupt rewards and not transition dynamics.

The most related work is (Lykouris et al., 2019), which extends the adversarial corruption setting to episodic reinforcement learning. They assume that both the reward and the transition of the underlying system could be corrupted, and the corruption level is unknown. In this case they present an algorithm that achieves $\tilde{O}(C\sqrt{SAHK}+CS^2A+C^2SA)$ regret bound. However, when C is large, their bound becomes vacuous. For example, when $C=O(\sqrt{K})$, their bound grows linearly with respect to K, which is unacceptable. Also, their setting is different from ours. To be concrete, in their setting, before each episode the adversary decides whether to corrupt the episode, and if an episode is corrupted, the reward function and transition will be changed. As a comparison, we allow the adversary to decide whether to corrupt the step after it sees the action of the agent in this episode, and the adversary will only mislead the agent by disturbing its observation on the state and reward signal, while leaving the underlying state and reward that the agent actually receives unchanged. See Section 3 for more discussions.

Episodic Tabular MDP. There is a long list of sample complexity guarantees for episodic tabular RL. Previous papers use two measures to quantify sample complexity: regret (Bartlett & Tewari, 2012; Jaksch et al., 2010; Osband et al., 2013; Azar et al., 2017; Osband & Van Roy, 2017;

[2]Here Δ_i is the suboptimal gap of arm i, A is the number of arms, and C is the total injected corruption.

Agrawal & Jia, 2017; Fruit et al., 2018; Talebi & Maillard, 2018; Simchowitz & Jamieson, 2019; Russo, 2019; Zhang & Ji, 2019; Cai et al., 2020; Zhang et al., 2020b; Yang et al., 2020b; Pacchiano et al., 2020; Neu & Pike-Burke, 2020) and PAC-RL sample complexity (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002; Kakade et al., 2003; Strehl et al., 2006; Strehl & Littman, 2008; Kolter & Ng, 2009; Szita & Szepesvári, 2010; Lattimore & Hutter, 2012; Dann & Brunskill, 2015; Dann et al., 2019; Dong et al., 2019). As is pointed out in (Dann et al., 2017; Jin et al., 2018), suppose that one has an algorithm that achieves $CK^{1-\alpha}$ regret for some $\alpha \in (0,1)$ and some C independent of T, by randomly selecting from policy π^k used in K episodes, π satisfies $\left[\mathbb{E}_{s_1\sim\mu}\left[V_1^*\left(s_1\right) - V^{\pi}\left(s_1\right)\right] = O\left(CK^{-\alpha}\right)\right.$. This reduction is often near-optimal to obtain PAC-RL sample complexity guarantee. On the other hand, there is no general near-optimal reduction that transform a PAC-RL bound to a regret bound.

Contextual Decision Process and Block Markov Decision Process The block Markov decision process (BMDP) setting belongs to a broader class of settings called contextual decision process (CDP), which was first studied in (Krishnamurthy et al., 2016). Jiang et al. (2017) gives an algorithm with polynomial sample complexity guarantee under the low bellman rank assumption. Dann et al. (2018) gives an oracle-efficient algorithm when the transition of hidden state is deterministic. Recently, Dong et al. (2020) develop an online learning algorithm that learns the optimal value function while at the same time achieving low cumulative regret during the learning process. This is the first algorithm that achieves $\sqrt{K}$ regret in CDP. However, their algorithm is also computationally intractable.

BMDP was first considered in (Du et al., 2019). Their algorithm, PCID, learns a policy cover with polynomial sample complexity under the reachability and separability assumption. Misra et al. (2020) removes the separability assumption, at the cost of higher sample complexity. Feng et al. (2020) also designed provably efficient algorithms for BMDP but using a different oracle. Recently, Agarwal et al. (2020) considers a more general low rank MDP setting, which includes BMDP as a special case. Their algorithm does not rely on reachability and separability, but the sample complexity is much higher. To our knowledge, none of these works provides a $\sqrt{K}$-type regret bound.

3. Preliminaries

3.1. Notations

We use $\Delta(\cdot)$ to represent the set of all probability distributions on a set. For $n \in \mathbb{N}_+$, we denote $[n] = \{1, 2, ..., n\}$. We use $O(\cdot)$, $\Theta(\cdot)$, $\Omega(\cdot)$ to denote the big-O, big-Theta, big-Omega notations. We use $\tilde{O}(\cdot)$ to hide logarithmic factors. Sometime we explicitly write out the polylog-dependency on H. For any finite set S, we write $U(S)$ to denote the uniform distribution over S. Given an h-step policy π we write $\pi \odot a$ for the $(h+1)$ -step policy that executes π for h steps and chooses action a in step $h+1$. Similarly, if η is a policy mixture and ν a distribution over $\mathcal{A}$, we write $\eta \odot \nu$ for the policy mixture equivalent to first sampling and following a policy according to η and then independently sampling and following an action according to ν. We write $\triangle_d$ for the simplex in $\mathbb{R}^d$.

3.2. Problem Formulation

We consider tabular MDP, which is represented by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, P, R)$, where $\mathcal{S}$ and $\mathcal{A}$ are the set of states and actions, H is the number of steps in each episode, P is the transition probability matrix. With a slight abuse of notation, We denote by $P(\cdot|s,a) \in \Delta(\mathcal{S})$ the probability distribution over the next state after the agent takes action a in state s. $R : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$ is the reward distribution so that $R(s,a)$ is the reward distribution after the agent takes action a in state s. We denote $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as the expectation of the reward. For notational convenience, we use $P_{s,a}$ and $P_{s,a,s'}$ to denote $P(\cdot|s,a)$ and $P(s'|s,a)$, respectively.

In each episode, the agent starts from an initial state s_1. At each step $h \in [H]$, the agent takes action a_h in state s_h and receive a reward r_h generated from the distribution $R(s,a)$, the state transits to s_{h+1} according to the distribution $P(\cdot|s,a)$.

Previous works assume that $r_h \in [0,1]$ for all $h \in [H]$, which implies that the total reward $\sum_{h=1}^H r_h \in [0,H]$. In order to eliminate artificial regret blow up due to scaling, one should scale down the reward by a $\frac{1}{H}$ factor such that the total reward is in $[0,1]$, hence leads to the $r_h \in [0, \frac{1}{H}]$ assumption, *a.k.a.* the uniformly bounded reward assumption. In this paper, we follow the more general assumption 1 in previous works (Jiang & Agarwal, 2018; Wang et al., 2020; Zhang et al., 2020a) which says that the total reward $\sum_{h=1}^H r_h$ is bounded by 1.

Assumption 1 (Bounded Total Reward)**.** *The reward r_h satisfied that $r_h \geq 0$ for all $h \in [H]$. Moreover, for all policy π, $\sum_{h=1}^H r_h \leq 1$ almost surely.*

We point out that by adopting either Assumption 1 or the uniformly bounded reward assumption, one ignores the difficulty caused by the scaling which is unimportant, and focus on the hardness due to the planning horizon and unknown transition.

3.2.1. Episodic Tabular MDP with Adversarial Corruptions

In this section we discuss our definition of corruption, and the difference between our setting and previous settings such as (Lykouris et al., 2019). In the corrupted setting, we have a nominal MDP $\mathcal{M}$, which the learner faces in all steps that are not corrupted by the adversary. There are two major kinds of adversary, which we denote by **weak adversary** and **strong adversary** respectively.

Weak adversary: Most prior works adopt this kind of adversary, for example (Lykouris et al., 2019). Assume that after the agent takes the action a_{t-1} at state s_{t-1} and receives a reward r_{t-1}, the adversary first observe the next step policy π_t and decides whether to corrupt the t^{th} time step before the environment generate the next state s_t. If the adversary decides to corrupt the next step, it generates an arbitrary state s'_t and the corresponding reward function $\tilde{r}(s'_t, \cdot) \in R^S$. After the agent observes the corrupted state s'_t, it takes action a_t and observes $\tilde{r}(s'_t, a_t)$ instead of $r(s_t, a_t)$.

Strong adversary: In contrast to a weak adversary, a strong adversary can decide whether to corrupt the current step reward r_t and the next state s_{t+1} after the agent takes an action. In other words, after the agent takes action a_t at state s_t, the adversary decides whether to corrupt the current reward and the next time step. If the adversary decides to corrupt, it replaces the reward r_t with an arbitrary corrupted reward r'_t and generates an arbitrary state s'_{t+1} as well as the corresponding reward function $\tilde{r}(s'_{t+1}, \cdot) \in R^S$.

A strong adversary at time $t-1$ and a weak adversary at time t have the same ability to corrupt the state and reward function at the t^{th} step, but the strong adversary can do more by corrupting the reward function at time step $t-1$. More importantly, a strong adversary can corrupt the reward after seeing the agent's action at this timestep, while the weak one can only see the random policy instead of a deterministic action.

In our setting, we assume that the adversary only confuses the agent's observation while not changing the underlying state and reward that the agent actually receives. To be more specific, when an adversary decides to corrupt the state s_t into s'_t, the agent will observe s'_t, while staying in s_t actually. This kind of corruption is common in practice. For example, a robot may only receive images with stochastically or adversarially perturbations, while the true environment remains unchanged (Eykholt et al., 2018).

The *corruption level* is defined as the total number of steps at which is corrupted. Note that our definition of corruption level is different from the definition in (Lykouris et al., 2019). We assume that the agent knows an upper bound of corruption level C at the beginning. However, the agent does not know which steps are corrupted. This is where the difficulty of the problem lies.

Remark: Under the strong adversary assumption there does not exist an algorithm that can achieve an $\tilde{O}(\text{poly}(S, A, H)(\sqrt{K} + C))$ regret upper bound without knowing the corruption level C beforehand. In fact, even in the multi-armed bandit setting, Proposition 1 shows that if an algorithm achieves a $\tilde{O}(K^\alpha C^\beta)$ regret upper bound in this setting, then $\alpha + \beta/2 \geq 1$. However, the above argument does not mean that the $\tilde{O}(\text{poly}(H, S, A)(\sqrt{K}C + C^2))$ regret bound in (Lykouris et al., 2019) is optimal, since they studied the weak adversary instead of strong adversary. The proof of Proposition 1 is in Appendix C.

Proposition 1. *In a MAB instance with adversarial corruptions, assume that the corruption level C is unknown. If there exists an algorithm $\mathcal{A}$ that can achieve a high probability regret upper bound $\tilde{O}\left(\sqrt{K} + K^\alpha C^\beta\right)$ for any C and K, then $\alpha + \beta/2 \geq 1$.*

A policy π is a set of functions $\{\pi_h : \mathcal{S} \mapsto \Delta(\mathcal{A})\}_{h\in[H]}$. Given a policy π, a level $h \in [H]$ and a state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$, the $Q-$function and the value function without corruption are defined as:

$$Q_h^\pi(s, a) = \mathbb{E}[\sum_{h'=h}^{H} r_{h'} | s_h = s, a_h = a, \pi],$$
$$V_h^\pi(s) = \mathbb{E}[\sum_{h'=h}^{H} r_{h'} | s_h = s, \pi].$$

We let $V_{H+1}(s) = 0$ and $Q_{H+1}(s, a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$. We use Q_h^* and V_h^* to denote the optimal Q-function and V-function at level $h \in [H]$ without corruptions, which satisfies $Q_h^*(s, a) = \max_\pi Q_h^\pi(s, a)$ and $V_h^*(s) = \max_a Q^*(s, a)$ respectively.

We use $\tilde{Q}$ and $\tilde{V}$ to denote the Q-function and value function with corruptions, which is the reward the agent actually receives under with corruption. Let $\tilde{s}_t$ be the state observed by the agent, *i.e.*, $\tilde{s}_t = s'_t$ if it is corrupted by the adversary, otherwise $\tilde{s}_t = s_t$. Due to corruptions, the agent will play action $\pi(\tilde{s}_t)$ instead of $\pi(s_t)$. Then $\tilde{Q}$ and $\tilde{V}$ are defined as:

$$\tilde{Q}_h^\pi(s, a) = \mathbb{E}[r(s, a) + \sum_{h'=h+1}^{H} r(s_{h'}, \pi(\tilde{s}_{h'})) | s_h = s, a_h = a],$$
$$\tilde{V}_h^\pi(s) = \mathbb{E}[\sum_{h'=h}^{H} r(s_{h'}, \pi(\tilde{s}_{h'})) | s_h = s].$$

In such setting, we define regret as the difference between the optimal value of the original MDP (without corruption)

and the expected cumulative reward the agent actually receives. That is,

$$\text{Regret}(K) = \sum_{k=1}^{K} V_1^*(s_1^k) - \tilde{V}_1^{\pi^k}(s_1^k), \tag{1}$$

where π_k is the policy in the k-th episode.

4. Corruption Robust Monotonic Value Propogation

In this section we introduce our algorithm *Corruption Robust Monotonic Value Propagation* (CR-MVP) for episodic tabular MDP with adversarial corruptions, which is inspired by the MVP algorithm by (Zhang et al., 2020a) with some novel modifications in order to deal with corruptions.

The pseudocode of CR-MVP is listed in Algorithm 1. The algorithm adopt an update framework similar to the doubling framework proposed in (Jaksch et al., 2010). We define a trigger set $\mathcal{L} = \{2^{i-1} + C | 2^{i-1} + C \le KH, i = 1, 2, ...\}$. The algorithm proceeds through epochs where each epoch ends whenever there exists an (s, a) pair such that the number of visits of (s, a) falls in $\mathcal{L}$. During each epoch, we keep using the same policy π^k induced by the current Q-function.

At the beginning of each epoch, we update the empirical reward and transition probability of the triggered (s, a) pair. For the transition probability, we use maximum likelihood approach. Suppose $\tilde{N}^k(s, a, s')$ and $\tilde{N}^k(s, a)$ are the number of visit of (s, a, s') and (s, a) observed by the agent before the kth update, we set $\tilde{P}^k_{s,a} = \frac{\tilde{N}^k(s,a,s')}{\tilde{N}^k(s,a)}$. For the ease of exposition we abuse the notation and drop the dependencies on k. Note that $\tilde{N}$ is not the actual number of visit due to corruptions. We denote $\hat{N}(s, a)$ and $\hat{N}(s, a, s')$, the actual visitation of (s, a) and (s, a, s'), respectively. Note $\tilde{P}$ is biased because of the contaminated nature of $\tilde{N}_h(s, a)$ and $\tilde{N}_h(s, a, s')$. We denote the unbiased estimator by $\hat{P}_{s,a} = \frac{\hat{N}(s,a,s')}{\hat{N}(s,a)}$ to distinguish it from the biased estimator $\tilde{P}$. We emphasize that the agent only have the access to the biased counter $\tilde{N}$ and biased transition $\tilde{P}$.

Our algorithm follows the idea of *optimism in the face of uncertainty*. A typical optimistic algorithm usually maintains an optimistic estimation of Q-function by adding a bonus term to the empirical Bellman Equation, *i.e.*, $\hat{Q}_h(s, a) = \hat{r}(s, a) + \hat{P}_{s,a}V_{h+1} + \hat{b}_h(s, a)$. For example, Zhang et al. (2020a) constructed a Bernstein type bonus of the form $\hat{b}_h(s, a) = c_1\sqrt{\frac{\mathbb{V}(\hat{P}, V_{h+1})}{\max\{\hat{N}(s,a),1\}}} + c_2\sqrt{\frac{\hat{r}(s,a)\iota}{\max\{\hat{N}(s,a),1\}}} + c_3\frac{\iota}{\max\{\hat{N}(s,a),1\}}$.

However, the above result relies on the access to the unbiased estimators $\hat{N}$ and $\hat{P}$. In order to maintain the optimism of the Q-function in the corrupted setting, we view Q_h as a function of $\tilde{N}, \tilde{P}$, which we denote as $Q_h(\tilde{N}, \tilde{P})$, and design $\tilde{b}_h$ such that Q_h is an upper bound of the $\hat{Q}_h(\hat{N}, \hat{P})$ constructed by the unbiased estimators, *i.e.*,

$$\begin{aligned} &Q_h(\tilde{N}, \tilde{P})(s, a) = \tilde{r}(s, a) + \tilde{P}_{s,a}V_{h+1} + \tilde{b}_h \\ \ge &\hat{Q}_h(\hat{N}, \hat{P})(s, a) = \hat{r}(s, a) + \hat{P}_{s,a}V_{h+1} + \hat{b}_h. \end{aligned}$$

We specify a choice of $\tilde{b}_h$ according to the following lemma,

Lemma 1. *Suppose* $c_1, c_2, c_3 \ge 0$, *let* $\tilde{b}_h = \tilde{b}_{h,con} + \tilde{b}_{h,bia}$,

Algorithm 1 Corruption Robust Monotonic Value Propagation

Input: Trigger set $\mathcal{L} \leftarrow \{2^{i-1}+C | 2^{i-1}+C \le KH, i = 1, 2, ...\}$, where C is the corruption level. Constants $c_1 = \frac{460}{9}, c_2 = 2\sqrt{2}, c_3 = \frac{544}{9}$
for $(s, a, s', h) \in \mathcal{S} \times \mathcal{A} \times \mathcal{S} \times [H]$ **do**
  $\tilde{N}(s, a) \leftarrow 0; \theta(s, a) \leftarrow 0; n(s, a) \leftarrow 0; V_h(s) \leftarrow 1;$
  $\tilde{N}(s, a, s') \leftarrow 0; \tilde{P}_{s,a,s'} \leftarrow 0; Q_h(s, a) \leftarrow 1.$
end for
for $k = 1, 2, ..., K$ **do**
  for $h = 1, 2, ..., H$ **do**
    Observe s^k_h, take action $a^k_h = \arg\max_a Q_h(s^k_h, a)$;
    Receive reward r^k_h and next state s^k_{h+1}.
    Set $(s, a, s', r) \leftarrow (s^k_h, a^k_h, s^k_{h+1}, r^k_h)$.
    Set $\tilde{N}(s, a, s') \leftarrow \tilde{N}(s, a, s') + 1$,
      $\tilde{N}(s, a) \leftarrow \tilde{N}(s, a) + 1$,
      $\theta(s, a) \leftarrow \theta(s, a) + r$.
    if $\tilde{N}(s, a) \in \mathcal{L}$ **then**
      Set $\tilde{r}(s, a) \leftarrow \frac{\theta(s,a)}{\tilde{N}(s,a)}$,
        $\tilde{P}_{s,a,\cdot} \leftarrow \tilde{N}_{s,a,\cdot}/\tilde{N}(s, a)$,
        $n(s, a) \leftarrow \tilde{N}(s, a)$.
      Set TRIGGERED= TRUE.
    end if
    if TRIGGERED **then**
      for $h = H, H-1, ..., 1$ **do**
        for $(s, a) \in \mathcal{S} \times \mathcal{A}$ **do**
          Set $\tilde{b}_h = 2\min\{\frac{2C}{|n-C|}, 1\} + c_1\min\{\sqrt{\frac{\mathbb{V}(\tilde{P}, V_{h+1})\iota}{|n-C|}} + \frac{\sqrt{C\iota}}{|n-C|}, 1\} + c_2\min\{\sqrt{\frac{\tilde{r}\iota}{|n-C|}} + \frac{\sqrt{C\iota}}{|n-C|}, 1\} + c_3\min\{\frac{\iota}{|n-C|}, 1\}$,
          $Q_h(s, a) \leftarrow \min\{\tilde{r}(s, a) + \tilde{P}_{s,a}V_{h+1} + \tilde{b}_h(s, a), 1\}$,
          $V_h(s) \leftarrow \max_a Q_h(s, a)$.
        end for
      end for
    end if
    Set TRIGGERED=FALSE
  end for
end for

where

$$\tilde{b}_{h,con} = c_1 \min\left\{\sqrt{\frac{\mathbb{V}(\tilde{P}, V_{h+1})\iota}{|\tilde{N} - C|}}, 1\right\} + c_2 \min\left\{\sqrt{\frac{\tilde{r}\iota}{|\tilde{N} - C|}}, 1\right\} + c_3 \min\left\{\frac{\iota}{|\tilde{N} - C|}, 1\right\},$$

and

$$\tilde{b}_{h,bia} = 2\min\left\{\frac{2C}{|\tilde{N} - C|}, 1\right\} + (c_1 + c_2)\min\left\{\frac{\sqrt{C\iota}}{|\tilde{N} - C|}, 1\right\}.$$

Then $Q_h \geq \hat{Q}_h$.

Note that the term $\tilde{b}_h$ consists of two terms $\tilde{b}_{h,con}$ and $\tilde{b}_{h,bia}$, where the first one $\tilde{b}_{h,con}$ comes from a standard concentration process. The second term $\tilde{b}_{h,bia}$ is novel because it is designed to offset the bias. To analyze the regret, we first note that it can be decomposed into the sum of bonus and a sum of martingales which is bounded by $\sqrt{SAK} + S^2A$, *i.e.*,

$$\text{Regret} \leq \tilde{O}(\sum_{k=1}^{K}\sum_{h=1}^{H} \tilde{b}_h^k) + \sqrt{SAK} + S^2A.$$

The bonus term can be further decomposed into the concentration-induced term $\sum_{k=1}^{K}\sum_{h=1}^{H} \tilde{b}_{h,con}$ and the bias-offset term $\sum_{k=1}^{K}\sum_{h=1}^{H} \tilde{b}_{h,bia}$. The concentration-induced term can be bounded by $\sqrt{SAK} + CSA$ using standard techniques from (Zhang et al., 2020a). Hence, we only need to deal with the bias-offset term $\tilde{b}_{h,bia}$. We can upper bound this term by using the inequality $\sum_{k=1}^{K}\sum_{h=1}^{H} \min\{\frac{D}{|\tilde{N}-C|}, 1\} \leq CSA + DSA\iota$ which holds for any $D \geq 0$, where $\iota = \log(HSAK/\delta)$ is a log factor. Set $D = 2C$ and $\sqrt{C\iota}$ respectively, we obtain $\sum_{k=1}^{K}\sum_{h=1}^{H} \tilde{b}_{h,bia}^k \leq \tilde{O}(CSA)$. In other words, we prove the following theorem regarding the upper bound. The full proof in given in Appendix A.

Theorem 1. *(Regret upper bound of CR-MVP) Under assumption 1, with probability at least* $1 - \delta$*, the regret of algorithm 1 satisfies:*

$$\text{Regret}(K) \leq \tilde{O}(\sqrt{SAK} + S^2A + CSA),$$

where K *is the total number of episodes. In other words, the regret caused by the corruptions only scales linearly with regard to* C*.*

In the unknown C setting, Lykouris et al. (2019) proposed the algorithm with $\tilde{O}(\text{poly}(H, S, A)(\sqrt{K}C + C^2))$ regret. Our theorem improves the regret in the known C setting to $\tilde{O}(\text{poly}(S, A)(\sqrt{K} + C))$ which separates $\sqrt{K}$ and C. Also, our algorithm is computationally efficient.

5. Lower Bounds

In this section we show that the bound $\tilde{O}(\sqrt{SAK} + CSA)$ in Theorem 1 is unimprovable. Previous literature (Osband & Van Roy, 2016; Jaksch et al., 2010; Bubeck & Cesa-Bianchi, 2012) have shown that the $\tilde{O}(\sqrt{SAK})$ term cannot be improved in tabular MDP without corruption. The following theorem guarantees that the second term $\tilde{O}(CSA)$ is optimal.

Theorem 2. *For any fixed* C*,* S*,* A*, and any algorithm* $\mathcal{A}$*, there exists an episodic MDP with horizon* $H = O(\log_A S)$*, such that the regret* $\mathcal{A}$ *incurred after* K *episodes in this MDP is at least* $\Omega(CSA)$*, where* K *satisfies* $K \geq 2CSA$*.*

The full proof is deferred to Appendix C. Here we give a brief description for the proof ideas. For simplicity we only consider the bandit setting with A arms (which means $H = 1$ and $S = 1$) and every time the agent chooses to pull an arm a, the agent may mislead the agent to believe that it pulls another arm a'.

Now we consider two environments. In the first environment there is no corruptions. Since an adversary can corrupt the bandit for at most C times, we claim that the agent must pull each arm for at least C times. Otherwise, there must exists an arm $\tilde{a}$ that the agent pulls for less than C times. Then we can find another environment in which the adversary chooses to corrupt the agent and gives a low reward in steps that the agent chooses $\tilde{a}$ while the reward of $\tilde{a}$ is actually the largest. So the agent will falsely regard $\tilde{a}$ as a suboptimal arm and thus incur a linear regret in this environment. However,if the agent pulls all the arms for at least C times, then it will suffer an $O(AC)$ regret since it pulls all the $A - 1$ suboptimal arms for C times. Hence we reached the conclusion.

Note that our lower bound requires the adversary to be adaptive to the action taken by the agent, so it does not conflict with the claim in (Gupta et al., 2019) since they assume that the adversary cannot observe the action of the agent in each round.

6. Application to Block MDP

In this section we study RL with Rich Observations (a.k.a. block MDPs), as an application of our algorithm. We propose a new algorithm ID-MVP based on CR-MVP, and obtain a statistically and computationally efficient algorithm (with access to a least square oracle) with a regret bound that achieves $\sqrt{K}$ growth rate.

6.1. Settings

A block Markov decision process, or BMDP, refers to an environment described by a finite but unobservable latent state space $\mathcal{S}$, a finite action space $\mathcal{A}$, a possibly infinite

but observable context space $\mathcal{X}$, and a reward distribution $R(s,a)$ for each $s \in \mathcal{S}, a \in \mathcal{A}$. The dynamics of a BMDP is described by the initial state s_1, the transition probability function $p(s'|s,a)$, and the context emission function $q(x|s)$ for all $s, s' \in \mathcal{S}, a \in \mathcal{A}, x \in \mathcal{X}$.

In this paper we consider episodic reinforcement learning tasks with a finite horizon H. In each episode, the environment starts from the initial state s_1. In step $h \in [H]$, the environment generates a context $x_h \sim q(\cdot \mid s_h)$, then the agent observes x_h, takes action a_h and receives a reward $r_h \sim R(s_h, a_h)$. The environment transits to $s_{h+1} \sim p(\cdot|s_h, a_h)$. We assume that the total reward in an episode is bounded by 1, as previously mentioned. We assume that the MDP admits a block structure:

Assumption 2 (Block Structure)**.** *We assume that each context $x \in \mathcal{X}$ uniquely determines its generating state $s \in \mathcal{S}$, i.e. there is a decoding function $f^* : \mathcal{X} \mapsto \mathcal{S}$, such that $q(\cdot|s)$ is supported on $(f^*)^{-1}(s)$.*

The block structure implies that a BMDP is actually an MDP with $\mathcal{X}$ being its state set and $P(x' \mid x, a) = q(x' \mid f^*(x'))\, p(f^*(x') \mid f^*(x), a)$ being the transition operator. However, the state space is too large that traditional algorithms cannot be directly applied to this problem.

To streamline our analysis, we make a standard assumption for episodic settings. We assume that $\mathcal{S}$ can be partitioned into disjoints sets $\mathcal{S}_h, h \in [H+1]$, such that $p(\cdot \mid s, a)$ is supported on $\mathcal{S}_{h+1}$ whenever $s \in \mathcal{S}_h$. We refer to h as the level and assume that it is observable as part of the context, so the context space is also partitioned into sets $\mathcal{X}_h$. We use notation $\mathcal{S}_{[h]} = \cup_{\ell \in [h]} \mathcal{S}_\ell$ for the set of states up to level h and similarly define $\mathcal{X}_{[h]} = \cup_{\ell \in [h]} \mathcal{X}_\ell$. We assume that $|\mathcal{S}_h| \le M$ and $|\mathcal{A}| = A$.

We also present several crucial concepts. The first one is policy cover. For any state s, we define the maximum reaching probability of s as $\mu(s) := \max_\pi \mathbb{P}^\pi(s)$, where $\mathbb{P}^\pi(s)$ is the probability of reaching the state s when executing π. We assume that all the states are reachable, *i.e.*, $\mu(s) > 0$ for all s. We write $\mu_{\min} = \min_{s \in \mathcal{S}} \mu(s)$. We say that a set of policies Π_h is an ϵ-policy cover of $\mathcal{S}_h$ if for any $s \in \mathcal{S}_h$, there is a policy $\pi \in \Pi_h$, such that $\mathbb{P}^\pi(s) \ge \mu(s) - \epsilon$. A set of policies Π is an ϵ-policy cover of $\mathcal{S}$ if it is an ϵ-policy cover of $\mathcal{S}_h$ for all $h \in [H+1]$.

The second core concept is backward probability vector. For any distribution ν over (s_{h-1}, a_{h-1}), any $s \in \mathcal{S}_{h-1}, a \in \mathcal{A}$ and $s' \in \mathcal{S}_h$, the backward probability is defined as

$$b_\nu(s, a \mid s') = \frac{p(s' \mid s, a)\, \nu(s, a)}{\sum_{\tilde{s}, \tilde{a}} p(s' \mid \tilde{s}, \tilde{a})\, \nu(\tilde{s}, \tilde{a})} \tag{2}$$

For a given $s' \in \mathcal{S}_h$, we collect the probabilities $b_\nu(s, a \mid s')$ across all $s \in \mathcal{S}_{h-1}, a \in \mathcal{A}$ into the backward probability vector $\mathbf{b}_\nu(s') \in \triangle_{MA}$, padding with zeros if $|\mathcal{S}_{h-1}| < M$.

Algorithm 2 ID-MVP

Input: $N_{\mathrm{g}}, N_\phi, \tau > 0, \epsilon_{\mathrm{f}}, \delta > 0$
\\ Phase 1
Run PCID and obtain a $\frac{\mu_{min}}{2}$-policy cover $\Pi = \{\Pi_1, ..., \Pi_{H+1}\}$, where Π_i is a policy cover of the i-th layer.
\\ Phase 2
Let $\widehat{\mathcal{S}}_1 = \{s_1\}$. Let $\Pi_1 = \{\pi_0\}$ for the 0 -step policy π_0.
for $h = 2, ..., H+1$ **do**
  Let $\eta_h = U(\Pi_{h-1}) \odot U(\mathcal{A})$.
  Execute η_h for N_{g} times. Let $D_{\mathrm{g}} = \left\{\hat{s}^i_{h-1}, a^i_{h-1}, x^i_h\right\}_{i=1}^{N_g}$ where $\hat{s}_{h-1}$ is the index of $\pi_{\hat{s}_{h-1}}$ sampled by η_h.
  Learn $\hat{\mathbf{g}}_h$ by calling the ERM oracle on input D_{g}: $\hat{\mathbf{g}}_h = \operatorname{argmin}_{\mathbf{g} \in \mathcal{G}} \sum_{(\hat{s}, a, x') \in D_{\mathrm{g}}} \left\|\mathbf{g}(x') - \mathbf{e}_{(\hat{s}, a)}\right\|^2$.
  Initialize $\mathcal{Z} = \emptyset$ (dataset for learning latent states).
  Execute η_h for N_ϕ times. Let $\mathcal{Z} = \{\hat{z}_i = \hat{\mathbf{g}}_h(x^h_i)\}_{i=1}^{N_g}$.
  Learn $\hat{\mathcal{S}}_h$ and the state embedding map $\hat{\boldsymbol{\phi}}_h : \widehat{\mathcal{S}}_h \to \mathcal{Z}$ by clustering $\mathcal{Z}$ with threshold τ (see Algorithm 3).
  Define $\hat{f}_h(x') = \operatorname{argmin}_{\hat{s} \in \hat{\mathcal{S}}_h} \left\|\hat{\boldsymbol{\phi}}(\hat{s}) - \hat{\mathbf{g}}_h(x')\right\|_1$.
end for
\\ Phase 3
Call Algorithm 1 for the remaining T' steps, with state set $\{\hat{\mathcal{S}}_h\}_{h=1}^H$ and corruption level $2\epsilon_{\mathrm{f}} T' + \sqrt{2T' \ln \frac{\delta}{2}}$.

We make the following separability assumption as in (Du et al., 2019):

Assumption 3 (γ-Separability)**.** *There exists $\gamma > 0$ such that for any $h \in \{2, \ldots, H+1\}$ and any distinct $s', s'' \in \mathcal{S}_h$ the backward probability vectors with respect to the uniform distribution are separated by a margin of at least γ, i.e., $\|\mathbf{b}_\nu(s') - \mathbf{b}_\nu(s'')\|_1 \ge \gamma$, where $\nu = U(\mathcal{S}_{h-1} \times \mathcal{A})$.*

For deterministic transition we have $\gamma = 2$. This is a fairly general non-degenerate condition imposed on the transition dynamics.

6.2. An Algorithm with $\sqrt{K}$ Regret

In this section we propose our algorithm that achieves sublinear regret. The algorithm is called ID-MVP (Inductive Decoding and Monotonic Value Propagation), and is described in Algorithm 2.

The algorithm consists of three phases. In phase 1, we call PCID directly to obtain a $\frac{\mu_{min}}{2}$-policy cover. In phase 2, we learn a decoding function $\hat{f}$ with ϵ_{f}-decoding error. That is, for each level h we learn a state set $\hat{\mathcal{S}}_h$, together with a function $\hat{f}_h : \mathcal{X}_h \mapsto \hat{\mathcal{S}}_h$, such that there exists a bijection $\alpha_h : \widehat{\mathcal{S}}_h \to \mathcal{S}_h$, such that $\mathbb{P}_{x \sim q(\cdot|\alpha_h(\hat{s}))}\left[\hat{f}_h(x) = \hat{s}\right] \ge 1 - \epsilon_{\mathrm{f}}$. Here ϵ_{f} depends on our input N_{g}, N_ϕ.

Algorithm 3 Clustering

Input: Data points $\mathcal{Z} = \{\mathbf{z}_i\}_{i=1}^n$ and threshold $\tau > 0$.
Output: Cluster indices $\widehat{\mathcal{S}}$ and centers $\widehat{\phi} : \widehat{\mathcal{S}} \to \mathcal{Z}$.
Let $\widehat{\mathcal{S}} = \emptyset, k = 0$ (number of clusters).
while $\mathcal{Z} \neq \emptyset$ **do**
  Pick any $\mathbf{z} \in \mathcal{Z}$ (a new cluster center).
  Let $\mathcal{Z}' = \{\mathbf{z}' \in \mathcal{Z} : \|\mathbf{z} - \mathbf{z}'\|_1 \leq \tau\}$.
  Add cluster: $k \leftarrow k+1, \widehat{\mathcal{S}} \leftarrow \widehat{\mathcal{S}} \cup \{k\}, \widehat{\phi}(k) = \mathbf{z}$.
  Remove the newly covered points: $\mathcal{Z} \leftarrow \mathcal{Z} \backslash \mathcal{Z}'$.
end while

In order to construct f, we learn low dimensional representations of contexts as well as latent states in a shared space, namely $\triangle_{MA}$. We learn embedding functions $\mathbf{g} : \mathcal{X} \to \triangle_{MA}$ for contexts and $\boldsymbol{\phi} : \mathcal{S} \to \triangle_{MA}$ for states, with the goal that $\mathbf{g}(x)$ and $\boldsymbol{\phi}(s)$ should be close if and only if $x \in \mathcal{X}_s$. We assume that $\mathbf{g}$ is chosen from a function class $\mathcal{G}$, and make the following realizability assumption.

Assumption 4. *For any $h \in [H+1]$ and $\boldsymbol{\phi} : \mathcal{S}_h \to \triangle_{MA}$, there exists $\mathbf{g}_h \in \mathcal{G}$ such that $\mathbf{g}_h(x) = \boldsymbol{\phi}(s)$ for all $x \in \mathcal{X}_s$ and $s \in \mathcal{S}_h$.*

ID-MVP learns the decoding function by the following steps:

(1) Regression step: learn $\hat{\mathbf{g}}_h$. We define $\eta_h = U(\Pi_{h-1}) \odot U(\mathcal{A})$, execute η_h for N_g times, and collect a dataset of samples $D_g = \{\hat{s}^i_{h-1}, a^i_{h-1}, x^i_h\}_{i=1}^{N_g}$, where $\hat{s}^i_{h-1} = \hat{f}_{h-1}(x^i_{h-1})$. Each sample is drawn from distribution $\nu = U(\mathcal{S}_{h-1} \times \mathcal{A})$. Then we solve the following least square problem:

$$\hat{\mathbf{g}}_h \in \operatorname*{argmin}_{\mathbf{g} \in \mathcal{G}} \mathbb{E}_{(s,a,x') \sim D_g} \left[\left\| \mathbf{g}(x') - \mathbf{e}_{(s,a)} \right\|^2 \right]. \quad (3)$$

Our choice of η_h ensures that each state on the next level is reached with sufficient large probability. And under Assumption 3, Theroem 3.1 in (Du et al., 2019) implies that $\hat{\mathbf{g}}_h(x') \approx \mathbf{g}_h(x') = \mathbf{b}_\nu(s')$ for the distribution $\nu(\hat{s}_{h-1}, a_{h-1})$ induced by η_h.

Note that the optimization problem (3) is typically not hard to solve, as plenty of researches focus on developing efficient algorithms for such least square problems (Newville et al., 2016). Therefore we may assume that we have access to a computationally efficient least square oracle to solve the problem.

(2) Clustering Step: learn $\hat{\boldsymbol{\phi}}$ and $\hat{f}_h$. Thanks to separability, we can use the learned context embedding $\mathbf{g}_h$ as the foundation of clustering. We again run η_h for N_ϕ times, collecting a dataset $\mathcal{Z} = \left\{\hat{\mathbf{z}}_i = \hat{\mathbf{g}}_h\left(x^i_h\right)\right\}_{i=1}^{N_\phi}$. Then we run Algorithm 3 to identify all contexts generated by the same latent state. Each cluster corresponds to some latent state s' and any vector $\hat{\mathbf{g}}_h(x')$ from that cluster can be used to define the state embedding $\hat{\boldsymbol{\phi}}(s')$. The decoding function $\hat{f}_h$ is defined to map any context x' to the state s' whose embedding $\hat{\boldsymbol{\phi}}(s')$ is the closest to $\hat{\mathbf{g}}_h(x')$.

After $\hat{f}$ is learned, we can run Algorithm 1 with state set $\hat{\mathcal{S}}$, and when the agent observes a context x, it uses $\hat{s} = \hat{f}(x)$ as the current state. Since $\hat{f}$ has ϵ_{f}-decoding error, there will be approximately $2\epsilon_{\mathrm{f}}T' + \sqrt{2T' \ln \frac{\delta}{2}}$ steps in which the agent makes a mistake, where T' is the number of remaining steps. And if in time step t the decoding function makes a mistake so that $\hat{f}(x_t) = s'_t \neq s_t$, it is just like that there is an "adversary" corrupting this step by replacing s_t with s'_t and $r(s'_t, a_t)$ with $\tilde{r}(s'_t, a_t) = r(s_t, a_t)$. Then the problem reduces to a special case of tabular reinforcement learning with $2\epsilon_{\mathrm{f}}T' + \sqrt{2T' \ln \frac{\delta}{2}}$ adversarial corruptions. Thus CR-MVP guarantees that we can learn the underlying MDP with low regret. By setting $\epsilon_{\mathrm{f}} = O(K^{-1/2})$, Theorem 3 shows that Algorithm 2 ensures $O(\sqrt{K})$ regret after running K episodes, which is optimal in terms of K.

Theorem 3. *In Algorithm 2, assume that $K > \frac{M^2A^2}{H\mu_{\min}^3\gamma^2} \log\left(\frac{|\mathcal{G}|H}{\delta}\right) \cdot \min\left\{\frac{\mu_{\min}^3\gamma}{100M^4A^3}, \frac{\delta}{100HN_\phi}\right\}^{-1}$. Set $\epsilon_{\mathrm{f}} = \sqrt{\frac{M^2A^2}{\mu_{\min}^3\gamma^2 T} \log\left(\frac{|\mathcal{G}|H}{\delta}\right)}$, $N_\phi = \Theta\left(\frac{MA}{\mu_{\min}} \log\left(\frac{MH}{\delta}\right)\right)$ and $N_{\mathrm{g}} = \Omega\left(\frac{M^3A^3}{\epsilon_{\mathrm{f}}\mu_{\min}^3\gamma^2} \log\left(\frac{|\mathcal{G}|H}{\delta}\right)\right)$. Then with probability at least $1 - O(\delta)$, the regret of Algorithm 1 is upper bounded by $\tilde{O}\left(\frac{H^{3/2}M^2A^2\sqrt{K}}{\mu_{min}^{3/2}\gamma} + \mathrm{poly}(H, M, A, \mu_{min}^{-1}, \gamma^{-1})\right)$.*

The proof of Theorem 3 is in Appendix B. Note that the regret is proportional to $\sqrt{K}$, which is optimal. Also, Algorithm 2 is computationally efficient if we have an access to a least square oracle. Previous algorithms such as (Dong et al., 2020) is not computationally efficient. Although the AVE algorithm in (Dong et al., 2020) can deal with general low Bellman rank MDPs, their algorithm relies on an elimination process on a general function class, which is proved to be oracle inefficient (Dann et al., 2018).

7. Conclusion and Future Work

In this paper we consider episodic tabular MDP with adversarial corruptions. We provide a statistically efficient and computationally efficient algorithm, CR-MVP, and derive an upper bound on the regret. We also derive a lower bound, which shows that our algorithm is optimal. As an application, we provide an algorithm in BMDP that achieves $O(\sqrt{K})$ regret and is oracle efficient.

Lastly, we list some future directions. First, our algorithm requires knowledge of C. When C is unknown, it is still unknown what is the optimal regret bound. Second, in the

BMDP setting, our regret bound may not be tight, since we treat the (randomly) mistaken steps as adaptive adversarial corruptions. It remains an open question whether we can develop a corresponding algorithm for stochastic corruptions.

Acknowledgements

This work was supported by National Key R&D Program of China (2018YFB1402600), Key-Area Research and Development Program of Guangdong Province (No. 2019B121204008), BJNSF (L172037) and Beijing Academy of Artificial Intelligence.

References

Agarwal, A., Kakade, S., Krishnamurthy, A., and Sun, W. Flambe: Structural complexity and representation learning of low rank mdps. *Advances in Neural Information Processing Systems*, 33, 2020.

Agrawal, S. and Jia, R. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. *Advances in Neural Information Processing Systems*, 30: 1184–1194, 2017.

Azar, M. G., Osband, I., and Munos, R. Minimax regret bounds for reinforcement learning. *arXiv preprint arXiv:1703.05449*, 2017.

Bartlett, P. L. and Tewari, A. Regal: A regularization based algorithm for reinforcement learning in weakly communicating mdps. *arXiv preprint arXiv:1205.2661*, 2012.

Brafman, R. I. and Tennenholtz, M. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct): 213–231, 2002.

Bubeck, S. and Cesa-Bianchi, N. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721*, 2012.

Cai, Q., Yang, Z., Jin, C., and Wang, Z. Provably efficient exploration in policy optimization. In *International Conference on Machine Learning*, pp. 1283–1294. PMLR, 2020.

Dann, C. and Brunskill, E. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2818–2826, 2015.

Dann, C., Lattimore, T., and Brunskill, E. Unifying pac and regret: Uniform pac bounds for episodic reinforcement learning. *Advances in Neural Information Processing Systems*, 30:5713–5723, 2017.

Dann, C., Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. On oracle-efficient pac rl with rich observations. In *Advances in neural information processing systems*, pp. 1422–1432, 2018.

Dann, C., Li, L., Wei, W., and Brunskill, E. Policy certificates: Towards accountable reinforcement learning. In *International Conference on Machine Learning*, pp. 1507–1516. PMLR, 2019.

Dong, K., Wang, Y., Chen, X., and Wang, L. Q-learning with ucb exploration is sample efficient for infinite-horizon mdp. *arXiv preprint arXiv:1901.09311*, 2019.

Dong, K., Peng, J., Wang, Y., and Zhou, Y. Root-n-regret for learning in markov decision processes with function approximation and low bellman rank. In *Conference on Learning Theory*, pp. 1554–1557. PMLR, 2020.

Du, S. S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudík, M., and Langford, J. Provably efficient rl with rich observations via latent state decoding. *arXiv preprint arXiv:1901.09018*, 2019.

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634, 2018.

Feng, F., Wang, R., Yin, W., Du, S. S., and Yang, L. Provably efficient exploration for reinforcement learning using unsupervised learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Fruit, R., Pirotta, M., and Lazaric, A. Near optimal exploration-exploitation in non-communicating markov decision processes. In *Advances in Neural Information Processing Systems*, pp. 2994–3004, 2018.

Gupta, A., Koren, T., and Talwar, K. Better algorithms for stochastic bandits with adversarial corruptions. *arXiv preprint arXiv:1902.08647*, 2019.

Huang, S., Papernot, N., Goodfellow, I., Duan, Y., and Abbeel, P. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

Jaksch, T., Ortner, R., and Auer, P. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(4), 2010.

Jiang, N. and Agarwal, A. Open problem: The dependence of sample complexity lower bounds on planning horizon. In *Conference On Learning Theory*, pp. 3395–3398, 2018.

Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. Contextual decision processes with low bellman rank are pac-learnable. In *International Conference on Machine Learning*, pp. 1704–1713. PMLR, 2017.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. Is q-learning provably efficient? In *Advances in neural information processing systems*, pp. 4863–4873, 2018.

Jin, C., Jin, T., Luo, H., Sra, S., and Yu, T. Learning adversarial mdps with bandit feedback and unknown transition. *arXiv preprint arXiv:1912.01192*, 2019.

Kakade, S. M. et al. *On the sample complexity of reinforcement learning*. PhD thesis, University of London London, England, 2003.

Kearns, M. and Singh, S. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

Kober, J., Bagnell, J. A., and Peters, J. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Kolter, J. Z. and Ng, A. Y. Near-bayesian exploration in polynomial time. In *Proceedings of the 26th annual international conference on machine learning*, pp. 513–520, 2009.

Krishnamurthy, A., Agarwal, A., and Langford, J. Pac reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, pp. 1840–1848, 2016.

Lattimore, T. and Hutter, M. Pac bounds for discounted mdps. In *International Conference on Algorithmic Learning Theory*, pp. 320–334. Springer, 2012.

Lykouris, T., Mirrokni, V., and Paes Leme, R. Stochastic bandits robust to adversarial corruptions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 114–122, 2018.

Lykouris, T., Simchowitz, M., Slivkins, A., and Sun, W. Corruption robust exploration in episodic reinforcement learning. *arXiv preprint arXiv:1911.08689*, 2019.

Ma, Y., Zhang, X., Sun, W., and Zhu, J. Policy poisoning in batch reinforcement learning and control. In *Advances in Neural Information Processing Systems*, pp. 14570–14580, 2019.

Misra, D., Henaff, M., Krishnamurthy, A., and Langford, J. Kinematic state abstraction and provably efficient rich-observation reinforcement learning. In *International conference on machine learning*, pp. 6961–6971. PMLR, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Neu, G. and Pike-Burke, C. A unifying view of optimism in episodic reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Newville, M., Stensitzki, T., Allen, D. B., Rawlik, M., Ingargiola, A., and Nelson, A. Lmfit: Non-linear least-square minimization and curve-fitting for python. *Astrophysics Source Code Library*, pp. ascl–1606, 2016.

Osband, I. and Van Roy, B. On lower bounds for regret in reinforcement learning. *arXiv preprint arXiv:1608.02732*, 2016.

Osband, I. and Van Roy, B. Why is posterior sampling better than optimism for reinforcement learning? In *International Conference on Machine Learning*, pp. 2701–2710, 2017.

Osband, I., Russo, D., and Van Roy, B. (more) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*, pp. 3003–3011, 2013.

Pacchiano, A., Ball, P., Parker-Holder, J., Choromanski, K., and Roberts, S. On optimism in model-based reinforcement learning. *arXiv preprint arXiv:2006.11911*, 2020.

Russo, D. Worst-case regret bounds for exploration via randomized value functions. In *Advances in Neural Information Processing Systems*, pp. 14433–14443, 2019.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Simchowitz, M. and Jamieson, K. G. Non-asymptotic gap-dependent regret bounds for tabular mdps. In *Advances in Neural Information Processing Systems*, pp. 1153–1162, 2019.

Strehl, A. L. and Littman, M. L. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Strehl, A. L., Li, L., Wiewiora, E., Langford, J., and Littman, M. L. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pp. 881–888, 2006.

Szita, I. and Szepesvári, C. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *ICML*, 2010.

Talebi, M. S. and Maillard, O.-A. Variance-aware regret bounds for undiscounted reinforcement learning in mdps. *arXiv preprint arXiv:1803.01626*, 2018.

Wang, R., Du, S. S., Yang, L., and Kakade, S. Is long horizon rl more difficult than short horizon rl? *Advances in Neural Information Processing Systems*, 33, 2020.

Yang, H., Liu, X.-Y., Zhong, S., and Walid, A. Deep reinforcement learning for automated stock trading: An ensemble strategy. *Available at SSRN*, 2020a.

Yang, K., Yang, L. F., and Du, S. S. q-learning with logarithmic regret. *arXiv preprint arXiv:2006.09118*, 2020b.

Zhang, Z. and Ji, X. Regret minimization for reinforcement learning by evaluating the optimal bias function. In *Advances in Neural Information Processing Systems*, pp. 2827–2836, 2019.

Zhang, Z., Ji, X., and Du, S. S. Is reinforcement learning more difficult than bandits? a near-optimal algorithm escaping the curse of horizon. *arXiv preprint arXiv:2009.13503*, 2020a.

Zhang, Z., Zhou, Y., and Ji, X. Almost optimal model-free reinforcement learning via reference-advantage decomposition. *arXiv preprint arXiv:2004.10019*, 2020b.

Zimmert, J. and Seldin, Y. An optimal algorithm for stochastic and adversarial bandits. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 467–475. PMLR, 2019.