Fast multi-agent temporal-difference learning via homotopy stochastic primal-dual method

Dongsheng Ding
University of Southern California
Los Angles, CA, USA
dongshed@usc.edu

Xiaohan Wei
Facebook Inc.
Menlo Park, CA, USA
xiaohanw@usc.edu

Zhuoran Yang
Princeton University
Princeton, NJ, USA
zy6@princeton.edu

Zhaoran Wang
Northwestern University
Evanston, IL, USA
zhaoranwang@gmail.com

Mihailo. R. Jovanović
University of Southern California
Los Angles, CA, USA
mihailo@usc.edu

Abstract

We study a distributed policy evaluation problem in which a group of agents with jointly observed states and private local actions and rewards collaborate to learn the value function of a given policy via local computation and communication. This problem arises in various large-scale multi-agent systems, including power grids, intelligent transportation systems, wireless sensor networks, and multi-agent robotics. We develop and analyze a new distributed temporal-difference learning algorithm that minimizes the mean-square projected Bellman error. Our approach is based on a stochastic primal-dual method and we improve the best-known convergence rate from $O(1/\sqrt{T})$ to $O(1/T)$, where T is the total number of iterations. Our analysis explicitly takes into account the Markovian nature of the sampling and addresses a broader class of problems than the commonly-used i.i.d. sampling scenario.

1 Introduction

Temporal-difference (TD) learning is a central idea for policy evaluation in modern reinforcement learning (RL) [24]. It was originally proposed in [23, 3, 2], and significant advances in solving problems with large number of states have been made [17, 21]. In this paper, we extend the TD learning to a distributed policy evaluation setting in which a group of agents communicates over an undirected connected graph. While all agents share a joint state, each agent follows a local policy and owns a private local reward. To maximize the total reward which is given by the sum of all local rewards, it is essential to estimate – using only the local data and information exchange between neighbors – performance that each agent achieves if it follows a particular policy. This problem is typically called distributed policy evaluation and it arises in various large-scale multi-agent systems, including power grids [16], intelligent transportation systems [10], wireless sensor networks [19], and multi-agent robotics [9].

Review of the distributed TD learning literature. References [14, 22] consider distributed consensus-based gradient temporal-difference (GTD) algorithms where the reward is global, but actions are local, and show the weak convergence without rate. In [15], an approach based on ordinary differential equations (ODEs) was used to show the asymptotic convergence of a gossip-based TD algorithm, where the reward is also global. When the reward is local, reference [11] studies a distributed TD learning algorithm for minimizing the mean square Bellman error (MSBE) and establish $O(1/\sqrt{T})$ convergence rate. Recent references [5, 6] improve the rate to $O(1/T)$. Since

Optimization Foundations for Reinforcement Learning Workshop at NeurIPS 2019, Vancouver, Canada.

most TD algorithms do not converge to the minimum of the MSBE, [25] proposes the mean square projected Bellman error (MSPBE) as the minimization objective. In the off-line scenario, [29, 4] present a batch consensus-based primal-dual gradient algorithm for minimizing a sampled version of MSPBE.

Apart from [15, 6], all other results utilize the i.i.d. state sampling in policy evaluation. In practical RL settings, this assumption is overly restrictive due to the Markovian nature of state trajectory samples. Therefore, it is an open question on how to design an online-type distributed learning algorithm for the policy evaluation (e.g., MSPBE minimization) in the Markovian setting. For the importance of such distributed learning algorithms, see a distributed variant of the policy gradient theorem [34].

Summary of our contributions. We consider a distributed policy evaluation problem where all agents observe joint state trajectories, but the rewards and actions of each agent are private. We propose a new distributed temporal-difference learning algorithm, namely distributed homotopy primal-dual algorithm, that minimizes the mean-square projected Bellman error (MSPBE). We establish the optimal convergence rate $O(1/T)$, where T is the number of iterations. The convergence analysis explicitly takes into account the Markovian nature of samples, thereby implying that our results are applicable to a broader class of problems than previous works.

We cast the MSPBE minimization as a stochastic primal-dual optimization problem with the objective function which is convex in primal variables and strongly-concave in dual variables. This formulation has three benefits. First, since the primal-dual objective depends on expectations linearly, it is more convenient to compute an unbiased estimate from samples. Second, the primal-dual formulation allows the distributed dual averaging type analysis and it allows us to quantify influence of the network size and topology. Third, the primal objective is exactly the original MSPBE that is strongly-convex. Since the convergence of the objective implies that of the iteration, the restarting scheme (i.e., homotopy method) can be used to achieve fast convergence. The established convergence rate $O(1/T)$ demonstrates that distributed convex-concave saddle point programs can be solved with a fast convergence rate.

2 Problem formulation

2.1 Multi-agent Markov decision process (MDP)

We consider an MDP with N agents over an undirected network $\mathcal{G} = (\mathcal{E}, \mathcal{V})$, where $\mathcal{V} := \{1, \cdots, N\}$ denotes the set of nodes and $\mathcal{E}$ is the set of edges. Let $\mathcal{S} := \mathcal{S}_1 \times \cdots \times \mathcal{S}_N$ be the state space and $\mathcal{A} := \mathcal{A}_1 \times \cdots \times \mathcal{A}_N$ be the joint action space. Let $\mathcal{P}^a = [\mathcal{P}^a_{s,s'}]_{s,s' \in \mathcal{S}}$ be the probability transition matrix under a joint action $a \in \mathcal{A}$, where $\mathcal{P}^a_{s,s'}$ is the transition probability from s to s'. Let $\mathcal{R}_j(s,a)$ be the local reward received by agent j for the pair (s,a). The multi-agent MDP can be represented as $\left(\mathcal{S}, \mathcal{A}, \mathcal{P}^a, \{\mathcal{R}_j\}_{j=1}^N, \gamma\right)$ where $\gamma \in (0,1)$ is the discount factor.

When the state, actions, and rewards are globally observable, the multi-agent MDP simplifies to a single-agent MDP. However, this is not the case in many network applications (e.g., [9, 16, 11]) where both actions a_j and rewards $\mathcal{R}_j(s,a)$ of each agent are private. Since every agent can communicate with their neighbors over the graph $\mathcal{G}$, it is crucial to extend single-agent TD learning algorithms to a setup in which only local information exchange is available. Here, we consider a cooperative learning task for agents to maximize the total reward $(1/N)\sum_{j=1}^N \mathcal{R}_j(s,a)$. Let π: $\mathcal{S} \times \mathcal{A} \to [0,1]$ be a joint policy which specifies the probability to take an action $a \in \mathcal{A}$ at state $s \in \mathcal{S}$. We define the global reward at state $s \in \mathcal{S}$ under π to be the expected value of the average of all local rewards,

$$R^\pi_c(s) \;=\; \frac{1}{N}\sum_{j=1}^N R^\pi_j(s), \;\; R^\pi_j(s) := \mathbb{E}_{a\sim\pi(\cdot|s)}\left[\mathcal{R}_j(s,a)\right]. \tag{1}$$

For any fixed joint policy π, the multi-agent MDP becomes a Markov chain over $\mathcal{S}$ with the probability transition matrix $\mathbf{P}^\pi$, where the (s,s')-element of $\mathbf{P}^\pi$ is given by $\mathbf{P}^\pi_{s,s'} \;=\; \sum_{a\in\mathcal{A}} \pi(a|s)\,\mathcal{P}^a_{s,s'}$, and $\pi(a|s)$ is the conditional probability of taking action a given state s. For the existence of stationary distribution, *we assume that such a Markov chain is aperiodic and irreducible*. This ensures that the Markov chain converges to the unique stationary distribution Π with a geometric rate [12].

2.2 Policy evaluation

Let the value function of a policy π, $V^\pi\colon\ \mathcal{S} \to \mathbb{R}$, be defined as the expectation $V^\pi(s) = \mathbb{E}[\sum_{p=0}^{\infty} \gamma^p \mathcal{R}_c^\pi(s_p) \,|\, s_0 = s, \pi]$ where $s_0 = s$ is the initial state. If we arrange $V^\pi(s)$ and $R_c^\pi(s)$ over all states $s \in \mathcal{S}$ into the vectors $\mathbf{V}^\pi$ and $\mathbf{R}_c^\pi$, the Bellman equation for $\mathbf{V}^\pi$ can be written as

$$\mathbf{V}^\pi \ = \ \mathbf{R}_c^\pi \ + \ \gamma \mathbf{P}^\pi \mathbf{V}^\pi. \tag{2}$$

Since it is impossible to evaluate $\mathbf{V}^\pi$ directly for a large state space, we approximate $V^\pi(s)$ using a family of linear functions $\{V_x(s) \ = \ \phi^T(s)x, \ x \in \mathbb{R}^d\}$, where $x \in \mathbb{R}^d$ is the vector of unknown parameters and $\phi(s)\colon\ \mathcal{S} \to \mathbb{R}^d$ is a known dictionary consisting of d features. If we arrange $\{V_x(s)\}_{s\in\mathcal{S}}$ into the vector $\mathbf{V}_x \in \mathbb{R}^{|\mathcal{S}|}$, we have $\mathbf{V}_x := \Phi x$ where the ith row of the matrix $\Phi \in \mathbb{R}^{|\mathcal{S}|\times d}$ is given by $\phi^T(s_i)$. We choose Φ to be the full column rank matrix.

The goal of policy evaluation now becomes to determine the vector x that minimizes the mean square Bellman error (MSBE) [26], $\|\mathbf{V}_x - \gamma \mathbf{P}^\pi \mathbf{V}_x - \mathbf{R}_c^\pi\|_D^2/2$, where $D := \operatorname{diag}\{\Pi(s), s \in \mathcal{S}\} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ is a diagonal matrix determined by the stationary distribution Π. As discussed in [25], the solution to the fixed point problem $\mathbf{V}_x = \gamma \mathbf{P}^\pi \mathbf{V}_x + \mathbf{R}_c^\pi$ may not exist because the right-hand-side may not stay in the column space of the matrix Φ. To address this issue, the GTD algorithm [25] proposes to minimize the mean square projected Bellman error (MSPBE), $\|P_\Phi(\mathbf{V}_x - \gamma \mathbf{P}^\pi \mathbf{V}_x - \mathbf{R}_c^\pi)\|_D^2/2$, via stochastic-gradient-type updates, where $P_\Phi := \Phi(\Phi^T D \Phi)^{-1}\Phi^T D$ is a projection operator onto the column subspace of Φ. We express MSPBE in the following quadratic form,

$$f(x) \ = \ \tfrac{1}{2} \left\| \Phi^T D \left(\mathbf{V}_x - \gamma \mathbf{P}^\pi \mathbf{V}_x - \mathbf{R}_c^\pi \right) \right\|^2_{(\Phi^T D \Phi)^{-1}} = \tfrac{1}{2} \|Ax - b\|^2_{C^{-1}} \tag{3}$$

where $A := \mathbb{E}_{s\sim\Pi}[\phi(s)(\phi(s) - \gamma\phi(s'))^T]$, $C := \mathbb{E}_{s\sim\Pi}[\phi(s)\phi(s)^T]$, and $b := \mathbb{E}_{s\sim\Pi}[\mathcal{R}_c^\pi(s)\phi(s)]$. It has been shown in [3, page 300] that full column rank Φ yields full rank A, that C is a positive definite matrix, and that (3) has a unique minimizer.

2.3 Decentralized convex-concave primal-dual optimization

Since $b = (1/N)\sum_{j=1}^N b_j$ with $b_j = \mathbb{E}_{s\sim\Pi}\left[\mathcal{R}_j^\pi(s)\phi(s)\right]$, the problem of minimizing objective function (3) can be written as

$$\underset{x \in \mathcal{X}}{\text{minimize}} \ \frac{1}{N} \sum_{j=1}^N f_j(x), \ \ f_j(x) := \tfrac{1}{2} \|Ax - b_j\|^2_{C^{-1}}, \tag{4}$$

where $\mathcal{X}$ is a compact convex subset of $\mathbb{R}^d$. Problem (4) is a distributed stochastic optimization problem with N private stochastic objectives f_j that involve products and inverses of the expectations. This unique feature of MSPBE makes it challenging to obtain an unbiased estimator of the objective from a few state samples, and it is not encountered in typical distributed optimization settings [18, 8]. As shown in [13, 30, 29], the Fenchel dual of (4) can be used to express each objective f_j as

$$f_j(x) \ = \ \max_{y_j \in \mathcal{Y}} \ \psi_j(x, y_j), \ \ \psi_j(x, y_j) \ := \ y_j^T(Ax - b_j) \ - \ \tfrac{1}{2} \, y_j^T C y_j, \tag{5}$$

where y_j is a dual variable and $\mathcal{Y} \subseteq \mathbb{R}^d$ is a convex compact set such that $C^{-1}(Ax - b_j) \in \mathcal{Y}$ for all $x \in \mathcal{X}$. Since C is positive definite and $\mathcal{X}$ is compact, $\mathcal{Y}$ exists. Thus, (4) can be cast as a distributed stochastic saddle point problem in which we only need samples of the problem data A, C, and b.

More abstractly, we are interested in a stochastic saddle point problem with the objective function,

$$\frac{1}{N} \sum_{j=1}^N \psi_j(x, y_j) \ := \ \frac{1}{N} \sum_{j=1}^N \mathbb{E}_{\xi\sim\Pi}[\Psi_j(x, y_j; \xi)], \tag{6}$$

where $\Psi_j(x, y_j; \xi)$ is a stochastic function with a random variable ξ which is distributed according to the stationary distribution Π, $x \in \mathcal{X}$ is the primal variable, and $y = (y_1, \ldots, y_N)$ is the dual variable with $y_j \in \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact. Assumption 3.2 highlights the lack of strong convexity of the function $\psi_j(x, y_j)$ defined in (5).

In our multi-agent MDP setting, the stationary distribution Π is unknown. Each agent receives samples ξ_t from a Markov process whose state distribution at time t is P_t, where P_t converges to Π

geometrically. Therefore, i.i.d. samples from the stationary distribution Π are not available. This is a typical ergodic setting in the classical stochastic optimization [7] and a recent application of the centralized GTD can be found in [30]. We are particularly interested in designing and analyzing distributed algorithms for stochastic saddle point problem (6) in the ergodic setting.

3 Algorithm and convergence result

3.1 Distributed homotopy primal-dual algorithm (DHPD)

Algorithm 1 Distributed Homotopy Primal-Dual (DHPD): (T_1, η_1, k)

Initialization: $x_{j,1}(1) = x'_{j,1}(1) = 0, y_{j,1}(1) = y'_{j,1}(1) = 0,\ \forall j \in \mathcal{V}$, and η_1, T_1, k

For $k = 1$ to k **do**

1. **For** $t = 1$ to $T_k - 1$ **do**

   - Primal update: For all agents $j \in \mathcal{V}$,

$$x'_{j,k}(t+1) \;=\; \sum_{i=1}^{N} W_{ij}\, x'_{i,k}(t) \;-\; \eta_k\, G_{j,x}(x_{j,k}(t), y_{j,k}(t); \xi_k(t))$$
$$x_{j,k}(t+1) \;=\; \mathcal{P}_{\mathcal{X}}(x'_{j,k}(t+1))$$

   - Dual update: For all agents $j \in \mathcal{V}$,

$$y'_{j,k}(t+1) \;=\; y'_{j,k}(t) \;+\; \eta_k\, G_{j,y}(x_{j,k}(t), y_{j,k}(t); \xi_k(t))$$
$$y_{j,k}(t+1) \;=\; \mathcal{P}_{\mathcal{Y}}(y'_{j,k}(t+1))$$

   end for

2. $x_{j,k+1}(1) = \dfrac{1}{T_k}\displaystyle\sum_{t=1}^{T_k} x_{j,k}(t),\ y_{j,k+1}(1) = \dfrac{1}{T_k}\sum_{t=1}^{T_k} y_{j,k}(t)$
3. $x'_{j,k+1}(1) = x_{j,k+1}(1),\ y'_{j,k+1}(1) = y_{j,k+1}(1)$
4. $\eta_{k+1} = \eta_k/2,\ T_{k+1} = 2T_k$

end for

Output for the jth agent: $\hat{x}_{j,K} := \dfrac{1}{T_K}\displaystyle\sum_{t=1}^{T_K} x_{j,K}(t)$ and $\hat{y}_{j,K} = \dfrac{1}{T_K}\sum_{t=1}^{T_K} y_{j,K}(t)$.

Let W be a doubly stochastic mixing matrix over the graph $\mathcal{G}$ and let $\mathcal{P}_{\mathcal{X}}(\cdot)$ and $\mathcal{P}_{\mathcal{Y}}(\cdot)$ be projections onto $\mathcal{X}$ and $\mathcal{Y}$. In Algorithm 1, we describe a distributed homotopy primal-dual (DHPD) method for computing the saddle-point of the function given in (6). The initial learning rate is η_1, the number of total rounds is k, the number of inner iterations in the first round is T_1, and the number of iterations doubles for subsequent rounds. Within round k, every agent i performs an inner loop update with T_k iterations, indexed by time t. At round k and time t, the random variable of the underlying Markov process is $\xi_k(t)$. Every agent j updates a pair of local primal-dual variables $z_{j,k}(t) := (x_{j,k}(t), y_{j,k}(t))$ using stochastic information

$$G_j(z_{j,k}(t); \xi_k(t)) \;:=\; \begin{bmatrix} G_{j,x}(z_{j,k}(t); \xi_k(t)) \\ G_{j,y}(z_{j,k}(t); \xi_k(t)) \end{bmatrix} \;=\; \begin{bmatrix} \nabla_x \Psi_j(z_{j,k}(t); \xi_k(t)) \\ \nabla_y \Psi_j(z_{j,k}(t); \xi_k(t)) \end{bmatrix}.$$

After each round k, every agent j maintains $\hat{x}_{j,k} = \frac{1}{T_k}\sum_{t=1}^{T_k} x_{j,k}(t)$ and $\hat{y}_{j,k} = \frac{1}{T_k}\sum_{t=1}^{T_k} y_{j,k}(t)$. At round $k+1$, we initialize primal and dual updates using previous values of $\hat{x}_{j,k}$ and $\hat{y}_{j,k}$. We then reduce the learning rate by half, $\eta_{k+1} = \eta_k/2$, and set the number of the inner loop iterations to $T_{k+1} = 2T_k$. This scheme of adaptively restarting the algorithm is typically referred to as *homotopy method*. As shown in [28, 33, 32, 31], faster convergence rates can be achieved by combing the homotopy method with classical algorithms. To the best of our knowledge, our work is the first to exploit the homotopy method to solve distributed stochastic saddle point programs with a convergence rate better than $O(1/\sqrt{T})$.

Let $(x^\star, y^\star)$ be a saddle point of problem (6) where $y^\star = (y_1^\star, \ldots, y_N^\star)$, and the corresponding $x^\star$ is the solution to (4). The (global) optimality gap for the ith agent at $\hat{x}_{i,k}$ is given by

$$\varepsilon(\hat{x}_{i,k}) \;:=\; f(\hat{x}_{i,k}) \;-\; f(x^\star) \;=\; \frac{1}{N}\sum_{j=1}^{N}\left(f_j(\hat{x}_{i,k}) \;-\; f_j(x^\star)\right). \tag{7}$$

3.2 Assumptions

We formally state assumptions required to establish the convergence rate for Algorithm 1.

Assumption 3.1 (Convex compact domain)**.** *The feasible sets $\mathcal{X}$ and $\mathcal{Y}$ contain the origin in $\mathbb{R}^d$ and they are convex and compact with radius $R > 0$, i.e., $\sup_{x\in\mathcal{X}, y\in\mathcal{Y}} \|(x,y)\|^2 \le R^2$.*

We make the following assumption on f_j and ψ_j that does not rely on choices of (4) and (5).

Assumption 3.2 (Convexity and concavity)**.** *The function $\psi_j(x, y_j)$ is convex in x for any fixed $y_j \in \mathcal{Y}$, and is strongly concave in y_j for any fixed $x \in \mathcal{X}$, i.e., there exists $\rho_y > 0$ such that*

$$\begin{aligned}
&\psi_j(x,y_j) \;\ge\; \psi_j(x',y_j) \;+\; \langle \nabla_x \psi_j(x',y_j), x - x'\rangle\,,\; \forall x,x' \in \mathcal{X}, y_j \in \mathcal{Y},\\
&\psi_j(x,y_j) \;\le\; \psi_j(x,y_j') \;-\; \left\langle \nabla_y \psi_j(x,y_j'), y_j - y_j'\right\rangle \;-\; \frac{\rho_y}{2}\|y_j - y_j'\|^2,\; \forall y_j, y_j' \in \mathcal{Y}, x \in \mathcal{X}.
\end{aligned}$$

Moreover, $f_j(x) = \max_{y_j\in\mathcal{Y}} \psi_j(x,y_j)$ is strongly convex, i.e., there exists $\rho_x > 0$ such that

$$f_j(x) \;\ge\; f_j(x') \;+\; \langle \nabla f_j(x'), x - x'\rangle \;+\; \frac{\rho_x}{2}\|x - x'\|^2,\; \forall x, x' \in \mathcal{X}.$$

Assumption 3.3 (Bounded gradient)**.** *For any t and k, there exists a positive constant G such that the sampled gradient $G_j(x, y_j; \xi_k(t))$ with probability one we have*

$$\|G_j(x, y_j; \xi_k(t))\| \;\le\; G,\; \forall x \in \mathcal{X}, y_j \in \mathcal{Y}. \tag{8}$$

Assumption 3.4 (Lipschitz gradient)**.** *For any t and k, there exists a positive constant L such that with probability one we have*

$$\begin{aligned}
\|G_j(x, y_j; \xi_k(t)) - G_j(x', y_j; \xi_k(t))\| \;&\le\; L\|x - x'\|,\; \forall x, x' \in \mathcal{X}, y_j \in \mathcal{Y},\\
\|G_j(x, y_j; \xi_k(t)) - G_j(x, y_j'; \xi_k(t))\| \;&\le\; L\|y_j - y_j'\|,\; \forall x \in \mathcal{X}, y_j, y_j' \in \mathcal{Y}.
\end{aligned} \tag{9}$$

We recall some important concepts from probability theory. The total variation distance between distributions P and Q on a set $\Xi \subseteq \mathbb{R}^{|S|}$ is given by $d_{\text{tv}}(P,Q) := \int_\Xi |p(\xi) - q(\xi)| d\mu(\xi) = 2\sup_{A\subset\Xi} |P(A) - Q(A)|$, where P and Q are continuous in the Lebesgue measure μ, whose densities p and q exist, and the supremum is taken over all measurable subsets of Ξ. The mixing time measures how fast a sequence of probability measures generated by a Markovian process converge to its (unique) stationary distribution Π, whose density π is assumed to exist. Let $\mathcal{F}_{k,t}$ be the σ-field generated by the first t samples at round k, $\xi_{k,1}, \ldots, \xi_{k,t}$, drawn from $P_{k,1}, \ldots, P_{k,t}$, where $P_{k,t}$ is the probability measure of the Markovian process at time t and round k. Let $P_{k,t}^{[s]}$ be the distribution of $\xi_{k,t}$ conditioned on $\mathcal{F}_{k,s}$ (i.e., given samples up to time slot s: $\xi_{k,1}, \ldots, \xi_{k,s}$) at round k, whose density $p_{k,t}^{[s]}$ also exists. The mixing time is defined for a Markovian process as follows.

Definition 3.1. *[7] The total variation mixing time $\tau_{\text{tv}}(P_k^{[s]}, \varepsilon)$ of the Markovian process conditioned on the σ-field of the initial s samples $\mathcal{F}_{k,s} = \sigma(\xi_{k,1}, \ldots, \xi_{k,s})$ is the smallest $t \in \mathbb{N}$ such that $d_{\text{tv}}(P_{k,s+t}^{[s]}, \Pi) \le \varepsilon$, namely, $\tau_{\text{tv}}(P_k^{[s]}, \varepsilon) := \inf\{t - s : t \in \mathbb{N}, \int_\Xi |p_{k,t}^{[s]}(\xi) - \pi(\xi)| d\mu(\xi) \le \varepsilon\}$.*

The mixing time $\tau_{\text{tv}}(P_k^{[s]}, \varepsilon)$ measures the number of additional steps required until the distribution of $\xi_{k,t}$ is within ε neighborhood of the stationary distribution Π given the initial s samples. For our multi-agent MDP setting, since the underlying Markov chain is irreducible and aperiodic, there exists $\Gamma \ge 1$ and $\rho \in (0,1)$ such that $\mathbb{E}\left[d_{tv}(P_{k,t+\tau}^{[t]}, \Pi)\right] \le \Gamma\rho^\tau$ for all $\tau \in \mathbb{N}$ and all k [12]. Furthermore, for any $\epsilon > 0$, the mixing time property satisfies that

$$\tau_{tv}(P_k^{[s]}, \varepsilon) \;\le\; \left\lceil \frac{\log \frac{\Gamma}{\varepsilon}}{|\log \rho|} \right\rceil + 1,\; \forall k, s \in \mathbb{N}. \tag{10}$$

3.3 Convergence result

In Theorem 3.1, we establish the convergence rate of Algorithm 1 for solving the stochastic saddle point problem with the objective function given in (6); see supplementary material in Section 5 for proof. The total number of iterations in Algorithm 1 is $T := \sum_{k=1}^{k} T_k = (2^k - 1)T_1$.

Theorem 3.1. *Let Assumptions 3.1, 3.2, 3.3, and 3.4 hold. For any $\eta_1 \geq 1/(4/\rho_y + 2/\rho_x)$, any T_1 and k satisfying*

$$T_1 \geq \tau := 1 + \log(\Gamma T)/|\log \rho|, \tag{11}$$

the output $\hat{x}_{j,k}$ of Algorithm 1 provides the solution to problem (4) with the convergence rate,

$$\bar{\varepsilon}_k \;:=\; \frac{1}{N}\sum_{j=1}^{N} \mathbb{E}[\varepsilon(\hat{x}_{j,k})] \;\leq\; C_1 \frac{G(RL+G)\log^2(\sqrt{N}T)}{T(1-\sigma_2(W))} \;+\; C_2 \frac{G(G+RL)(1+T_1)}{T}, \tag{12}$$

where the optimality gap $\varepsilon(\hat{x}_{j,k})$ is defined in (7), C_1, C_2 are constants independent of T, $\sigma_2(W)$ is the second largest eigenvalue of W, and N is the total number of agents.

We next briefly comment on the result established in Theorem 3.1. (1) **The MSPBE minimization.** For $\psi_j(x, y_j)$ given by (5), Assumptions 3.1-3.4 hold with $\rho_x = 2\lambda + \sigma_{\max}^2(A)/\sigma_{\min}(C)$, $\rho_y = \sigma_{\min}(C)$, $G \geq \sqrt{(2\beta_1^2 + \beta_2^2 + 4\lambda^2)R^2 + \beta_0^2}$, and $L \geq \max(\sqrt{\beta_1^2 + \beta_2^2}, \sqrt{4\lambda^2 + \beta_1^2})$ where constants β_0, β_1 and β_2 provide upper bounds to $\beta_0 \geq \|\mathcal{R}_j^\pi(s)\phi(s)\|$, $\beta_1 \geq \|\phi(s)(\phi(s) - \gamma\phi(s'))\|$, and $\beta_2 \geq \|\phi(s)\phi(s)^T\|$. (2) **The optimal convergence rate.** For $T > \tau$, one can always find T_1 and k such that condition (11) holds. For instance, choosing $T_1 = \tau$ and $k = \log(1 + T/\tau)$ yields the convergence rate $O(\log^2(\sqrt{N}T)/T)$, i.e., it achieves the optimal rate $O(1/T)$ for stochastic optimization [1] up to a logarithmic factor. (3) **The mixing time.** The constant τ is the upper bound of mixing time for $\varepsilon = 1/T$ in (10). The bound in Theorem 3.1 is governed by how fast the process $P_k^{[s]}$ reaches $1/T$ mixing. (4) **Influence of the network size and topology.** The factor $\log^2(\sqrt{N}T)/(1 - \sigma_2(W))$ quantifies the dependence on the network size N and the topology of W.

4 Computational experiments

We conduct a computational experiment on the example of Mountain Car Task in [24]. We generate the dataset following the approach of [29], obtain a policy by running Sarsa with $d = 300$ features, and sample the trajectories of states and actions according to the policy. We simulate the Erdős-Rényi network with size N and connectivity 0.1. For every sample, each agent observes a local reward that is a random fraction of the total reward.

We compare Algorithm 1 (i.e., DHPD) with stochastic primal-dual (SPD) algorithm under different settings. For $N = 1$, SPD corresponds to GTD in [13, 30, 27], and for $N > 1$, SPD corresponds to multi-agent GTD [11]. We show computational results in Figure 1 for $\lambda = 0$ and $\eta_1 = 0.1$. We see that our algorithm converges faster than SPD in all cases. For detailed setups and additional computational results, see subsection 5.4 in the supplemental material.

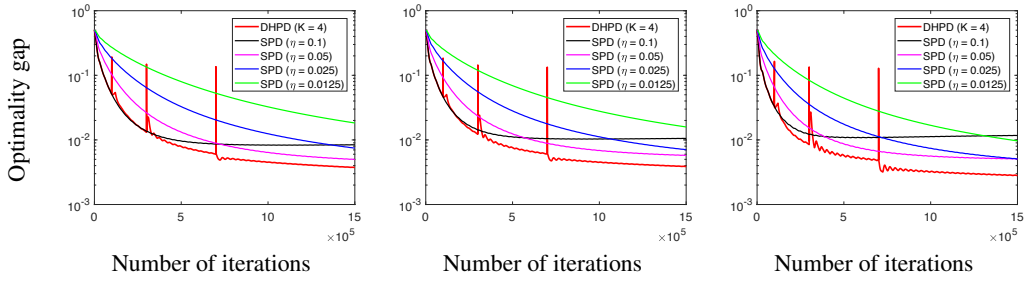

Figure 1: Experimental results. Left: $N = 1$. Middle: $N = 10$. Right: $N = 100$.

References

[1] A. Agarwal, M. J. Wainwright, P. L. Bartlett, and P. K. Ravikumar. Information-theoretic lower bounds on the oracle complexity of convex optimization. In *Advances in Neural Information Processing Systems*, pages 1–9, 2009.

[2] L. Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

[3] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*, volume 5. Athena Scientific Belmont, MA, 1996.

[4] L. Cassano, K. Yuan, and A. H. Sayed. Multi-agent fully decentralized off-policy learning with linear convergence rates. *arXiv preprint arXiv:1810.07792*, 2018.

[5] T. Doan, S. Maguluri, and J. Romberg. Finite-time analysis of distributed TD(0) with linear function approximation on multi-agent reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 1626–1635, 2019.

[6] T. T. Doan, S. T. Maguluri, and J. Romberg. Finite-time performance of distributed temporal difference learning with linear function approximation. *arXiv preprint arXiv:1907.12530*, 2019.

[7] J. C. Duchi, A. Agarwal, M. Johansson, and M. I. Jordan. Ergodic mirror descent. *SIAM Journal on Optimization*, 22(4):1549–1578, 2012.

[8] J. C. Duchi, A. Agarwal, and M. J. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control*, 57(3):592–606, 2012.

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

[10] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis. Multiagent reinforcement learning for urban traffic control using coordination graphs. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 656–671, 2008.

[11] D. Lee, H. Yoon, and N. Hovakimyan. Primal-dual algorithm for distributed reinforcement learning: distributed GTD. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 1967–1972, 2018.

[12] D. A. Levin and Y. Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[13] B. Liu, J. Liu, M. Ghavamzadeh, S. Mahadevan, and M. Petrik. Finite-sample analysis of proximal gradient TD algorithms. In *UAI*, pages 504–513, 2015.

[14] S. V. Macua, J. Chen, S. Zazo, and A. H. Sayed. Distributed policy evaluation under multiple behavior strategies. *IEEE Transactions on Automatic Control*, 60(5):1260–1274, 2014.

[15] A. Mathkar and V. S. Borkar. Distributed reinforcement learning via gossip. *IEEE Transactions on Automatic Control*, 62(3):1465–1470, 2016.

[16] S. Misra, A. Mondal, S. Banik, M. Khatua, S. Bera, and M. S. Obaidat. Residential energy management in smart grid: A Markov decision process-based approach. In *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of things and IEEE Cyber, Physical and Social Computing*, pages 1152–1157, 2013.

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

[18] A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48, 2009.

[19] P. Pennesi and I. C. Paschalidis. A distributed actor-critic algorithm and applications to mobile sensor network coordination problems. *IEEE Transactions on Automatic Control*, 55(2):492–497, 2010.

[20] I. Pinelis. Optimum bounds for the distributions of martingales in banach spaces. *The Annals of Probability*, 22(4):1679–1706, 1994.

[21] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[22] M. S. Stanković and S. S. Stanković. Multi-agent temporal-difference learning with linear function approximation: Weak convergence under time-varying network topologies. In *2016 American Control Conference (ACC)*, pages 167–172, 2016.

[23] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[24] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[25] R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvári, and E. Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 993–1000, 2009.

[26] R. S. Sutton, H. R. Maei, and C. Szepesvári. A convergent $\mathrm{O}(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In *Advances in neural information processing systems*, pages 1609–1616, 2009.

[27] A. Touati, P.-L. Bacon, D. Precup, and P. Vincent. Convergent TREE BACKUP and RETRACE with function approximation. In *International Conference on Machine Learning*, pages 4962–4971, 2018.

[28] K. I. Tsianos and M. G. Rabbat. Distributed strongly convex optimization. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 593–600, 2012.

[29] H.-T. Wai, Z. Yang, Z. Wang, and M. Hong. Multi-agent reinforcement learning via double averaging primal-dual optimization. In *Advances in Neural Information Processing Systems*, pages 9649–9660, 2018.

[30] Y. Wang, W. Chen, Y. Liu, Z.-M. Ma, and T.-Y. Liu. Finite sample analysis of the GTD policy evaluation algorithms in markov setting. In *Advances in Neural Information Processing Systems*, pages 5504–5513, 2017.

[31] X. Wei, H. Yu, Q. Ling, and M. Neely. Solving non-smooth constrained programs with lower complexity than $\mathrm{O}(1/\varepsilon)$: A primal-dual homotopy smoothing approach. In *Advances in Neural Information Processing Systems*, pages 3999–4009, 2018.

[32] Y. Xu, Y. Yan, Q. Lin, and T. Yang. Homotopy smoothing for non-smooth problems with lower complexity than $\mathrm{O}(1/\epsilon)$. In *Advances In Neural Information Processing Systems*, pages 1208–1216, 2016.

[33] T. Yang and Q. Lin. RSG: Beating subgradient method without smoothness and strong convexity. *The Journal of Machine Learning Research*, 19(1):236–268, 2018.

[34] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *International Conference on Machine Learning*, pages 5867–5876, 2018.

[35] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.