No-regret Exploration in Shuffle Private Reinforcement Learning

Shaojie Bai[1,2], Mohammad Sadegh Talebi[2], Chengcheng Zhao[1], Peng Cheng[1], and Jiming Chen[1]

***Abstract*— Differential privacy (DP) has recently been introduced into episodic reinforcement learning (RL) to formally address user privacy concerns in personalized services. Previous work mainly focuses on two trust models of DP: the central model, where a central agent is responsible for protecting users' sensitive data, and the (stronger) local model, where the protection occurs directly on the user side. However, they either require a trusted central agent or incur a significantly higher privacy cost, making it unsuitable for many scenarios. This work introduces a trust model stronger than the central model but with a lower privacy cost than the local model, leveraging the emerging *shuffle* model of privacy. We present the first generic algorithm for episodic RL under the shuffle model, where a trusted shuffler randomly permutes a batch of users' data before sending it to the central agent. We then instantiate the algorithm using our proposed shuffle Privatizer, relying on a shuffle private binary summation mechanism. Our analysis shows that the algorithm achieves a near-optimal regret bound comparable to that of the centralized model and significantly outperforms the local model in terms of privacy cost.**

I. Introduction

Reinforcement learning is a prominent sequential decision-making framework that has gained remarkable attraction in real-world applications across several fields such as healthcare [1], online recommendation [2], and language models [3]. In these applications, the learning agent continuously improves its performance by learning from users' personal feedback, which usually contains sensitive information. Without privacy protection mechanisms in place, the learning agent can memorize information about users' interaction history [4], which makes the learning agent vulnerable to various privacy attacks [5].

Over the past decade, *differential privacy* [6] has been extensively applied in various decision-making settings including multi-armed bandits [7], linear control systems [8], and network systems [9]. In the case of RL, there is a growing literature dealing with privacy issue of the reward function [10], the environment transition [11], and policies [12]. Regarding privacy issues in interaction history, prior work focused on episodic RL in the regret setting under Joint DP (JDP) [13], [14] and Local DP (LDP) constraints [15], [16]. In these settings, each episode is regarded as an interaction with one user, and the aim is to ensure that the user's states and generated rewards will not be inferred by an adversary during the learning process. Specifically speaking, JDP guarantees that all other users' decisions will not leak much information about any specific user; it is known that ε-JDP can be obtained at the expense of an additive logarithmic term in the regret bound [17], $\mathcal{O}(\sqrt{K} + \log(K)/\varepsilon)$ after K episodes. However, a trusted central agent is required to collect the raw interaction history under JDP, which may render it infeasible. On the other hand, ε-LDP provides a stronger privacy guarantee, where each user's raw data is privatized on their local side before being sent to the learning agent. But it leads to a significantly worse regret, $\mathcal{O}(\sqrt{K}/\varepsilon)$ after K episodes [15], which could be unsatisfactory in high privacy regimes, i.e., when ε is chosen small. This naturally leads to the following question: Can a finer trade-off between privacy and regret in RL be achieved, i.e., achieving utility comparable to that of centralized privacy models but without relying on a trusted central agent?

Motivated by these, this paper focuses on the online RL problem under an intermediate trust model of differential privacy, known as the *shuffle differential privacy* (SDP) [18] in the hope of attaining a finer regret-privacy trade-off. In this new trust model, a secure shuffler is assumed between the users and the central agent, which is often implemented by a trusted third party via cryptographic mixnets or trusted hardware [19]. The shuffler permutes a *batch* of users' noisy data before they are viewed by the agent so that it can not distinguish two users' data. The shuffle model provides a stronger privacy guarantee than the central model but usually suffers a smaller cost than the local model, which has achieved a good privacy/utility trade-off in several learning problems such as (federated) supervised learning [20], and bandits [21], [22]. However, it is still not investigated within RL to our best knowledge, due to challenges that arise upon applying algorithmic ideas from simpler decision-making models like bandits. Specifically, the SDP model relies on a batch updating mechanism and data perturbation, which inherently delays exploitation and complicates exploration. This creates many difficulties in the algorithm design regret analysis, making it challenging to develop an effective private RL algorithm that attains sublinear regret in K while ensuring SDP constraints.

Due to the necessity of batch update to ensure SDP, a relevant line of work is multi-batched RL whose primary concern is to reduce update frequency and enhance the efficiency of parallelism and re-deployment, which is often implemented through minimizing the number of policy switches. Algorithms with sublinear regret and policy switches have been devised through adaptive batch selection [23], [24] and static batch selection [25], [26]. The static batch selection approach is more relevant in our context, as the learning agent can decide when to start a new batch before the interactions

[1]College of Control Science and Engineering, Zhejiang University, 310027 Hangzhou, China. Emails: {bai_shaojie, chengchengzhao, lunarheart, cjm}@zju.edu.cn

[2]Department of Computer Science, University of Copenhagen, 2100, Copenhagen, Denmark. Emails: {shaojie.bai, m.shahi}@di.ku.dk

begin, which also meets the privacy requirements. However, the existing methods cannot be directly applied to our problem, since we must control the impact of the privacy perturbation on the learning process.

In this paper, we introduce *Shuffle Differentially Private Policy Elimination* (SDP-PE), a first RL algorithm satisfying the SDP constraint and attaining a sublinear regret. Our contributions are threefold.

1) Generic Algorithm Design: We develop two key strategies to design our private RL algorithm based on the *policy elimination* idea [26]: a) divide exploration into stages with exponentially growing batch size, iteratively update an "absorbing MDP" and an active policy set for policy elimination. b) use only the data from *the current stage* for updating, and forget earlier data to prevent noise accumulation.

2) Novel SDP Privatizer for RL: We design an appropriate SDP Privatizer with desirable properties for required statistics, built on a shuffle private binary summation mechanism. This Privatizer can be directly integrated into the proposed SDP-PE algorithm to preserve privacy and ensure utility.

3) Theoretical Results: We show that SDP-PE algorithm obtains a regret of $\widetilde{\mathcal{O}}\big(\sqrt{X^2AH^5K} + \frac{X^3A^2H^6}{\varepsilon}\big)$,[1] where X, A, K, H, ε represent the number of states, actions, episodes, the episode length, and the privacy budget, respectively. Compared to the optimal result in the local model [17], SDP-PE reduces the dependence on $1/\varepsilon$ from multiplicative to additive. Additionally, SDP-PE matches the best results in the centralized model regarding the dependence on K and ε [17], achieving an improved privacy-regret trade-off.

However, it is worth noting that the dependence on X, A, H remains sub-optimal, and our proposed algorithm is also computationally inefficient – a challenge that remains unresolved even in non-private batched RL. We leave these open problems for future work. We summarize the comparison of best-known results regarding regret under ε-JDP, ε-LDP, and (ε, β)-SDP guarantees in Table I.

TABLE I
RESULTS COMPARISON

Algorithm	Privacy model	Best-known regret bounds
DP-UCBVI [17]	ε-JDP	$\widetilde{\mathcal{O}}\left(\sqrt{XAH^3K} + \frac{X^2AH^3}{\varepsilon}\right)$
	ε-LDP	$\widetilde{\mathcal{O}}\left(\sqrt{XAH^3K} + \frac{X^2AH^3\sqrt{K}}{\varepsilon}\right)$
PBPE (Ours)	(ε, β)-SDP	$\widetilde{\mathcal{O}}\left(\sqrt{X^2AH^5K} + \frac{X^3A^2H^6}{\varepsilon}\right)$

The remainder of this paper is organized as follows. In Section II, we provide the preliminaries of episodic RL and adopt Shuffle DP for RL. Section III shows the designed policy elimination algorithm under SDP constraints. The regret and privacy guarantees are presented in Section IV and Section V. The conclusion is presented in Section VI.

[1]Here $\tilde{\mathcal{O}}(\cdot)$ hides terms that are poly-logarithmic in K.

II. PRELIMINARY

A. Episodic Reinforcement Learning

An episodic Markov decision process (MDP) is defined by a tuple $\big(\mathcal{X}, \mathcal{A}, H, \{P_h\}_{h=1}^H, \{\mathcal{R}_h\}_{h=1}^H, d_1\big)$, where $\mathcal{X}, \mathcal{A}$ are state and action spaces with respective cardinalities X and A, and H is the episode length. At step h, the transition function $P_h(\cdot|x,a)$ takes a state-action pair and returns a distribution over states, the reward distribution $\mathcal{R}_h(x,a)$ is a distribution over $\{0,1\}$ with expectation $r_h(x,a)$, and d_1 is the distribution of initial state.[2] A deterministic policy is defined as a collection $\pi = (\pi_1, \ldots, \pi_H)$ of policies $\pi_h : \mathcal{X} \to \mathcal{A}$. The value function V_h^π and Q function Q_h^π are defined as: $V_h^\pi(x) = \mathbb{E}_\pi[\sum_{t=h}^H r_t | x_h = x], Q_h^\pi(x,a) = \mathbb{E}_\pi[\sum_{t=h}^H r_t | x_h, a_h = x, a], \forall x, a \in \mathcal{X} \times \mathcal{A}$. There exists an optimal deterministic policy $\pi^\star$ such that $V_h^\star(x) = V_h^{\pi^\star}(x) = \max_\pi V_h^\pi(x)$ for all $x, h \in \mathcal{X} \times [H]$. [3] The Bellman (optimality) equation follows $\forall h \in [H] : Q_h^\star(x,a) = r_h(x,a) + \max_{a'} \mathbb{E}_{x' \sim P_h(x,a)}[V_{h+1}^\star(x')]$. The optimal policy is the greedy policy: $\pi_h^\star(x) = \operatorname{argmax}_a Q_h^\star(x,a), \forall x \in \mathcal{X}$. For generalization, we define the value function of π under MDP transition p and reward function r' as $V^\pi(r', p)$.

We assume the learning agent (e.g., a personalized service) interacts with an unknown MDP for K episodes. Each episode $k \in [K]$ is regarded as the interaction with a *unique* user $u_k \in \mathcal{U}$, where $\mathcal{U}$ is the user space. Following [13], a user u_k can be seen as a tree of depth H encoding the state and reward responses they would reply to all A^H possible sequences of actions from the agent. For each episode $k \in [K]$, the learner determines policy π_k and sends it to user u_k for execution. The output of the execution, a trajectory $S_k = \big(x_h^k, a_h^k, r_h^k\big)_{h \in [H]}$, is sent back to the learner for updating the policy. We measure the performance of a learning algorithm by its cumulative regret after K episodes,

$$\text{Regret}(K) := \sum_{k=1}^K \left[V_1^\star(x_1^k) - V_1^{\pi_k}(x_1^k)\right], \tag{1}$$

where x_1^k is the initial state sampled from d_1.

B. Shuffle Differential Privacy for Episodic RL

Consider a general case where $\mathcal{D}$ is the data universe, and we have n *unique* users. We say $D, D' \in \mathcal{D}^n$ are neighboring batched datasets if they only differ in one user's data for some $i \in [n]$. Then, we have the standard definition of differential privacy [6]:

Definition 1 (Differential Privacy (DP)): For $\varepsilon, \beta > 0$, a randomized mechanism $\mathcal{M}$ is (ε, β)-DP if for all neighboring datasets D, D' and any event E in the range of $\mathcal{M}$, we have

$$\mathbb{P}[\mathcal{M}(D) \in E] \leq \exp(\varepsilon) \cdot \mathbb{P}[\mathcal{M}(D') \in E] + \beta.$$

The special case of $(\varepsilon, 0)$-DP is also called *pure DP*, whereas, for $\beta > 0$, (ε, β)-DP is referred to as *approximate DP*.

[2]For simplicity, we assume that the rewards are binary. However, our results naturally extend to $[0,1]$, by replacing our private binary summation mechanism (defined in Section V) with a private summation mechanism for real numbers in $[0,1]$ with similar guarantees.

[3]For a positive integer n, we define $[n] := \{1, \ldots, n\}$.

Shuffle Differential Privacy. A standard shuffle-model protocol $\mathcal{T} = (\mathcal{E}, \mathcal{F}, \mathcal{G})$ consists of three parts: (i) a (local) random encoder $\mathcal{E}$ at each user's side; (ii) a secure shuffler $\mathcal{F}$; and (iii) an analyzer $\mathcal{G}$ at the central server. For n users, each user u_i first locally applies the encoder $\mathcal{E}$ on its raw data D_i and sends the resulting messages $\mathcal{E}(D_i)$ to the shuffler $\mathcal{F}$. The shuffler $\mathcal{F}$ permutes messages from all the batch users uniformly at random and then reports $\mathcal{F}(\mathcal{E}(D_1), \ldots, \mathcal{E}(D_n))$ to the analyzer. The analyzer $\mathcal{G}$ then aggregates the received messages from the shuffler and outputs desired statistics. In this protocol, the users trust the shuffler but not the analyzer. Hence, the goal is to ensure the output of the shuffler $\mathcal{F}$ on two neighboring datasets is indistinguishable in the analyzer's view. Here, we define the mechanism $(\mathcal{F} \circ \mathcal{E}^n)(D) = \mathcal{F}(\mathcal{E}(D_1), \ldots, \mathcal{E}(D_n))$, where $D \in \mathcal{D}^n$. Finally, we have the definition of shuffle DP [18]:

Definition 2 (Shuffle Differential Privacy (SDP)): A protocol $\mathcal{T} = (\mathcal{E}, \mathcal{F}, \mathcal{G})$ for n users is (ε, β)-SDP if the mechanism $\mathcal{F} \circ \mathcal{E}^n$ satisfies (ε, β)-DP.

Shuffle Privacy in RL. To adapt the shuffle model into RL protocol, it is natural to divide total K users into B batches as in [21], with each user's data being their trajectory. Let L_b denote the size of batch b, such that $K = \sum_{b=1}^{B} L_b$. For each batch $b \in [B]$, L_b users execute the same policy and generate a batch dataset containing their trajectories. The shuffle-model protocol is then applied to this dataset to output desired private statistics, which are subsequently sent to the learner for policy updates. Here, instead of a single-batch output, one needs to protect the outputs across all B batches. To this end, we define the (composite) mechanism $\mathcal{M}_{\mathcal{T}} = (\mathcal{F} \circ \mathcal{E}^{L_1}, \ldots, \mathcal{F} \circ \mathcal{E}^{L_B})$, where each mechanism $\mathcal{F} \circ \mathcal{E}^{L_b}$ operates on a trajectories dataset of L_b users. With this notation, we have the following definition [18].

Definition 3 (B-batch SDP): A B-batch shuffle protocol $\mathcal{T}$ is (ε, β)-SDP if the mechanism $\mathcal{M}_{\mathcal{T}}$ satisfies (ε, β)-DP.

In the central DP model [13], the privacy burden lies with a central server, which injects carefully designed noise into the necessary statistics to protect privacy. On the other hand, in the local DP model [15], each user's data is privatized by adding noise to their local data. In contrast, in the shuffle privacy model, privacy without a trusted central server is achieved by ensuring that the inputs to the analyzer $\mathcal{G}$ already satisfy DP. Specifically, by incorporating a secure shuffler $\mathcal{F}$ and properly adjusting the noise level in the encoder $\mathcal{E}$, we ensure that the final added noise in the aggregated data over the batch of users matches the noise that would have otherwise been added by the central server in the central model. Through this, the shuffle model provides the possibility to achieve similar utility to the central model, but maintain privacy without a trusted central server.

III. Algorithm

In this section, we introduce a generic algorithmic framework, *Shuffle Differentially Private Policy Elimination* (SDP-PE, Algorithm 1). Unlike existing private RL algorithms that rely on the *optimism in the face of uncertainty* principle [13], [15], SDP-PE builds on the *policy elimination* idea from [26], which extends action elimination from bandits to RL.

At a high level, SDP-PE divides the exploration process into stages, with exponentially increasing batch sizes. In each stage, it maintains an active policy set ϕ, and refines value estimates and confidence intervals for all active policies using the private model estimate from sufficient shuffle-private statistics. These value estimates are then used to eliminate policies that are likely sub-optimal. At the last stage, the remaining policies are guaranteed to be near-optimal. An important aspect of SDP-PE is the concept of *forgetting*, where only the data of the current stage is used to estimate the model, preventing noise accumulation from earlier stages.

To estimate the value for all active policies with uniform convergence, we address this problem by obtaining an accurate estimate of the transition and reward functions. This requires sufficient exploration for each state-action pair, and injecting appropriate noise into the visitation counts and cumulative rewards of these pairs to preserve privacy. However, some states are rarely visited due to their very low transition probabilities. To remedy this, we construct an *absorbing* MDP that replaces these infrequent-visited states with an absorbing state $x^\dagger$. For the remaining states, we ensure they are visited sufficiently often by some policy in ϕ, allowing the estimated value to uniformly approximate the true value under the original MDP.

Inspired by the `APEVE` algorithm in [26], we design SDP-PE shown in Algorithm 1. It divides the K episodes (users) into a sequence of stages. Each stage b consists of three steps, and the batch length for sub-steps grows exponentially as $L_b := 2^b$ for $b = 1, \ldots, B$, with $B = O(\log K)$.

1) Crude Exploration: For each pair (x, a), explore them layer by layer from scratch using the policy in the current policy set ϕ_b that has the highest visitation probability. We apply shuffle Privatizer to the exploration data of each layer to obtain the private counts. These counts help identify the infrequently-visited tuples $\mathcal{W}$, and construct a private crude estimate $\tilde{P}^{\text{cru},b}$ of the corresponding absorbing MDP P'.
2) Fine Exploration: Using the crude transition estimate from the previous step, we explore each h, x, a with the policy that has the highest visitation probability under the crude transition model. We then apply the shuffle Privatizer to the entire exploration data from this step and construct a refined private estimate $\tilde{P}^{\text{ref},b}$ of P' and reward estimate $\tilde{r}^b$.
3) Policy Elimination: We evaluate all policies in ϕ_b using the refined estimates $\tilde{P}^{\text{ref},b}$ and $\tilde{r}^b$. The active policy set is updated by eliminating all policies whose value upper confidence bound (UCB) is less than the lower confidence bound (LCB) of any other active policy.

As the algorithm proceeds and more data is collected in further stages, our confidence intervals shrink, along with the policy elimination step, ensuring that the optimal policy stays in the active policy set with high probability, which guarantees that the algorithm eventually converges to the optimal policy.

Algorithm 1 Shuffle Private Policy Elimination

Parameters: Episode number K, universal constant C, failure probability δ, privacy budget $\varepsilon > 0$ and a Privatizer.
Initialization: $\phi_1 = \{\text{all the deterministic policies}\}$, $\iota = \log(2HAK/\delta)$. Set precision levels $E_{\varepsilon,\delta}$ for Privatizer.

for $b = 1$ to B **do**
  if $2(\sum_{i=1}^{b} L_i) \geq K$ **then**
    $L_b = \frac{K - 2(\sum_{i=1}^{b} L_i)}{2}$. (o.w. $L_b = 2^b$)
  end if
  $\mathcal{W}^b, \tilde{P}^{\text{cru},b} =$ Crude Exploration(ϕ_b, L_b, ε,Privatizer).
  $\tilde{P}^{\text{ref},b}, \tilde{r}^b =$ Fine Exploration($\mathcal{W}^b, \tilde{P}^{\text{cru},b}, \phi_b, L_b, \varepsilon$, Privatizer).
  $\psi_b = \emptyset$
  for $\pi \in \phi_b$ **do**
    if $\sup_{\pi' \in \phi_b} V^{\pi'}(\tilde{r}^b, \tilde{P}^{\text{ref},b}) - V^{\pi}(\tilde{r}^b, \tilde{P}^{\text{ref},b}) \geq 2C\big(\sqrt{\frac{H^5X^2A\iota}{L_b}} + \frac{X^3A^2H^5E_{\varepsilon,\delta}\iota}{L_b}\big)$ **then**
      Update $\psi_b \leftarrow \psi_b \cup \{\pi\}$.
    end if
  end for
  $\phi_{b+1} \leftarrow \phi_b \backslash \psi_b$.
end for

A. Counts in Algorithm 1

The algorithm introduced in the previous section employs a *model-based* approach for solving the private RL problem and uses the *forgetting* idea to estimate the MDP model. In this approach, only the data from the current step is used to construct the batch dataset. However, the dataset construction differs between the Crude Exploration and Fine Exploration steps, as will be detailed later. Once a batch dataset is formed, the counts for model estimation are established as follows.

Consider the general case where a batch dataset D from n users is sent to the Privatizer. We aim to collect the visitation counts and cumulative rewards for each pair (x, a, x') at h-th step. $N_h(x, a, x') := \sum_{i=1}^{n} \mathbb{1}\{x_h^i, a_h^i, x_{h+1}^i = x, a, x'\}$, similarly $N_h(x, a) := \sum_{x' \in \mathcal{X}} N_h(x, a, x')$, and $R_h(x, a) := \sum_{i=1}^{n} \mathbb{1}\{x_h^i, a_h^i = x, a\} \cdot r_h^i$. The shuffle Privatizer then releases privatized versions of these counts, denoted as $\widetilde{N}_h(x, a, x')$, $\widetilde{N}_h(x, a)$, and $\widetilde{R}_h(x, a)$. Assumption 4 below guarantees that the private counts closely approximate the true counts, as justified in Section V.

Assumption 4 (Private counts): For any privacy budget $\varepsilon > 0$ and failure probability $\delta \in (0, 1)$, the private counts returned by Privatizer satisfy, for some $E_{\varepsilon,\delta} > 0$, with probability at least $1 - 3\delta$, over all (h, x, a, x'), $|\widetilde{N}_h(x, a) - N_h(x, a)| \leq E_{\varepsilon,\delta}$, $|\widetilde{N}_h(x, a, x') - N_h(x, a, x')| \leq E_{\varepsilon,\delta}$, $|\widetilde{R}_h(x, a) - R_h(x, a)| \leq E_{\varepsilon,\delta}$ and $\widetilde{N}_h(x, a) = \sum_{x' \in \mathcal{X}} \widetilde{N}_h(x, a, x') \geq N_h(x, a)$, $\widetilde{N}_h(x, a, x') > 0$.

Based on Assumption 4, we define the private estimation of P and r built using counts from current batch:

$$\widetilde{P}_h(x'|x, a) := \frac{\widetilde{N}_h(x, a, x')}{\widetilde{N}_h(x, a)}, \tilde{r}_h(x, a) := \frac{\widetilde{R}_h(x, a)}{\widetilde{N}_h(x, a)}. \quad (2)$$

By construction, $\widetilde{P}_h(\cdot|x, a)$ is a valid probability distribution.

B. Crude Exploration in Algorithm 2

Algorithm 2 Crude Exploration

Input: Policy set ϕ, number of episodes L, privacy budget ε and a Privatizer.
Initialization: $L_0 = \frac{L}{HXA}, C_1 = 6, \mathcal{W} = \emptyset, \iota = \log(2HAK/\delta)$. $1_{h,x,a}$ is a reward function r' where $r'_{h'}(x', a') = \mathbb{1}\{(h', x', a') = (h, x, a)\}$. $x^\dagger$ is an additional absorbing state. $\tilde{P}^{\text{cru}}$ is a transition function over extended space $\mathcal{X} \cup \{x^\dagger\} \times \mathcal{A}$, initialized arbitrarily.
Output: Infrequent tuples $\mathcal{W}$, and crude estimated transition function $\tilde{P}^{\text{cru}}$

for $h = 1$ to H **do**
  Set data set $D^h = \emptyset$.
  for $(x, a) \in \mathcal{X} \times \mathcal{A}$ **do**
    $\pi_{h,x,a} = \text{argmax}_{\pi \in \phi} V^{\pi}(1_{h,x,a}, \tilde{P}^{\text{cru}})$
    Run $\pi_{h,x,a}$ for L_0 episodes, and add the trajectories into dataset D^h.
  end for
  Send batch dataset D^h to the Privatizer.
  Receive private counts $\widetilde{N}_h(x, a, x')$ and $\widetilde{N}_h(x, a)$ for all (x, a, x') in h-th horizon from Privatizer.
  $\mathcal{W} = \mathcal{W} \cup \{(h, x, a, x') | \widetilde{N}_h(x, a, x') \leq C_1 E_{\varepsilon,\delta} H^2 \iota\}$
  $\tilde{P}^{\text{cru}} =$ Estimate Transition($\widetilde{N}_h, \mathcal{W}, x^\dagger, h, \tilde{P}^{\text{cru}}$).
end for
return $\{\mathcal{W}, \tilde{P}^{\text{cru}}\}$.

To learn an accurate private estimate of $P_h(x'|a, x)$ for any tuple (h, x, a, x'), it is necessary to collect private counts that occur at least $O(E_{\varepsilon,\delta} H^2 \iota)$ times for each tuple. Thus, we try to visit each tuple as frequently as possible by using policies from active policy set ϕ. We define the set of infrequently visited tuples (h, x, a, x') as $\mathcal{W}$, which consists of all the tuples (h, x, a, x') that are visited less than $O(E_{\varepsilon,\delta} H^2 \iota)$ times in current exploration. For the tuples not in $\mathcal{W}$, we can get accurate estimates, and for tuples in $\mathcal{W}$, they have little influence on the value estimate.

In crude exploration step, we perform layer-wise exploration. During the exploration of the h-th layer, we construct $\pi_{h,x,a}$ that has the highest visit probability for (h, x, a) under the private crude estimate $\tilde{P}^{\text{cru},b}$, and then run each $\pi_{h,x,a}$ for the same number of episodes. At the same time, we collect the interaction history for the h-th layer as a batch dataset D^h, and send it to the Privatizer to generate the private counts of this batch. The private counts are then used to update the infrequent tuple set $\mathcal{W}$ and crude estimate of the h-th layer.

Similar to [26], we construct an absorbing MDP transition function P' by letting $P' = P$ first and then move the probability $P'_h(x'|x, a)$ to $P'_h(x^\dagger|x, a)$ for all $(h, x, a, x') \in \mathcal{W}$ to help an accurate estimate for P.

Definition 5 (Abosorbing MDP P'): Given $\mathcal{W}$ and P, $\forall (h, x, a, x') \notin \mathcal{W}$, let $P'_h(x'|x, a) = P_h(x'|x, a)$, $\forall (h, x, a, x') \in \mathcal{W}$, $P'_h(x'|x, a) = 0$. For any $(h, x, a) \in [H] \times \mathcal{X} \times \mathcal{A}$, define $P'_h(x^\dagger|x^\dagger, a) = 1$ and $P'_h(x^\dagger|x, a) = 1 - \sum_{x' \in \mathcal{X}: (h,x,a,x') \notin \mathcal{W}} P'_h(x'|x, a)$.

With the same clipping process, $\tilde{P}^{\text{cru},b}$ is derived from $\tilde{P}^b$ as a private estimate of P' where $\tilde{P}^b$ is computed by equation (2). Besides, with high probability, for all (h,x,a,x'), one of the following conditions holds: $(1-\frac{1}{H})\cdot \tilde{P}_h^{\text{cru},b}(x'|x,a) \le P'_h(x'|x,a) \le (1+\frac{1}{H})\cdot \tilde{P}_h^{\text{cru},b}(x'|x,a)$, $\tilde{P}_h^{\text{cru},b}(x'|x,a) = P'_h(x'|x,a) = 0$. Based on this property, the policies $\pi_{h,x,a}$ are guaranteed to be efficient for exploration, as they maximize the probability of visiting the relevant state-action pairs.

C. Fine Exploration in Algorithm 3

Algorithm 3 Fine Exploration

Input: Infrequent tuples $\mathcal{W}$, crude estimated transition $\tilde{P}^{\text{cru}}$, policy set ϕ, number of episodes L, privacy budget ε and a Privatizer.

Initialization: $L_0 = \frac{L}{HXA}, D = \emptyset$. $1_{h,x,a}$ is a reward function r' where $r'_{h'}(x',a') = \mathbb{1}\{(h',x',a') = (h,x,a)\}$. Initialize refined transition estimate $\tilde{P}^{\text{ref}} = \tilde{P}^{\text{cru}}$.

Output: Refined estimated transition function $\tilde{P}^{\text{ref}}$ and reward function $\tilde{r}$.

1: **for** $(h,x,a) \in [H]\times\mathcal{X}\times\mathcal{A}$ **do**
2: $\quad \pi_{h,x,a} = \text{argmax}_{\pi\in\phi} V^\pi(1_{h,x,a}, \tilde{P}^{\text{cru}})$
3: $\quad$ Run $\pi_{h,x,a}$ for L_0 episodes, and add the trajectories into dataset D.
4: **end for**
5: Send batch dataset D to the Privatizer.
6: Receive private counts $\widetilde{N}_h(x,a,x')$, $\widetilde{N}_h(x,a)$ and $\widetilde{R}_h(x,a,x')$ for all (h,x,a,x') from Privatizer.
7: **for** $h \in [H]$ **do**
8: $\quad \tilde{P}^{\text{ref}} = \text{Estimate Transition}(\widetilde{N}_h, \mathcal{W}, x^\dagger, h, \tilde{P}^{\text{ref}})$.
9: $\quad \tilde{r} = \text{Estimate Rewards}(\widetilde{R}_h, \widetilde{N}_h, h, \tilde{r})$
10: **end for**
11: **return** $\tilde{P}^{\text{ref}}$, $\tilde{r}$.

The idea of fine exploration is to use $\tilde{P}^{\text{cru},b}$ to construct policies that ensure efficient visitation of each tuple. Specially, this is done with the guarantee that $\sup_{\pi\in\phi_b} \frac{V^\pi(1_{h,x,a},P')}{\mu_h(x,a)} \le 12HXA$, where μ is the true distribution of our data. In this way, we can get a refined private estimate $\tilde{P}^{\text{ref},b}$ for P', which allows $V^\pi(r',\tilde{P}^{\text{ref},b})$ to be an accurate estimate of $V^\pi(r',P')$ simultaneously across all policies $\pi\in\phi_b$ and any reward function r'.

During the fine exploration, for each (h,x,a), we find policy $\pi_{h,x,a}$ that visits (h,x,a) with highest probability. After running $\pi_{h,x,a}$ of all (h,x,a) for the same number of episodes, we compile the entire history into a batch dataset D and pass it to the Privatizer. This process along with the same clipping process in crude exploration yields a refined private estimate $\tilde{P}^{\text{ref},b}$ for P' and reward function $\tilde{r}^b$.

IV. Regret Guarantee

The following theorem shows a regret bound of SDP-PE.

Theorem 6 (Regret bound of SDP-PE): For any privacy budget $\varepsilon > 0$ and failure probability $\delta \in (0,1)$, and any Privatizer that satisfies Assumptions 4, with probability at least $1-9\delta$, the regret of SDP-PE (Algorithm 1) is

$$\text{Regret}(K) \le \tilde{\mathcal{O}}\left(\sqrt{H^5X^2AK} + X^3A^2H^5E_{\varepsilon,\delta}\right).$$

The proof parallels the arguments in [26] for the analysis of APEVE. The key difference lies in adjusting the uniform value confidence bound for all active deterministic policies to account for the noise introduced by the private counts.

For each stage b, assume that for any $\pi \in \phi_b$, the value function $V^\pi(r,P)$ can be estimated up to an error ξ_b with high probability. With this estimation, policies that are at least $2\xi_b$ sub-optimal can be eliminated based on their estimated value. Therefore, the optimal policy will never be eliminated, as its value is always within the confidence interval. All remaining policies will be at most $4\xi_b$ sub-optimal. By summing the regret across all stages, we have with high probability that

$$\text{Regret}(K) \le 2HL_1 + \sum_{b=2}^{B} 2L_b\cdot 4\xi_{b-1}. \tag{3}$$

The following lemma gives a bound of ξ_b using our private estimate $\tilde{P}^{\text{ref},b}$ and $\tilde{r}^b$ of the absorbing MDP.

Lemma 7: With probability $1-7\delta$, it holds that for any stage b and $\pi\in\phi_b$,

$$|V^\pi(r,P) - V^\pi(\tilde{r}^b,\tilde{P}^{\text{ref},b})| \le \tilde{\mathcal{O}}\left(\sqrt{\frac{X^2AH^5}{L_b}} + \frac{X^3A^2H^5E_{\varepsilon,\delta}}{L_b}\right).$$

As shown in [27], if each (h,x,a) tuple is visited frequently enough, the empirical transition is sufficient for a uniform approximation to $V^\pi(r,P)$. In our setting, we leverage the absorbing MDP P' as the key intermediate step to ensure this guarantee. Thus, we can decompose this confidence bound into three components, "Model Bias" $|V^\pi(r,P)-V^\pi(r,P')|$, "Reward Error" $|V^\pi(r,P')-V^\pi(\tilde{r}^b,P')|$ and "Model Variance" $|V^\pi(\tilde{r}^b,P')-V^\pi(\tilde{r}^b,\tilde{P}^{\text{ref},b})|$. In this sketch, we focus on the "Model Bias" and "Model Variance" terms, leaving a detailed analysis of the lower-order term "Reward Error" and the complete proof for the full paper.

A. "Model Bias": Difference between P and P'

To analyze the difference between the true MDP with P and the absorbing MDP with P', we first clarify the properties of the crude transition estimate $\tilde{P}^{\text{cru},b}$.

Property of $\tilde{P}^{\text{cru},b}$. In b-th stage, if the private visitation count $\widetilde{N}_h(x,a,x')$ for a tuple (h,x,a,x') exceeds $O(E_{\varepsilon,\delta}H^2\iota)$, the following holds with high probability, $(1-\frac{1}{H})\cdot\tilde{P}_h^{\text{cru},b}(x'|x,a) \le P'_h(x'|x,a) \le (1+\frac{1}{H})\cdot\tilde{P}_h^{\text{cru},b}(x'|x,a)$, which can be proven using Bernstein's inequality. By the construction of $\mathcal{W}$, and $P', \tilde{P}^{\text{cru},b}$, the above equation holds for any (h,x,a,x'). Consequently, for any $(h,x,a)\in[H]\times\mathcal{X}\times\mathcal{A}, \pi\in\phi_b$, we have,

$$\frac{1}{4}V^\pi(1_{h,x,a},\tilde{P}^{\text{cru},b}) \le V^\pi(1_{h,x,a},P') \le 3V^\pi(1_{h,x,a},\tilde{P}^{\text{cru},b}).$$

Because $\pi_{h,x,a} = \text{argmax}_{\pi\in\phi_b} V^\pi(1_{h,x,a},\tilde{P}^{\text{cru},b})$,

$$V^{\pi_{h,x,a}}(1_{h,x,a},P') \ge \frac{1}{12}\sup_{\pi\in\phi_b} V^\pi(1_{h,x,a},P'),$$

which shows that $\pi_{h,x,a}$ efficiently covers the tuple (h,x,a).

Uniform bound on $|V^\pi(r,P)-V^\pi(r,P')|$. Next, we aim to bound $\sup_{\pi\in\phi_b}\sup_{r'}|V^\pi(r',P)-V^\pi(r',P')|$. This leads to bounding $\sup_{\pi\in\phi_b}\mathbb{P}_\pi[\mathcal{B}]$, where the bad event $\mathcal{B}$ occurs when the trajectory visits some infrequently visited tuples in $\mathcal{W}$. By the definition of $\mathcal{W}$, we can show that these tuples are difficult to visit for any policy in ϕ_b, i.e., with high probability, $\sup_{\pi\in\phi_b}\mathbb{P}_\pi[\mathcal{B}] \le \tilde{\mathcal{O}}\left(\frac{E_{\varepsilon,\delta}X^3A^2H^4}{L_b}\right)$. Based on this observation, we have the model bias bounded.

Lemma 8: With high probability, for any policy $\pi\in\phi_b$ and reward function r', it holds that

$$0 \le V^\pi(r',P) - V^\pi(r',P') \le \tilde{\mathcal{O}}\left(\frac{X^3A^2H^5E_{\varepsilon,\delta}}{L_b}\right).$$

B. "Model Variance": Difference between P' and $\tilde{P}^{ref,b}$

The idea here is to separate the impact of privacy noise and empirical uncertainty on the value estimation. Using the simulation lemma from [28], we can bound the variance by decomposing it into two components: "Empirical Variance" $\sum_{h,x,a}|(P'_h-\bar{P}_h^{\text{ref},b})\tilde{V}^\pi_{h+1}(x,a)|\cdot V^\pi(1_{h,x,a},P')$, and "Privacy Variance" $\sum_{h,x,a}|(\bar{P}_h^{\text{ref},b}-\tilde{P}_h^{\text{ref},b})\tilde{V}^\pi_{h+1}(x,a)|\cdot V^\pi(1_{h,x,a},P')$, where $\bar{P}_h^{\text{ref},b}$ represents the non-private estimate version of $\tilde{P}_h^{\text{ref},b}$.

Since our transition estimate is accurate, the problem reduces to bounding $\sum_{h,x,a}\frac{V^\pi(1_{h,x,a},P')}{N_h(x,a)}$. With the desirable properties of the exploration policy $\pi_{h,x,a}$ and the layer-wise exploration strategy, we could have the "Model Variance" bounded as follows.

Lemma 9: With high probability, for any policy $\pi\in\phi_b$ and reward function r', it holds that

$$|V^\pi(r',P')-V^\pi(r',\tilde{P}^{\text{ref},b})| \le \tilde{\mathcal{O}}\left(\sqrt{\frac{X^2AH^5}{L_b}}+\frac{X^3A^2H^3E_{\varepsilon,\delta}}{L_b}\right).$$

Using a similar principle, we can bound the "Reward Error" by decomposing it into empirical and private terms with the help of the simulation lemma. As a result, the "Reward Error" will appear to be a lower-order term.

C. Putting Together

Combining the bounds of all three terms, we arrive at the bound in Theorem 6 by substituting $\xi_b = \tilde{\mathcal{O}}(\sqrt{\frac{X^2AH^5}{L_b}} + \frac{X^3A^2H^5E_{\varepsilon,\delta}}{L_b})$ in equation (3).

V. Privacy Guarantee

In this section, we introduce a shuffle Privatizer based on the shuffle binary summation mechanism introduced by [18], and show that Algorithm 1 satisfies (ε,β)-SDP.

A. Achieving Shuffle DP

To protect the information of the batch users (D^h in Crude Exploration and D in Fine Exploration), we privatize the visitation counts and rewards using the shuffling binary summation mechanism. Following Section III-A, with n users in the current batch, we outline the process for generating the privatized visitation counts $\widetilde{N}_h(x,a,x')$ for a specific (h,x,a,x'). The procedure for other counts like $\widetilde{N}_h(x,a)$ and the cumulative rewards $\widetilde{R}_h(x,a)$ is analogous.

Firstly, we allocate privacy budget $\varepsilon' = \frac{\varepsilon}{3H}$ and privacy confidence $\beta>0$ to this counter, and define a parameter $\tau = \mathcal{O}(H^2\log(1/\beta)/\varepsilon^2)$ that controls noise addition. On the local side, each user encodes their data using a local randomizer $\mathcal{E}$. If $n\le\tau$, user i encodes local count as $z_i = \mathbb{1}_h(x,a,x') + \sum_{j=1}^m y_j$, where $\{y_j\}_{j=1}^m$ are i.i.d. sampled, $y_j\sim$ Bernoulli$(1/2)$, and $m=\lceil\frac{\tau}{n}\rceil$. Otherwise, the output is $z_i = \mathbb{1}_h(x,a,x')+y$ where $y\sim$ Bernoulli$(\frac{\tau}{2n})$. Once encoded, users send their local messages z_i to a secure shuffler $\mathcal{F}$, which permutes their messages randomly and then forwards them to the analyzer $\mathcal{G}$.

On the analyzer side, it will recover a noisy estimate, denoted as $\ddot{N}_h(x,a,x')$. If $n\le\tau$, $\ddot{N}_h(x,a,x') = \sum_{i=1}^n z_i - \lceil\frac{\tau}{n}\rceil\cdot\frac{n}{2}$, otherwise, $\ddot{N}_h(x,a,x') = \sum_{i=1}^n z_i - \frac{\tau}{2}$. The noisy counts $\ddot{N}_h(x,a,x')$ are post-processed to ensure they meet the probability distribution constraints and Assumption 4. The final private counts $\widetilde{N}_h(x,a,x')$ and $\widetilde{N}_h(x,a)$ are derived from this post-processing procedure. The private cumulative rewards $\widetilde{R}_h(x,a)$ are directly obtained from the analyzer's output.

Post-processing steps for $\ddot{N}_h(x,a,x')$. Given the noisy counts $\ddot{N}_h(x,a)$, $\ddot{N}_h(x,a,x')$ for all (x,a,x') from the analyzer, we will adopt the techniques from [14], [17] to satisfy Assumption 4.

Firstly, we solve the optimization problem efficiently for all (x,a) below.

$$\begin{aligned}\min t \;\; \text{s.t. } & n(x')\ge 0, |n(x')-\ddot{N}_h(x,a,x')|\le t, \forall x', \\ & |\sum\nolimits_{x'\in\mathcal{X}} n(x') - \ddot{N}_h(x,a)| \le \frac{E_{\varepsilon,\delta}}{4}.\end{aligned} \tag{4}$$

Let $\bar{N}_h(x,a,x')$ denote a minimizer of this problem, we define $\bar{N}_h(x,a) = \sum_{x'\in\mathcal{X}}\bar{N}_h(x,a,x')$. By adding some term, as done below, we make sure that the private counts $\widetilde{N}_h(x,a)$ never underestimate the respective true counts:

$$\begin{aligned}\widetilde{N}_h(x,a) &= \bar{N}_h(x,a) + \frac{E_{\varepsilon,\delta}}{2}, \\ \widetilde{N}_h(x,a,x') &= \bar{N}_h(x,a,x') + \frac{E_{\varepsilon,\delta}}{2X}.\end{aligned} \tag{5}$$

If $\ddot{N}_h(x,a)$ satisfies $|\ddot{N}_h(x,a,x')-N_h(x,a,x')|\le\frac{E_{\varepsilon,\delta}}{4}$, $|\ddot{N}_h(x,a)-N_h(x,a)|\le\frac{E_{\varepsilon,\delta}}{4}$, for all (h,x,a,x'), with probability $1-2\delta$. Then, $\widetilde{N}_h(x,a)$ derived from Eq. (4) and Eq. (5) satisfy Assumption 4.

Using the privacy composition theorem [6] and utility lemma from [18], we summarize the properties of our shuffle Privatizer in the following lemma:

Lemma 10 (Shuffle DP Privatizer): For any $\varepsilon\in(0,1)$, $\beta\in(0,1)$, the Shuffling Privatizer satisfies (ε,β)-SDP and Assumption 4 with $E_{\varepsilon,\delta} = \tilde{\mathcal{O}}\left(\frac{H}{\varepsilon}\right)$.

As corollaries of Theorem 6, we obtain the regret and privacy guarantees for SDP-PE using the shuffle Privatizer.

Theorem 11 (Regret Bound under SDP): For any $\varepsilon\in(0,1), \beta\in(0,1)$, using the Shuffling Privatizer, SDP-PE satisfies (ε,β)-SDP. Furthermore, we obtain Regret$(K)\le\tilde{\mathcal{O}}\left(\sqrt{H^5X^2AK}+\frac{X^3A^2H^6}{\varepsilon}\right)$ with high probability.

B. Discussion

Regret-privacy trade-off. SDP-PE achieves an improved regret-privacy trade-off under shuffle DP, with a regret bound that is optimal regarding the episode number K and privacy budget ε. It outperforms the optimal LDP result [17] by making the dependency on $1/\varepsilon$ additive rather than multiplicative, and matches the dependence on K and ε in the JDP case [17]. This also addresses the concerns outlined in [15], where a burn-in phase was required for their algorithm.

Dependence on H, X, A in the regret. Although our regret bound is optimal with respect to K, a gap remains in dependence on X and H compared to the lower bound $\Omega(\sqrt{H^3XAK})$ in the non-private setting. It is still open whether this can be improved under SDP constraints.

Computational efficiency. Our algorithm's efficiency is limited by the need to evaluate all policies in the active set, which may be exponential in size. Computational efficiency could potentially be improved by using an external optimizer, as in [25], to optimize over the full policy space without explicitly maintaining the active set.

VI. Conclusion and Future Work

In this paper, we presented the first generic algorithm for shuffle private RL, *Shuffle Differentially Private Policy Elimination*, which achieves a refined privacy-regret trade-off under shuffle differential privacy constraints. Unlike prior approaches, our approach relies on novel strategies, i.e., batch data collection and private updating, and the policy elimination principle. By instantiating the proposed Privatizer, SDP-PE attains optimal regret over episode number K and privacy budget ε. Notably, the regret bound under SDP matches that under JDP while providing a stronger privacy guarantee, and improves the results under LDP which expands possibilities for private RL.

Despite these, our algorithm is sub-optimal in its dependence on MDP parameters, X, A, H, which requires some refined ideas for dealing with the trade-off between batch-based exploration and exploitation under privacy. Meanwhile, improving the computation efficiency is also one interesting problem. Our generic algorithm can also be extended to other advanced privacy notions, e.g., Renyi DP [29].

References

[1] O. Gottesman, F. Johansson, M. Komorowski, A. Faisal, D. Sontag, F. Doshi-Velez, and L. A. Celi, "Guidelines for reinforcement learning in healthcare," *Nature medicine*, vol. 25, no. 1, pp. 16–18, 2019.

[2] M. M. Afsar, T. Crump, and B. Far, "Reinforcement learning based recommender systems: A survey," *ACM Computing Surveys*, vol. 55, no. 7, pp. 1–38, 2022.

[3] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Gray *et al.*, "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems*, 2022.

[4] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 267–284.

[5] Y. Lei, D. Ye, S. Shen, Y. Sui, T. Zhu, and W. Zhou, "New challenges in reinforcement learning: a survey of security and privacy," *Artificial Intelligence Review*, vol. 56, no. 7, pp. 7195–7236, 2023.

[6] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[7] A. Tossou and C. Dimitrakakis, "Algorithms for differentially private multi-armed bandits," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[8] K. Yazdani, A. Jones, K. Leahy, and M. Hale, "Differentially private lq control," *IEEE Transactions on Automatic Control*, vol. 68, no. 2, pp. 1061–1068, 2022.

[9] Y. Wang, Z. Huang, S. Mitra, and G. E. Dullerud, "Differential privacy in linear distributed control systems: Entropy minimizing mechanisms and performance tradeoffs," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 118–130, 2017.

[10] A. Benvenuti, C. Hawkins, B. Fallin, B. Chen, B. Bialy, M. Dennis, and M. Hale, "Differentially private reward functions for markov decision processes," in *2024 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2024, pp. 631–636.

[11] P. Gohari, M. Hale, and U. Topcu, "Privacy-preserving policy synthesis in markov decision processes," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 6266–6271.

[12] A. Rajabi, B. Ramasubramanian, A. Al Maruf, and R. Poovendran, "Privacy-preserving reinforcement learning beyond expectation," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022.

[13] G. Vietri, B. Balle, A. Krishnamurthy, and S. Wu, "Private reinforcement learning with pac and regret guarantees," in *International Conference on Machine Learning*. PMLR, 2020, pp. 9754–9764.

[14] S. Bai, L. Zeng, C. Zhao, X. Duan, M. S. Talebi, P. Cheng, and J. Chen, "Differentially private no-regret exploration in adversarial markov decision processes," in *The 40th Conference on Uncertainty in Artificial Intelligence*, 2024.

[15] E. Garcelon, V. Perchet, C. Pike-Burke, and M. Pirotta, "Local differential privacy for regret minimization in reinforcement learning," *Advances in Neural Information Processing Systems*, 2021.

[16] S. R. Chowdhury and X. Zhou, "Differentially private regret minimization in episodic markov decision processes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[17] D. Qiao and Y.-X. Wang, "Near-optimal differentially private reinforcement learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2023, pp. 9914–9940.

[18] A. Cheu, A. Smith, J. R. Ullman, D. Zeber, and M. Zhilyaev, "Distributed differential privacy via shuffling," *Advances in Cryptology-EUROCRYPT 2019*, vol. 1, 2019.

[19] A. Bittau, Ú. Erlingsson, P. Maniatis, I. Mironov, A. Raghunathan, D. Lie, M. Rudominer, U. Kode, J. Tinnes, and B. Seefeld, "Prochlo: Strong privacy for analytics in the crowd," in *Proceedings of the 26th symposium on operating systems principles*, 2017, pp. 441–459.

[20] A. Lowy, A. Ghafelebashi, and M. Razaviyayn, "Private non-convex federated learning without a trusted server," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2023.

[21] J. Tenenbaum, H. Kaplan, Y. Mansour, and U. Stemmer, "Differentially private multi-armed bandits in the shuffle model," *Advances in Neural Information Processing Systems*, 2021.

[22] S. R. Chowdhury and X. Zhou, "Shuffle private linear contextual bandits," in *International Conference on Machine Learning*, 2022.

[23] Y. Bai, T. Xie, N. Jiang, and Y.-X. Wang, "Provably efficient q-learning with low switching cost," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[24] E. Johnson, C. Pike-Burke, and P. Rebeschini, "Sample-efficiency in multi-batch reinforcement learning: The need for dimension-dependent adaptivity," in *The Twelfth International Conference on Learning Representations*, 2024.

[25] Z. Zhang, Y. Jiang, Y. Zhou, and X. Ji, "Near-optimal regret bounds for multi-batch reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 586–24 596, 2022.

[26] D. Qiao, M. Yin, M. Min, and Y.-X. Wang, "Sample-efficient reinforcement learning with loglog (t) switching cost," in *International Conference on Machine Learning*. PMLR, 2022, pp. 18 031–18 061.

[27] C. Jin, A. Krishnamurthy, M. Simchowitz, and T. Yu, "Reward-free exploration for reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4870–4879.

[28] C. Dann, T. Lattimore, and E. Brunskill, "Unifying pac and regret: Uniform pac bounds for episodic reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[29] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE, 2017, pp. 263–275.