Combinatorial Pure Exploration with Full-Bandit or Partial Linear Feedback

Yihan Du,[1*] **Yuko Kuroki,**[2*] **Wei Chen**[3]
[1]IIIS, Tsinghua University, [2]The University of Tokyo, RIKEN, [3]Microsoft Research
duyh18@mails.tsinghua.edu.cn, ykuroki@ms.k.u-tokyo.ac.jp, weic@microsoft.com

Abstract

In this paper, we first study the problem of combinatorial pure exploration with full-bandit feedback (CPE-BL), where a learner is given a combinatorial action space $\mathcal{X} \subseteq \{0,1\}^d$, and in each round the learner pulls an action $x \in \mathcal{X}$ and receives a random reward with expectation $x^\top \theta$, with $\theta \in \mathbb{R}^d$ a latent and unknown environment vector. The objective is to identify the optimal action with the highest expected reward, using as few samples as possible. For CPE-BL, we design the first *polynomial-time adaptive* algorithm, whose sample complexity matches the lower bound (within a logarithmic factor) for a family of instances and has a light dependence of $\Delta_{\min}$ (the smallest gap between the optimal action and sub-optimal actions). Furthermore, we propose a novel generalization of CPE-BL with flexible feedback structures, called combinatorial pure exploration with partial linear feedback (CPE-PL), which encompasses several families of sub-problems including full-bandit feedback, semi-bandit feedback, partial feedback and nonlinear reward functions. In CPE-PL, each pull of action x reports a random feedback vector with expectation of $M_x \theta$, where $M_x \in \mathbb{R}^{m_x \times d}$ is a transformation matrix for x, and gains a random (possibly nonlinear) reward related to x. For CPE-PL, we develop the first *polynomial-time* algorithm, which simultaneously addresses limited feedback, general reward function and combinatorial action space (e.g., matroids, matchings and s-t paths), and provide its sample complexity analysis. Our empirical evaluation demonstrates that our algorithms run orders of magnitude faster than the existing ones, and our CPE-BL algorithm is robust across different $\Delta_{\min}$ settings while our CPE-PL algorithm is the first one returning correct answers for nonlinear reward functions.

1 Introduction

The problem of *best arm identification* (BAI) is the pure-exploration framework in stochastic multi-armed bandits. In BAI, at each step a learner chooses an arm and observes its reward sampled from an unknown distribution, with the goal of returning the best arm with the highest expected reward using as few exploration steps as possible. This problem abstracts a decision making model in the face of uncertainty with a wide range of applications, and has received much attentions in the literature (Even-Dar, Mannor, and Mansour 2006; Audibert, Bubeck, and Munos 2010; Chen and Li 2015; Kaufmann, Cappé, and Garivier 2016).

In many application domains, possible actions have a certain combinatorial structure. For example, each action may be a size-k subset of keywords in online advertisements (Rusmevichientong and Williamson 2006), or an assignment between workers and tasks in crowdsourcing (Lin et al. 2014), or a spanning tree in communication networks (Huang, Liu, and Ding 2008). To deal with such a combinaotrial action space, the model of *combinatorial pure exploration of multi-armed bandits* (CPE-MB) was first proposed by Chen et al. (2014). In this model, there are d base arms, each of which is associated with an unknown reward distribution, and a collection of *super arms*, each of which is a subset of base arms. A learner plays a base arm at each step and observes its random reward, with the goal of identifying the best super arm that maximizes the sum of expected rewards at the end of exploration. CPE-MB generalizes the classical BAI problem (Kalyanakrishnan and Stone 2010; Kalyanakrishnan et al. 2012; Bubeck, Wang, and Viswanathan 2013).

However, many real-world scenarios may not fit into CPE-MB. In particular, CPE-MB assumes that the learner can directly play each base arm and observe its outcome, but this might not be allowed due to system constraints or privacy issues. Only a few studies avoid such an assumption. Kuroki et al. (2020b) studied the *combinatorial pure exploration with full-bandit linear feedback (CPE-BL)*, in which the learner pulls a super arm (rather than base arm) and only observes the sum of rewards from the involved base arms. They designed an efficient algorithm for CPE-BL, but the algorithm is nonadaptive and its sample complexity heavily depends on the smallest gap between the best and the other super arms (denoted by $\Delta_{\min}$). Rejwan and Mansour (2020) also designed an efficient algorithm with an adaptive Successive-Accept-Reject algorithm, but the algorithm only works for the top-k case of CPE-BL, which we show can be simply reduced to previous CPE-MB (see Appendix D in the full version).

Note that CPE-BL can be regarded as an instance of the *best arm identification in linear bandits* (BAI-LB), which has received increasing attention recently (Soare, Lazaric, and Munos 2014; Tao, Blanco, and Zhou 2018; Fiez et al. 2019). However, none of the existing algorithms for BAI-LB can efficiently solve CPE-BL, because their running times have

[*]The first two authors have equal contributions.

polynomial dependence on the size of action space, which is exponential in the combinatorial setting.

In this paper, we provide the first algorithm solving CPE-BL that simultaneous achieves the following properties: (a) polynomial-time complexity, (b) adaptive sampling, such that the sample complexity is not heavily dependent on $\Delta_{\min}$; (c) general combinatorial constraints, and (d) nearly optimal sample complexity for some family of instances.

Next, we propose a more general setting, *combinatorial pure exploration with partial linear feedback (CPE-PL)*, which simultaneously models limited feedback, general (possibly nonlinear) reward and combinatorial action space. In CPE-PL, given a combinatorial action space $\mathcal{X} \subseteq \{0,1\}^d$, where each dimension corresponds to a base arm and each action $x \in \mathcal{X}$ can also be viewed as a super arm that contains those dimensions with coordinate 1. At each step the learner chooses an action (super arm) $x_t \in \mathcal{X}$ to play and observes a random partial linear feedback with expectation of $M_{x_t}\theta$, where M_{x_t} is a transformation matrix for x_t and $\theta \in \mathbb{R}^d$ is an unknown environment vector. The learner also gains a random (possibly nonlinear) reward related to x_t and θ, which may not be a part of the feedback and thus may not be directly observed. Given a confidence level δ, the objective is to identify the optimal action with the maximum expected reward with probability at least $1-\delta$, using as few samples as possible. CPE-PL framework includes CPE-BL as its important sub-problem. In CPE-BL, the learner observes full-bandit feedback (i.e. $M_x = x^\top$) and gains linear reward (with expectation of $x^\top\theta$) after each play.

The model of CPE-PL appears in many practical scenarios. For example, in online ranking (Chaudhuri and Tewari 2017), a company recommends their products to users by presenting rankings of entire items, and wants to find the best ranking with limited feedback on the top-ranked item due to user burden constraints and privacy concerns. In crowdsourcing (Lin et al. 2014), an employer assigns crowdworkers to tasks according to the worker-task performance, and wants to find the best matching with limited feedback on a small subset of the completed tasks, owing to the burden of entire feedback and privacy issues (see Section 4.3 for detailed applications).

We remark that, CPE-PL is a novel and general model that encompasses several families of sub-problems across full-bandit feedback, semi-bandit feedback and nonlinear reward function, and it cannot be translated to CPE-BL or BAI-LB. For example, when the reward function is $(x^\top\theta)/\|x\|_1$ and $M_x = \mathrm{diag}(x)$, CPE-PL reduces to a semi-bandit problem with nonlinear reward function, and no existing CPE-BL or BAI-LB algorithm could solve this problem.

Finally, we empirically compare our algorithms with several state-of-the-art CPE-BL and BAI-LB algorithms. Our result demonstrates that (a) our algorithms run much faster than all others, some of which cannot even finish after days of running; (b) For CPE-BL, our adaptive algorithm is much more robust on different $\Delta_{\min}$ settings than the existing non-adaptive algorithm; and (c) For CPE-PL, our algorithm is the only one that correctly outputs the optimal action for a nonlinear reward function among all the compared algorithms.

To summarize, our contributions include: (a) proposing the first *polynomial-time adaptive* algorithm for CPE-BL with general constraints that achieves near optimal sample complexity for some family of instances; and (b) proposing the general CPE-PL framework and the first *polynomial time* algorithm for CPE-PL with its sample complexity analysis.

Due to the space constraint, full proofs with additional results and discussions are moved to the appendices in the full version (Du, Kuroki, and Chen 2020).

1.1 State-of-the-art Related Work

Here we compare with the most related and state-of-the-art works (see Table 1), and the full discussion and comparison table with notation definitions are included in Appendix A in the full version. For CPE-BL, Kuroki et al. (2020b) propose a polynomial-time but static algorithm ICB, which has a heavy dependence on $\Delta_{\min}$ in the sample complexity and requires a large number of samples for small-$\Delta_{\min}$ instances empirically (see Appendix I in the full version). Rejwan and Mansour (2020) develop a polynomial-time adaptive algorithm CSAR but it only works for the top-k case, which has a naive reduction to previous CPE-MB (see Appendix D in the full version).

For BAI-LB where the action space is often considered small, Tao, Blanco, and Zhou (2018) propose an adaptive algorithm ALBA with a light $\Delta_{\min}$ dependence. Fiez et al. (2019) present the first lower bound and a nearly optimal algorithm RAGE. Recently, Katz-Samuels et al. (2020) also design an improved nearly optimal algorithm Peace, which is built upon the previous RAGE. Degenne et al. (2020) and Jedra and Proutiere (2020) develop asymptotically optimal algorithms, but a fair way to compare their results with other non-asymptotical results is unknown. While the existing BAI-LB algorithms achieve satisfactory sample complexity, none of them can efficiently solve CPE-BL with an exponentially large combinatorial action space. This paper proposes the first polynomial-time adaptive algorithm for CPE-BL, which is nearly optimal for some family of instances, and the first polynomial-time algorithm for CPE-PL.

2 Problem Statements

Combinatorial pure exploration with full-bandit linear feedback (CPE-BL). In CPE-BL, a learner is given d base arms numbered $1, 2, \ldots, d$. We define $\mathcal{X} \subseteq \{0,1\}^d$ as a collection of subsets of base arms, which satisfies a certain combinatorial structure such as size-k subsets, matroids, paths and matchings. A subset of base arms $x \in \mathcal{X}$ is called a super arm (or an action). Let m denote the maximum number of base arms that a super arm in $\mathcal{X}$ contains, i.e. $m = \max_{x\in\mathcal{X}} \|x\|_1$ $(m \le d)$. There is an unknown environment vector $\theta \in \mathbb{R}^d$ with $\|\theta\|_2 \le L$. At each time step t, a learner pulls a super arm x_t and receives a random reward (full-bandit feedback) $y_t = x^\top(\theta + \eta_t)$, where η_t is a zero-mean noise vector bounded in $[-1,1]^d$ and it is independent among different time step t. Let $x^* = \mathrm{argmax}_{x\in\mathcal{X}}\, x^\top\theta$ denote the optimal super arm, and we assume that the optimal x^* is unique as previous pure exploration works (Chen et al. 2014; Lin et al. 2014; Fiez et al. 2019) do. Let Δ_i denote the gap of the expected rewards between x^* and the super arm with the i-th largest expected reward.

Table 1: Comparison between our results and state-of-the-art results for CPE-BL(PL). “General” represents that the algorithm works for any combinatorial structure. $\tilde{O}(\cdot)$ only omits $\log\log$ factors. Main notations is defined in Section 2.

Algorithm	Sample complexity	Case	Problem Type	Strategy	Time
GCB-PE (ours, Thm. 2)	$O\big(\frac{\lvert\sigma\rvert\beta_\sigma^2 L_p^2}{\Delta_{\min}^2}\log\frac{\beta_\sigma^2 L_p^2}{\Delta_{\min}^2\delta}\big)$	General	CPE-PL	Static	Poly(d)
PolyALBA (ours, Thm. 1)	$\tilde{O}\big(\sum_{i=2}^{\lfloor\frac{d}{2}\rfloor}\frac{1}{\Delta_i^2}\log\frac{\lvert\mathcal{X}\rvert}{\delta}+\frac{d^2 m\xi_{\max}(\widetilde{M}(\lambda)^{-1})}{\Delta_{d+1}^2}\log\frac{\lvert\mathcal{X}\rvert}{\delta}\big)$	General	CPE-BL	Adaptive	Poly(d)
ICB (Kuroki et al. 2020b)	$\tilde{O}\big(\frac{d\xi_{\max}(M(\lambda)^{-1})\rho(\lambda)}{\Delta_{\min}^2}\log\frac{d\xi_{\max}(M(\lambda)^{-1})\rho(\lambda)}{\Delta_{\min}^2\delta}\big)$	General	CPE-BL	Static	Poly(d)
CSAR (Rejwan and Mansour 2020)	$\tilde{O}\big(\sum_{i=2}^{d}\frac{1}{\tilde{\Delta}_i^2}\log\frac{d}{\delta}\big)$	Top-k	CPE-BL	Adaptive	Poly(d)
ALBA (Tao, Blanco, and Zhou 2018)	$\tilde{O}\big(\sum_{i=2}^{d}\frac{1}{\Delta_i^2}(\log\delta^{-1}+\log\lvert\mathcal{X}\rvert)\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\lvert\mathcal{X}\rvert)$
RAGE (Fiez et al. 2019)	$O\big(\sum_{t=1}^{\lfloor\log_2(4/\Delta_{\min})\rfloor}2(2^t)^2\tilde{\rho}(\mathcal{Y}(S_t))\log(t^2\lvert\mathcal{X}\rvert^2/\delta)\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\lvert\mathcal{X}\rvert)$
LinGame(-C) (Degenne et al. 2020)	$\limsup_{\delta\to 0}\frac{\mathbb{E}_\theta[\tau_\delta]}{\log(1/\delta)}\le\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{2\lvert\lvert x^*-x\rvert\rvert_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\lvert\mathcal{X}\rvert)$
Peace (Katz-Samuels et al. 2020)	$O\big((\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{\lvert\lvert x^*-x\rvert\rvert_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}+\gamma^*)\log(1/\delta)\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\lvert\mathcal{X}\rvert)$
LT&S (Jedra and Proutiere 2020)	$\limsup_{\delta\to 0}\frac{\mathbb{E}_\theta[\tau_\delta]}{\log(1/\delta)}\le\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{\lvert\lvert x^*-x\rvert\rvert_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\lvert\mathcal{X}\rvert)$
Lower Bound (Fiez et al. 2019)	$\mathbb{E}_\theta[\tau_\delta]\ge\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{\lvert\lvert x^*-x\rvert\rvert_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}\log(1/2.4\delta)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	-	-

Given a confidence level $\delta\in(0,1)$, the objective is to use as few samples as possible to identify the optimal super arm with probability at least $1-\delta$. This is often called the fixed confidence setting in the bandit literature, and the number of samples required by the learner is called *sample complexity*.

Combinatorial pure exploration with partial-monitoring linear feedback (CPE-PL). CPE-PL is a generalization of CPE-BL to partial linear feedback and nonlinear reward functions. In CPE-PL, each super arm $x\in\mathcal{X}$ is associated with a transformation matrix $M_x\in\mathbb{R}^{m_x\times d}$, whose row dimension m_x depends on x. At each timestep t, a learner pulls a super arm x_t and observes a random linear feedback vector $y_t = M_{x_t}(\theta+\eta_t)\in\mathbb{R}^{m_{x_t}}$, where η_t is the noise vector. Meanwhile, the learner gains a random reward with expectation of $\bar{r}(x_t,\theta)$. Note that for each pull of super arm x_t, the actual expected reward $\bar{r}(x_t,\theta)$ may not be part of the linear feedback vector y_t and thus may not be directly observed by the learner. Similarly, given a confidence $\delta\in(0,1)$, the learner aims to use as few samples as possible to identify the optimal super arm with probability at least $1-\delta$.

CPE-PL allows more flexible feedback structures than CPE-BL or BAI-LB, and encompasses several families of sub-problems including full-bandit feedback, semi-bandit feedback and nonlinear reward functions. For example, when $M_x = x^\top\in\mathbb{R}^{1\times d}$ for all $x\in\mathcal{X}$, this model reduces to CPE-BL. When $M_x=\mathrm{diag}(x)\in\mathbb{R}^{d\times d}$ for all $x\in\mathcal{X}$, this model reduces to combinatorial pure exploration with semi-bandit feedback (see Appendix B in the full version for illustration examples).

The regret minimization version of CPE-PL has been studied in Lin et al. (2014); Chaudhuri and Tewari (2016). In this paper, we study the pure exploration version and inherit the two technical assumptions from Lin et al. (2014); Chaudhuri and Tewari (2016) in order to design an efficient algorithm.

Assumption 1 (Lipschitz continuity of the expected reward function)**.** *There exists a constant L_p such that for any $x\in\mathcal{X}$ and any $\theta_1,\theta_2\in\mathbb{R}^d$, $\lvert\bar{r}(x,\theta_1)-\bar{r}(x,\theta_2)\rvert\le L_p\lvert\lvert\theta_1-\theta_2\rvert\rvert_2$.*

Assumption 2 (Global observer set)**.** *There exists a global observer set $\sigma=\{x_1,x_2,\dots,x_{\lvert\sigma\rvert}\}\subseteq\mathcal{X}$, such that the stacked $\sum_{i=1}^{\lvert\sigma\rvert}m_{x_i}\times d$ transformation matrix $M_\sigma=(M_{x_1};M_{x_2};\dots;M_{x_{\lvert\sigma\rvert}})$ is of full column rank ($rank(M_\sigma)=d$).*

Then, the Moore-Penrose pseudoinverse M_σ^+ satisfies $M_\sigma^+M_\sigma=I_d$, where I_d is the $d\times d$ identity matrix. We justify Assumption 2 by the fact that without the existence of global observer set, the learner cannot recover θ and may not distinguish two different actions. With Assumption 2, we can systematically construct a global observer set with $\lvert\sigma\rvert\le d$ by sequentially adding an action that strictly increases the rank of M_σ, until M_σ reaches the full rank. Section 4.3 provides more detailed discussion on the global observer set with applications of CPE-PL.

Notations. For clarity, we also introduce the following notations. Let $[d]=\{1,2,\dots,d\}$. For a vector $x\in\mathbb{R}^d$ and a matrix $B\in\mathbb{R}^{d\times d}$, let $\|x\|_B=\sqrt{x^\top Bx}$. For a positive definite matrix $B\in\mathbb{R}^{d\times d}$, we use $B^{1/2}$ to denote the unique positive definite matrix whose square is B. For a given family $\mathcal{X}$, we use $\triangle(\mathcal{X})$ to denote the set of probability distributions over $\mathcal{X}$. For distribution $\lambda\in\triangle(\mathcal{X})$, we define $\mathrm{supp}(\lambda)=\{x:\lambda(x)>0\}$, $M(\lambda)=\mathbb{E}_{z\sim\lambda}[zz^\top]$ and $\widetilde{M}(\lambda)=\sum_{x\in\mathrm{supp}(\lambda)}xx^\top$. We denote the maximum (minimal) eigenvalue of matrix B by $\xi_{\max}(B)$ ($\xi_{\min}(B)$).

3 Combinatorial Pure Exploration with Full-bandit Feedback (CPE-BL)

In this section, we propose the first polynomial-time adaptive algorithm PolyALBA for CPE-BL, and show that its sample complexity matches the lower bound (within a logarithmic factor) for a family of instances.

3.1 Algorithm Procedure

ALBA algorithm. Before stating the main algorithm, we introduce the Adaptive Linear Best Arm (ALBA) algorithm for BAI-LB (Tao, Blanco, and Zhou 2018) (see Algorithm 1 for

Algorithm 1: $\mathsf{ALBA}(S, \delta)$ (Tao, Blanco, and Zhou 2018)

```
Input  : Action set $S$ and confidence $\delta$.
1 Initialize $S_1 \leftarrow S$;
2 for $q \leftarrow 1, \ldots, \lfloor \log_2 d \rfloor$ do
3     $\delta_q \leftarrow \frac{6}{\pi^2} \frac{\delta}{(q+1)^2}$;
4     $S_{q+1} \leftarrow \mathrm{ElimTil}_{\lfloor \frac{d}{2^q} \rfloor}(S_q, \delta_q)$;
5     $q \leftarrow q + 1$;
Output : $x \in S_{q+1}$
```

Algorithm 2: $\mathsf{ElimTil}_p(S, \delta)$

```
Input  : A parameter $p$, arms set $S$ and confidence level $\delta$.
1 Compute $\lambda^*_S \leftarrow \min_{\lambda \in \triangle(S)} \max_{x \in S} x^\top M(\lambda)^{-1} x$;
2 Initialize $S_1 \leftarrow S$, $r \leftarrow 1$;
3 while $|S_r| > p$ do
4     Set $\varepsilon_r \leftarrow 1/2^r$, $\delta_r \leftarrow 6/\pi^2 \cdot \delta/r^2$;
5     $\hat{\theta}_r \leftarrow \mathsf{VectorEst}(\lambda^*_S, c_0 \frac{2+(6+\varepsilon_r/2)d}{(\varepsilon/2)^2} \ln \frac{5|S|}{\delta_r})$;
6     $x_r \leftarrow \mathrm{argmax}_{x \in S_r} x^\top \hat{\theta}_r$;
7     $S_{r+1} \leftarrow S_r \setminus \{x \in S_r \mid x^\top \hat{\theta}_r < x_r^\top \hat{\theta}_r - \varepsilon_r\}$;
8     $r \leftarrow r + 1$;
Output : $S_r$
```

its description), which is the key subroutine of our proposed method $\mathsf{PolyALBA}$. First, we describe the randomized least-square estimator defined by Tao et al. (2018). Let $y_1, \ldots, y_n$ be n *i.i.d.* samples following a given distribution $\lambda \in \triangle(\mathcal{X})$, and let the corresponding rewards be $r_1, \ldots, r_n$ respectively. Let $b = \sum_{i=1}^n r_i y_i$. Then, the randomized estimator $\hat{\theta}$ is given by $\hat{\theta} = A^{-1}b$, where $A = nM(\lambda) \in \mathbb{R}^{d \times d}$ (recall that $M(\lambda) = \mathbb{E}_{z \sim \lambda}[zz^\top]$). The procedure for computing the estimate for θ is described in $\mathsf{VectorEst}$ (Algorithm 3). ALBA is an elimination-based algorithm, where in round q it identifies the top $d/2^q$ arms and discards the remaining arms by means of $\mathsf{ElimTil}_p$ (Algorithm 2). Note that $\mathsf{ALBA}(S, \delta)$ runs in time polynomial to $|S|$. However, since in CPE-BL, $|\mathcal{X}|$ is exponential to the instance size, it is infeasible to run ALBA with $S = \mathcal{X}$. Our main contribution is the nontrivial construction of a polynomial sized S_1 to run ALBA with.

Main algorithm. Now we present our proposed algorithm $\mathsf{PolyALBA}$ (see Algorithm 4 for its description), in which ALBA is invoked with $S = S_1$ with $|S_1| = d$. Set S_1 is constructed by a novel preparation procedure in the first epoch ($q = 0$). In this preparation epoch, we first compute a fixed distribution $\lambda \in \triangle(\mathcal{X})$ that has a polynomial-size support and a key parameter α (line 1). Then, based on λ we apply static estimation to estimate θ, until we see a big enough gap between the empirically best and $(d+1)$-th best actions (lines 4–13). The empirical top-d actions, excluding those that also have big gaps to the best one, form the set S_1 (lines 10–11), which is used to call ALBA to obtain the final result $\hat{x}^*$.

Note that, computing the empirical best $d+1$ super arms

Algorithm 3: $\mathsf{VectorEst}(\lambda, n)$

```
Input  : distribution $\lambda$ and the number of samples $n$
1 Let $y_1 \ldots, y_n$ be the $n$ samples acquired from supp$(\lambda)$ according to the distribution $\lambda$;
2 Pull arms $y_1, \ldots, y_n$;
3 Observe the rewards $r_1, \ldots, r_n$;
4 $A \leftarrow n \cdot \sum_{x \in \mathrm{supp}(\lambda)} \lambda(x) x x^\top$;
5 $b \leftarrow \sum_{i=1}^n r_i y_i$;
Output : The estimate $\hat{\theta} \leftarrow A^{-1} b$
```

Algorithm 4: $\mathsf{PolyALBA}$

```
Input  : confidence level $\delta$, $c_0 = \max\{4L^2, 3\}$.
1 Set $q \leftarrow 0$ and $\delta_q \leftarrow \frac{6}{\pi^2} \frac{\delta}{(q+1)^2}$;
2 Compute a distribution $\lambda \leftarrow \lambda^*_{\mathcal{X}_\sigma}$ and parameter
  $\alpha \leftarrow \sqrt{md/\xi_{\min}(\widetilde{M}(\lambda^*_{\mathcal{X}_\sigma}))}$ by Algorithm 5;
3 $r \leftarrow 1$;
4 while true do
5     Set $\varepsilon_r \leftarrow \frac{1}{2^r}$ and $\delta_r \leftarrow \frac{6}{\pi^2} \frac{\delta_q}{r^2}$;
6     $\ell(\varepsilon) \leftarrow \frac{2m + 2\alpha\sqrt{m}d + 4\alpha^2 d + \alpha\varepsilon d}{\varepsilon^2}$;
7     $\hat{\theta}_r \leftarrow \mathsf{VectorEst}(\lambda, c_0 \ell(\frac{\varepsilon_r}{2}) \ln(\frac{5|\mathcal{X}|}{\delta_r}))$;
8     Select $d+1$ actions $\hat{x}_1, \ldots, \hat{x}_d, \hat{x}_{d+1}$ with the highest $d+1$ empirical means $x^\top \hat{\theta}_r$ in all $x \in \mathcal{X}$;
9     if $\hat{x}_1^\top \hat{\theta}_r - \hat{x}_{d+1}^\top \hat{\theta}_r > \varepsilon_r$ then
10        $B_1 \leftarrow \{\hat{x}_1, \ldots, \hat{x}_d\}$;
11        $S_1 \leftarrow B_1 \setminus \{x \in B_1 \mid \hat{x}_1^\top \hat{\theta}_r - x^\top \hat{\theta}_r > \varepsilon_r\}$;
12        break;
13    $r \leftarrow r + 1$;
14 $\hat{x}^* \leftarrow$ output by $\mathsf{ALBA}(S_1, \delta_1)$
Output : $\hat{x}^*$
```

Algorithm 5: Computing a distribution λ

```
Input  : $d$-base arms
1 Choose any $d$ super arms $\mathcal{X}_\sigma = \{b_1, \ldots, b_d\}$ from $\mathcal{X}$, such that rank$(X) = d$ where $X = (b_1, \ldots, b_d)$;
2 $\lambda^*_{\mathcal{X}_\sigma} \leftarrow \mathrm{argmin}_{\lambda \in \triangle(\mathcal{X}_\sigma)} \max_{x \in \mathcal{X}_\sigma} x^\top M(\lambda)^{-1} x$ by the entropy mirror descent algorithm of (Tao, Blanco, and Zhou 2018) (see Algorithm 8 in Appendix E in the full version) ;
3 $\alpha \leftarrow \sqrt{md/\xi_{\min}(\widetilde{M}(\lambda^*_{\mathcal{X}_\sigma}))}$;
Output : $\lambda^*_{\mathcal{X}_\sigma}$ and $\alpha$
```

can be done in polynomial time by using *Lawler's k-best procedure* (Lawler 1972). This procedure only requires the existence of the efficient maximization oracle, which is satisfied in many combinatorial problems such as maximum matching, shortest paths and minimum spanning tree. Moreover, the computational efficiency of $\mathsf{PolyALBA}$ is not merely owing to the Lawler's k-best procedure. In fact, even if previous BAI-LB algorithms apply the same procedure, they cannot run in polynomial time since they explicitly maintain exponential-sized action set and sample on distributions with exponential supports. These render heavy computation and memory in every round of previous algorithms. In contrast, we avoid the naive enumeration and sampling on the combinatorial space directly, and instead find empirical top-d actions as representatives through a novel polynomial-time computation procedure.

3.2 Theoretical Analysis

Now we provide the sample complexity bound of PolyALBA.

Theorem 1. *With probability at least $1-\delta$, the PolyALBA algorithm (Algorithm 4) returns the best super arm x^* with sample complexity*

$$O\Bigg(\sum_{i=2}^{\lfloor \frac{d}{2} \rfloor} \frac{c_0}{\Delta_i^2} (\ln \delta^{-1} + \ln |\mathcal{X}| + \ln\ln \Delta_i^{-1}) + \frac{c_0 d(\alpha\sqrt{m}+\alpha^2)}{\Delta_{d+1}^2} \left(\ln \delta^{-1} + \ln |\mathcal{X}| + \ln\ln \Delta_{d+1}^{-1}\right) \Bigg),$$

where $\alpha = \sqrt{md/\xi_{\min}(\widetilde{M}(\lambda^*_{\mathcal{X}_\sigma}))}$.

Analysis of the statistical and computational efficiency. The first term in Theorem 1 is for the remaining epochs required by subroutine ALBA and the second term is for the preparation procedure. As shown in Theorem 1, our sample complexity bound has lighter dependence on $1/\Delta^2_{\min}$, compared with the existing result (see Table 1). Now we explain the key role for the polynomial-time complexity of PolyALBA in the first epoch played by the distribution $\lambda^*_{\mathcal{X}_\sigma}$ and parameter α. Notice that even if we employ a uniform distribution on a polynomial-size support $\mathcal{X}_\sigma \subseteq \mathcal{X}$, i.e., $\lambda_{\mathcal{X}_\sigma} = (1/|\mathcal{X}_\sigma|)_{x\in\mathcal{X}_\sigma}$, computing the maximal confidence bound $\max_{x\in\mathcal{X}} \|x\|_{M(\lambda_{\mathcal{X}_\sigma})^{-1}}$ is NP-hard, while many (UCB-based) algorithms in LB ignore this issue and simply use a brute force method. In contrast, PolyALBA utilizes G-optimal design (Pukelsheim 2006) and runs in polynomial time while guaranteeing the optimality. In the following lemma, we show that $\alpha\sqrt{d}$ gives the upper bound on the maximal ellipsoidal norm associated to $M(\lambda^*_{\mathcal{X}_\sigma})^{-1}$.

Lemma 1. *For $\lambda^*_{\mathcal{X}_\sigma}$ and α obtained by Algorithm 5, it holds that $\max_{x\in\mathcal{X}} \|x\|_{M(\lambda^*_{\mathcal{X}_\sigma})^{-1}} \leq \alpha\sqrt{d}$, where $\alpha = \sqrt{md/\xi_{\min}(\widetilde{M}(\lambda^*_{\mathcal{X}_\sigma}))}$.*

From the equivalence theorem for optimal experimental designs (Proposition 2 in Appendix G in the full version), it holds that $\min_{\lambda\in\triangle(\mathcal{X})} \max_{x\in\mathcal{X}} \|x\|_{M(\lambda)^{-1}} = \sqrt{d}$. From this fact and Lemma 1, we see that $\lambda^*_{\mathcal{X}_\sigma}$ is α (≥ 1)-approximate solution to $\min_{\lambda\in\triangle(\mathcal{X})} \max_{x\in\mathcal{X}} \|x\|_{M(\lambda)^{-1}}$ where $\mathcal{X}$ can be defined by general combinatorial constraints. Note that α can be easily obtained by computing $\xi_{\min}(\widetilde{M}(\lambda^*_{\mathcal{X}_\sigma}))$ (recall that $\widetilde{M}(\lambda) = \sum_{x\in\mathrm{supp}(\lambda)} xx^\top$). Therefore, by employing $\lambda^*_{\mathcal{X}_\sigma}$ and a prior knowledge of its approximation ratio α, we can guarantee that the preparation sampling scheme identifies a set S_1 containing the optimal super arm x^* with high probability. In the remaining epochs, PolyALBA can successfully focus on sampling near-optimal super arms by ALBA owing to the optimality of S_1. Note that $\alpha = 1$ if we compute $\min_{\lambda\in\triangle(\mathcal{X})} \max_{x\in\mathcal{X}} \|x\|_{M(\lambda)^{-1}}$ exactly. If we approximately solve it, α is independent on the arm-selection ratio but it can depend on the support of λ. For further discussion on improving α, please see Appendix E in the full version.

Discussion on the optimality. Fiez et al. (2019) give a sample complexity lower bound for BAI-LB (see Table 1) and propose a nearly (within a logarithmic factor) optimal algorithm RAGE with sample complexity of $O\left(\sum_{t=1}^{\lfloor \log_2(4/\Delta_{\min})\rfloor} 2(2^t)^2 \tilde{\rho}(\mathcal{Y}(S_t)) \log(t^2|\mathcal{X}|^2/\delta)\right)$. Note that the existing lower bound (Fiez et al. 2019) and nearly (or asymptotically) optimal algorithms (Fiez et al. 2019; Degenne et al. 2020; Katz-Samuels et al. 2020) do not consider computational efficiency for combinatorially-large $|\mathcal{X}|$, and the lower bound for polynomial-time CPE-BL algorithms is still an open problem.

When compared to the lower bound (Fiez et al. 2019), there exists a family of instances such that $\Delta_{\lfloor d/2^{(t-2)}\rfloor+1} = 4\cdot 2^{-t}$, $t = 2,3,\dots,\log_2(\frac{4}{\Delta_{\min}})$, in which our PolyALBA achieves $O(\sum_{t=2}^{\lfloor \log_2(4/\Delta_{\min})\rfloor} 2(2^t)^2\tilde{\rho}(\mathcal{Y}(S_t))\log(t^2|\mathcal{X}|^2/\delta) + md\xi_{\max}(\tilde{M}^{-1}(\lambda))\tilde{\rho}(\mathcal{Y}(S_1))\log(|\mathcal{X}|^2/\delta))$ sample complexity (see Appendix C in the full version for more details). When ignoring a logarithmic factor and with sufficiently small $\Delta_{\min}$, the additional term related to $\xi_{\max}(\tilde{M}^{-1}(\lambda))$ is absorbed and the result matches the lower bound, which shows superiority over other heavily $\Delta_{\min}$-dependent algorithms (Soare, Lazaric, and Munos 2014; Karnin 2016; Kuroki et al. 2020b). Note that the term related to $\xi_{\max}(\tilde{M}^{-1}(\lambda))$ can be viewed as the cost for achieving computational efficiency.

To our best knowledge, our PolyALBA is the first polynomial-time adaptive algorithm that works for CPE-BL with general combinatorial structures and achieves nearly optimal sample complexity for a family of problem instances.

4 Combinatorial Pure Exploration with Partial Linear Feedback (CPE-PL)

In this section, we present the first polynomial-time algorithm GCB-PE for CPE-PL with sample complexity analysis, and discuss its further improvements via a non-uniform allocation strategy. We also give practical applications for CPE-PL and explain the corresponding global observer set and sample complexity result in these scenarios.

4.1 Algorithm Procedure

We illustrate GCB-PE in Algorithm 6. GCB-PE estimates the environment vector θ by repeatedly pulling the global observer set $\sigma = \{x_1, x_2, \dots, x_{|\sigma|}\}$, which in turn helps estimate the expected rewards $\bar{r}(x,\theta)$ of all super arms $x \in \mathcal{X}$ using the Lipschitz continuity (Assumption 1). We call one pull of global observer set σ *one exploration round*, the specific procedure of which is described as follows: for the n-th exploration round, the learner plays all actions in $\sigma = \{x_1, x_2, \dots, x_{|\sigma|}\}$ once and respectively observes feedback $y_1, y_2, \dots, y_{|\sigma|}$, the stacked vector of which is denoted by $\vec{y}_n = (y_1; y_2; \dots; y_{|\sigma|})$. The estimate of environment vector θ in this exploration round is $\hat{\theta}_n = M_\sigma^+ \vec{y}_n$, where M_σ^+ is the Moore-Penrose pseudoinverse of M_σ. From Assumption 2, we have $\mathbb{E}[\hat{\theta}_n] = \theta$. Then, we can use the independent estimates in multiple rounds, i.e., $\hat{\theta}(n) = \frac{1}{n}\sum_{j=1}^{n} \hat{\theta}_j$, to obtain an accurate estimate of θ.

Similar to Lin et al. (2014), we define a constant $\beta_\sigma := \max_{\eta_1, \cdots, \eta_{|\sigma|} \in [-1,1]^d} \|(M_\sigma^\top M_\sigma)^{-1} \sum_{i=1}^{|\sigma|} M_{x_i}^\top M_{x_i} \eta_i\|_2$, which only depends on global observer set σ, and bounds the estimate error of one exploration round, i.e., for any n, $\|\hat{\theta}_n - \theta\|_2 \le \beta_\sigma$, the proof of which is given in Appendix H.1 in the full version. Based on β_σ, we further design a global confidence radius $\text{rad}_n = \sqrt{2\beta_\sigma^2 \log(4n^2e^2/\delta)/n}$ for the estimate $\hat{\theta}(n)$, and show that with high probability, rad_n bounds the estimate error of $\hat{\theta}(n)$.

Compared with GCB in Lin et al. (2014), which works for the regret minimization metric of the combinatorial partial monitoring game with linear feedback problem, GCB-PE targets the best action identification and mainly controls the stopping time of the exploration phase rather than balancing the frequency of exploration and exploitation phases. For the pure exploration metric, our global confidence radius rad_n is novelly designed to bound the estimate error. In addition, the stopping condition, which uses the designed confidence radius and Lipschitz continuity of the expected reward function, is also novelly adopted to fit the CPE-PL setting.

The computational efficiency of GCB-PE relies on the polynomial-time offline maximization oracle for the specific combinatorial instance, which is used in the two argmax operations in GCB-PE. It is reasonable to assume the existence of polynomial-time offline maximization oracle, otherwise we cannot efficiently address the exponentially large action space even if the real environment vector θ is known.

4.2 Theoretical Analysis

We give the sample complexity of GCB-PE below.

Theorem 2. *With probability at least $1 - \delta$, the GCB-PE algorithm (Algorithm 6) will return the optimal super arm x^* with sample complexity*

$$O\left(\frac{|\sigma|\beta_\sigma^2 L_p^2}{\Delta_{\min}^2} \log\left(\frac{\beta_\sigma^2 L_p^2}{\Delta_{\min}^2 \delta}\right)\right),$$

where $|\sigma| \le d$.

When the expected reward function is linear, i.e. $\bar{r}(x, \theta) = x^\top \theta$, we have $L_p = \sqrt{m}$, where m ($\le d$) is the maximum number of base arms a super arm contains. In addition, $\beta_\sigma = \text{Poly}(d)$ in several practical applications of CPE-PL (see Section 4.3 for our detailed discussion).

Discussion on the optimality. While the sample complexity of GCB-PE is sometimes worse than the CPE-BL or BAI-LB algorithms (PolyALBA, ALBA and RAGE), it solves a more general class of problems than CPE-BL and

Algorithm 6: GCB-PE

```
Input  : Confidence level $\delta$, global observer set $\sigma$,
         constant $\beta_\sigma$, Lipschitz constant $L_p$
1  for $s = 1, \dots, |\sigma|$ do
2      Pull $x_s$ in observer set $\sigma$, and observe $y_s$;
3  $n \leftarrow 1$;
4  $\vec{y}_1 \leftarrow (y_1; y_2; \dots; y_{|\sigma|})$;
5  $\hat{\theta}_1 \leftarrow M_\sigma^+ \vec{y}_1$ and $\hat{\theta}(1) \leftarrow \hat{\theta}_1$;
6  while true do
7      $\hat{x} \leftarrow \text{argmax}_{x \in \mathcal{X}} \bar{r}(x, \hat{\theta}(n))$;
8      $\hat{x}^- \leftarrow \text{argmax}_{x \in \mathcal{X} \setminus \{\hat{x}\}} \bar{r}(x, \hat{\theta}(n))$;
9      $\text{rad}_n \leftarrow \sqrt{\frac{2\beta_\sigma^2 \log(\frac{4n^2e^2}{\delta})}{n}}$;
10     if $\bar{r}(\hat{x}, \hat{\theta}(n)) - \bar{r}(\hat{x}^-, \hat{\theta}(n)) > 2L_p \cdot \text{rad}_n$ then
11         return $\hat{x}$;
12     else
13         for $s = 1, \dots, |\sigma|$ do
14             Pull $x_s$ in observer set $\sigma$, and observe $y_s$;
15         $n \leftarrow n + 1$;
16         $\vec{y}_n \leftarrow (y_1; y_2; \dots; y_{|\sigma|})$;
17         $\hat{\theta}_n \leftarrow M_\sigma^+ \vec{y}_n$;
18         $\hat{\theta}(n) \leftarrow \frac{1}{n}\sum_{j=1}^{n} \hat{\theta}_j$;
Output : $\hat{x}$
```

BAI-LB. We emphasize that our contribution mainly focuses on proposing the first polynomial-time algorithm GCB-PE that simultaneously addresses combinatorial action apace, partial linear feedback and nonlinear reward function. **On non-uniform or adaptive allocation strategy.** GCB-PE can be further improved by employing a *non-uniform* allocation strategy when considering the global observer set σ with multiplicity: we can obtain such an allocation by solving an optimization $\text{argmin}_{\lambda \in \triangle(\sigma)} \beta_\sigma(\lambda)$ and rounding the result, where $\beta_\sigma(\lambda) := \max_{\eta_1, \cdots, \eta_{|\sigma|} \in [-1,1]^d} \|(M_\sigma^\top M_\sigma)^{-1} \sum_{i=1}^{|\sigma|} \lambda_i M_{x_i}^\top M_{x_i} \eta_i\|_2$. Since uniform sampling is not essential in our analysis, the proposed improvement for GCB-PE via non-uniform allocation does not violate Assumption 2 and keeps our theoretical analysis. GCB-PE is a static algorithm, and we leave the study of adaptive strategies for CPE-PL as future work. In Appendix D in the full version, we discuss a fully-adaptive algorithm for CPE-BL (special case of CPE-PL), and show that the result depends on a non-controllable term $M(\lambda)^{-1}$, which indicates that the static control may be required to deal with linear feedback efficiently.

4.3 Applications for GCB-PL

CPE-PL characterizes more flexible feedback structures than CPE-BL (or BAI-LB) and finds many real-world applications. Below we present two practical applications and discuss the global observer set (Assumption 2) and parameter β_σ.

Online ranking. Consider that a company wishes to recommend their products to users by presenting the ranked list of items. Due to user burden constraints and privacy concerns, collecting a large amount of data on the relevance of all items might be infeasible, and thus the company usually collects the relevance of only the top-ranked item (Chaudhuri and Tewari 2015, 2016, 2017). In this scenario, a learner selects a permutation of d items (each action x is a permutation) at each step, and observes the relevance of the top-ranked item, i.e., M_x contains a single row with 1 in the place of the top-ranked item and 0 everywhere else. The objective is to identify the best permutation as soon as possible. Then, we can construct a global observable set σ to be the set of any d actions which places a distinct item at top. Here M_σ is the $d \times d$ identity matrix and $\beta_\sigma = \sqrt{d}$.

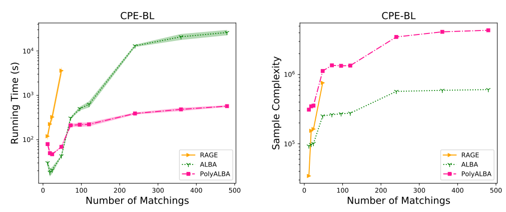
Task assignments in crowdsourcing. Consider that an employer wishes to assign crowdworkers to tasks with high quality performance, and it wants to avoid the high cost and the privacy concern of collecting each individual worker-task pair performance (Lin et al. 2014). Thus, the employer sequentially chooses an assignment from N workers to M tasks (each action x is a worker-task matching) and only collects the sum of performance feedback for $1 \leq s < N$ matched worker-task pairs, i.e., M_x contains a single row with 1s in the places of s matched pairs and 0 everywhere else. The objective is to find the best worker-task matching as soon as possible. For $1 \leq s < N$, Lin et al. (2014) provide a systematic method to construct a global observer set.

5 Experiments

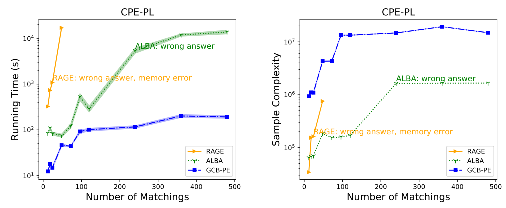
We conduct experiments for CPE-BL and CPE-PL on the matching and top-k instances, and compare our algorithms with the state-of-the-arts in both running time and sample complexity. Due to the space limit, here we only present the results on matchings and defer the top-k results with discussion on $\Delta_{\min}$-dependence to Appendix I in the full version.

We evaluate all the compared algorithms on Intel Xeon E5-2640 v3 CPU at 2.60GHz with 132GB RAM. For both CPE-BL and CPE-PL, we set action space $\mathcal{X}$ as matchings in 3-by-3, 4-by-4 and 5-by-5 complete bipartite graphs. The dimension d, i.e. the number of edges, is set from 9 to 25. The number of matchings $|\mathcal{X}|$ are set from 12 to 480. $\theta_1, \dots, \theta_d$ is set as a geometric sequence in $[0, 1]$. We simulate the random feedback for action x by a Gaussian distribution with mean of $x^\top \theta$ and unit variance. For CPE-PL, we use the full-bandit feedback as CPE-BL ($M_x = x^\top$) but a nonlinear reward function $\bar{r}(x, \theta) = x^\top \theta / \|x\|_1$. For each algorithm, we perform 20 independent runs and present the average running time and sample complexity with 95% confidence intervals across runs. In the experiments, RAGE (Fiez et al. 2019) reports memory errors when $|\mathcal{X}| > 48$ due to its heavy memory burden, and thus we only obtain its results on small-$|\mathcal{X}|$ instances. For PolyALBA, ALBA and RAGE, we obtain the same sample complexity in different runs, since these algorithms compute the required samples at the beginning of each phase and then perform the fixed samples.

Experiments for CPE-BL. For CPE-BL, we compare our PolyALBA with the state-of-the-art BAI-LB algorithms



(a) CPE-BL



(b) CPE-PL

Figure 1: Experimental results of running time and sample complexity for CPE-BL and CPE-PL.

ALBA and RAGE in running time and sample complexity. As shown in Figure 1(a) with a logarithmic y-axis, our PolyALBA runs about two orders of magnitude faster than ALBA and RAGE, and the running time of PolyALBA increases more slowly than the others as $|\mathcal{X}|$ increases. Due to the extra preparation epoch, PolyALBA has a higher sample complexity, but we argue that in practice one has to keep the computation time low first to make an algorithm useful, and for that matter ALBA and RAGE are too slow to run and PolyALBA is the only feasible option.

Experiments for CPE-PL. For CPE-PL, we compare GCB-PE with BAI-LB algorithms ALBA and RAGE in running time and sample complexity on a more challenging nonlinear reward task. In the experiments for CPE-PL, ALBA and RAGE return wrong answers because they are not designed to handle nonlinear reward functions. Nevertheless, we can still analyze the running times presented in Figure 1(b). It shows that our GCB-PE runs two orders of magnitude faster than ALBA and RAGE while reporting the correct answer. In addition, as $|\mathcal{X}|$ increases, the running time of GCB-PE increases in a much slower pace than the others. The experimental results demonstrate the capability of GCB-PE to simultaneously deal with combinatorial action space, nonlinear reward function and partial feedback in a computationally efficient way.

6 Future Work

There are several interesting directions worth further investigation. First, it is open to prove a lower bound of polynomial-time algorithms for both CPE-PL and CPE-BL. Another challenging direction is to design efficient algorithms for specific combinatorial cases to choose the global observer set σ and the distribution $\lambda^*_{\mathcal{X}_\sigma}$, and derive specific sample complexity bounds. Furthermore, the extension of CPE-PL to nonlinear feedback is also a practical and valuable problem.

Acknowledgement

YK would like to thank Masashi Sugiyama and Junya Honda for helpful comments on the manuscript, and Kento Nozawa for his support to use the server. YK is supported by Microsoft Research Asia, KAKENHI 18J23034, and JST ACT-X 1124477.

References

Abbasi-Yadkori, Y.; Pál, D.; and Szepesvári, C. 2011. Improved Algorithms for Linear Stochastic Bandits. In *Proc. NIPS'14*, 2312–2320.

Audibert, J.-Y.; Bubeck, S.; and Munos, R. 2010. Best Arm Identification in Multi-Armed Bandits. In *Proc. COLT'10*, 41–53.

Bubeck, S.; Wang, T.; and Viswanathan, N. 2013. Multiple identifications in multi-armed bandits. In *Proc. ICML'13*, 258–265.

Chaudhuri, S.; and Tewari, A. 2015. Online ranking with top-1 feedback. In *Proc. AISTATS'15*, 129–137.

Chaudhuri, S.; and Tewari, A. 2016. Phased Exploration with Greedy Exploitation in Stochastic Combinatorial Partial Monitoring Games. In *Proc. NIPS'16*, 2433–2441.

Chaudhuri, S.; and Tewari, A. 2017. Online learning to rank with top-k feedback. *The Journal of Machine Learning Research* 18(1): 3599–3648.

Chen, L.; and Li, J. 2015. On the Optimal Sample Complexity for Best Arm Identification. *arXiv preprint* arXiv:1511.03774.

Chen, S.; Lin, T.; King, I.; Lyu, M. R.; and Chen, W. 2014. Combinatorial Pure Exploration of Multi-Armed Bandits. In *Proc. NIPS'14*, 379–387.

Chen, W.; Du, Y.; Huang, L.; and Zhao, H. 2020. Combinatorial Pure Exploration for Dueling Bandit. In *Proc. ICML'20*, 1531–1541.

Dani, V.; Hayes, T. P.; and Kakade, S. M. 2008. Stochastic Linear Optimization Under Bandit Feedback. In *Proc. COLT'08*, 355–366.

Degenne, R.; Menard, P.; Shang, X.; and Valko, M. 2020. Gamification of Pure Exploration for Linear Bandits. In *Proc. ICML'20*, 2432–2442.

Du, Y.; Kuroki, Y.; and Chen, W. 2020. Combinatorial Pure Exploration with Full-Bandit or Partial Linear Feedback. In *arXiv preprint arXiv:2006.07905*. URL https://arxiv.org/pdf/2006.07905.pdf.

Even-Dar, E.; Mannor, S.; and Mansour, Y. 2006. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research* 7: 1079–1105.

Fiez, T.; Jain, L.; Jamieson, K. G.; and Ratliff, L. 2019. Sequential Experimental Design for Transductive Linear Bandits. In *Proc. NeurIPS' 19*, 10667–10677.

Gabillon, V.; Ghavamzadeh, M.; and Lazaric, A. 2012. Best Arm Identification: A Unified Approach to Fixed Budget and Fixed Confidence. In *Proc. NIPS'12*, 3212–3220.

Gabillon, V.; Ghavamzadeh, M.; Lazaric, A.; and Bubeck, S. 2011. Multi-Bandit Best Arm Identification. In *Proc. NIPS'11*, 2222–2230.

Grötschel, M.; Lovász, L.; and Schrijver, A. 1981. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* 1: 169–197.

Huang, S.; Liu, X.; and Ding, Z. 2008. Opportunistic Spectrum Access in Cognitive Radio Networks. In *Proc. INFOCOM'08*, 1427–1435.

Huang, W.; Ok, J.; Li, L.; and Chen, W. 2018. Combinatorial Pure Exploration with Continuous and Separable Reward Functions and Its Applications. In *Proc. IJCAI '18*, 2291–2297.

Jedra, Y.; and Proutiere, A. 2020. Optimal Best-arm Identification in Linear Bandits. *to appear in NeurIPS'20* .

Kalyanakrishnan, S.; and Stone, P. 2010. Efficient Selection of Multiple Bandit Arms: Theory and Practice. In *Proc. ICML'10*, 511–518.

Kalyanakrishnan, S.; Tewari, A.; Auer, P.; and Stone, P. 2012. PAC Subset Selection in Stochastic Multi-armed Bandits. In *Proc. ICML'12*, 655–662.

Karnin, Z. S. 2016. Verification Based Solution for Structured MAB Problems. In *Proc. NIPS' 16*, 145–153.

Katz-Samuels, J.; Jain, L.; Karnin, Z.; and Jamieson, K. 2020. An Empirical Process Approach to the Union Bound: Practical Algorithms for Combinatorial and Linear Bandits. *arXiv preprint arXiv:2006.11685* .

Kaufmann, E.; Cappé, O.; and Garivier, A. 2016. On the complexity of best-arm identification in multi-armed bandit models. *Journal of Machine Learning Research* 17: 1–42.

Kawase, Y.; and Sumita, H. 2019. Randomized strategies for robust combinatorial optimization. In *Proc. AAAI '19*, 7876–7883.

Kiefer, J.; and Wolfowitz, J. 1960. The Equivalence of Two Extremum Problems. *Canadian Journal of Mathematics* 12: 363–366.

Kuroki, Y.; Miyauchi, A.; Honda, J.; and Sugiyama, M. 2020a. Online Dense Subgraph Discovery via Blurred-Graph Feedback. In *Proc. ICML'20*, 5522–5532.

Kuroki, Y.; Xu, L.; Miyauchi, A.; Honda, J.; and Sugiyama, M. 2020b. Polynomial-Time Algorithms for Multiple-Arm Identification with Full-Bandit Feedback. *Neural Computation* 32(9): 1733–1773.

Lawler, E. L. 1972. A Procedure for Computing the K Best Solutions to Discrete Optimization Problems and Its Application to the Shortest Path Problem. *Management Science* 18(7): 401—405.

Lin, T.; Abrahao, B.; Kleinberg, R.; Lui, J.; and Chen, W. 2014. Combinatorial partial monitoring game with linear feedback and its applications. In *Proc. ICML'14*, 901–909.

Pukelsheim, F. 2006. *Optimal Design of Experiments*. Society for Industrial and Applied Mathematics.

Rejwan, I.; and Mansour, Y. 2020. Top-k Combinatorial Bandits with Full-Bandit Feedback. In *Proc. ALT' 20*, 752–776.

Rusmevichientong, P.; and Williamson, D. P. 2006. An Adaptive Algorithm for Selecting Profitable Keywords for Search-based Advertising Services. In *Proc. EC '06*, 260–269.

Soare, M.; Lazaric, A.; and Munos, R. 2014. Best-Arm Identification in Linear Bandits. In *Proc. NIPS'14*, 828–836.

Tao, C.; Blanco, S.; and Zhou, Y. 2018. Best Arm Identification in Linear Bandits with Linear Dimension Dependency. In *Proc. ICML'18*, 4877–4886.

Xu, L.; Honda, J.; and Sugiyama, M. 2018. A fully adaptive algorithm for pure exploration in linear bandits. In *Proc. AISTATS'18*, 843–851.

Zaki, M.; Mohan, A.; and Gopalan, A. 2019. Towards Optimal and Efficient Best Arm Identification in Linear Bandits. *arXiv preprint arXiv:1911.01695* .

Zaki, M.; Mohan, A.; and Gopalan, A. 2020. Explicit Best Arm Identification in Linear Bandits Using No-Regret Learners. *arXiv preprint arXiv:2006.07562* .

Zhong, Z.; Cheung, W. C.; and Tan, V. 2020. Best Arm Identification for Cascading Bandits in the Fixed Confidence Setting. In *Proc. ICML'20*, 11481–11491.

Appendix

A Additional Related Work

In this section, we further review the full-literature of BAI-LB and variants of CPE recently proposed. Comparison between our algorithms and existing algorithms is summarized in Table 1.

Best Arm Identification in Linear Bandits (BAI-LB). Soare et al. (2014) addressed the BAI-LB in the fixed confidence setting and first provided the static allocation algorithm for BAI-LB by introducing the interesting connection between BAI-LB and *G-optimal experimental design* (Pukelsheim 2006). Tao et al. (2018) analyzed the novel randomized estimator based on the convex relaxation of G-optimal design, and devised the adaptive algorithm whose sample complexity depends linearly on the dimension d. Xu et al. (2018) proposed a fully adaptive algorithm inspired by UGapE (Gabillon, Ghavamzadeh, and Lazaric 2012). Karnin (2016) analyzed the explore-verify algorithms for several settings of BAI including linear, dueling, unimodal, and graphical bandits. Fiez et al. (2019) introduced the transductive BAI-LB, and proposed the first non-asymptotic algorithm that nearly achieves the information-theoretic lower bound. Zaki et al. (2019) proposed a generalized LUCB algorithm for BAI-LB with sample complexity analysis for the special cases of two and three arms. Degenne et al. (2020) designed the first asymptotically optimal sampling rules for BAI-LB. Katz-Samuels et al. (2020) proposed near-optimal algorithms for both fixed confidence and fixed budget settings by leveraging the theory of suprema of empirical processes. Although they further discussed a computationally efficient algorithm for CPE-MB with linear rewards, it cannot be applied to CPE-BL since the set of measurement vectors considered in (Katz-Samuels et al. 2020) is also exponentially large for CPE-BL. Zaki, Moha, and Gopalan (2020) proposed the explicitly described algorithm by using tools from game theory and no-regret learning to solve minimax games.

Despite the recent advances in BAI-LB described above, no existing algorithms can solve CPE-BL efficiently, since their computation for distribution λ_* and ways to maintain alive action set cost exponential time in the settings of CPE-BL and CPE-PL. Moreover, no existing algorithms can deal with the nonlinear reward functions or partial linear feedback in CPE-PL.

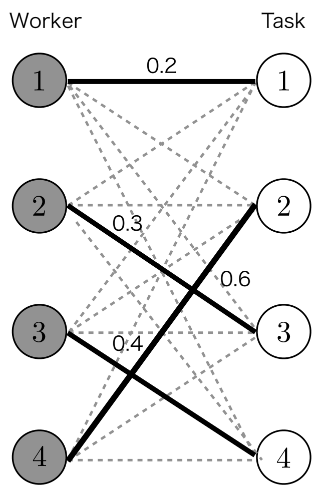


Figure 2: Example of the crowdsourcing application.

Combinatorial Pure Exploration (CPE). For variants in CPE, Huang et al. (2018) designed the algorithm for CPE with continuous and separable reward functions. Although their model considers nonlinear rewards, it cannot deal with either full-bandit or partial linear feedback and thus cannot be applied to our settings. Recently, the best arm identification in cascading bandits is investigated by Zhong et al. (2020). Their problem setting can be seen as another model of top-k identification with partial feedback. However, their algorithm is designed for identifying the top-k actions and thus it cannot be applied to the case under other combinatorial structures such as paths, mathchings, and matroids. Kuroki et al. (2020a) studied a special sub-problem of CPE-PL, where the offline optimization is the densest subgraph problem and the learner observes full-bandit feedback for a set of edges. Their algorithms and analysis use the property of the average degree, and thus they cannot be directly applied to solve either CPE-BL or CPE-PL. Chen et al. (2020) studied an adaptation of CPE to the dueling bandit setting, where at each timestep the leaner plays a pair of edges (base arms) in a bipartite graph and observes a random outcome of the comparison with the objective of identifying the optimal matching. They only consider relative feedback between two compared base arms in the matching case, and thus their algorithms cannot be applied to either CPE-BL or CPE-PL.

B Illustration Examples for CPE-PL

In this section, we will provide specific examples to illustrate the feedback model of CPE-PL by considering the crowdsourcing scenario (see Figure 2). There are $N = 4$ workers and $M = 4$ tasks, which can be represented as a complete bipartite graph $G = (L, R, E)$. In this bipartite graph, each edge corresponds to a pair between a worker and a task, and its edge-weight corresponds to the utility that is unknown to the learner. In the model of CPE-PL, each base arm $i \in \{1, 2, ..., d\}$ prescribes each edge $e \in E$ where $d = |E|$ and its mean is the unknown edge-weight θ_e. Let edges $(1, 1), (2, 3), (3, 4), (4, 2)$ be the first, second, third, fourth base arms respectively. Suppose that random outcome

[1]Notations appearing in the table but not relevant in our problem setting are given below: $\rho(\lambda) = \max_{x \in \mathcal{X}} \|x\|^2_{M(\lambda)^{-1}}$. $\tilde{\Delta}_i = \theta_i - \theta_{k+1}$ if $i \leq k$ and $\theta_k - \theta_i$ otherwise. $H_x = \max\limits_{x_i, x_j \in \mathcal{X}} \frac{\bar{\rho}_x(x_i, x_j)}{\max\{\bar{\Delta}_i^2 \bar{\Delta}_j^2\}}$ where $\bar{\Delta} = (x^* - x_i)^\top \theta$ if $x_i \neq x^*$, $\operatorname{argmin}_{x \in \mathcal{X}} x^* - x$ otherwise, and $\bar{\rho}_x(x_i, x_j)$ is a term defined by the optimal solution to a convex optimization (see (11) in (Xu, Honda, and Sugiyama 2018)). $S_t = \{x \in \mathcal{X} : (x^* - x)^\top \theta \leq 4 \cdot 2^{-t}\}$. $\mathcal{Y}(S_t) = \{x - x' : \forall x, x' \in S_t, x \neq x'\}$. $\tilde{\rho}(\mathcal{Y}(S_t)) = \min_{\lambda \in \triangle(\mathcal{X})} \max_{v \in \mathcal{Y}(S_t)} \|v\|_{M(\lambda)^{-1}}$. $\gamma^* = \min_{\lambda \in \triangle(\mathcal{X})} \mathbb{E}_{\eta \sim N(0,1)} \left[\max_{x \in \mathcal{X} \setminus \{x^*\}} \frac{(x^* - x)^\top M(\lambda)^{-1/2} \eta}{(x^* - x)^\top \theta}\right]^2$.

Table 2: Comparison between our results and existing results for CPE-PL (BL). "General" represents that the algorithm works for any combinatorial structure. $\tilde{O}(\cdot)$ only omits $\log\log$ factors. Main notations are defined in Section 2 and other specific notations are given in the footnote.

Algorithm	Sample complexity [1]	Case	Problem Type	Strategy	Time
GCB-PE (ours, Thm. 2)	$O\big(\frac{\|\sigma\|\beta_\sigma^2 L_p^2}{\Delta_{\min}^2}\log\frac{\beta_\sigma^2 L_p^2}{\Delta_{\min}^2\delta}\big)$	General	CPE-PL	Static	Poly(d)
PolyALBA (ours, Thm. 1)	$\tilde{O}\big(\sum_{i=2}^{\lfloor \frac{d}{2}\rfloor}\frac{1}{\Delta_i^2}\log\frac{\|\mathcal{X}\|}{\delta}+\frac{d^2 m\xi_{\max}(\widetilde{M}(\lambda)^{-1})}{\Delta_{d+1}^2}\log\frac{\|\mathcal{X}\|}{\delta}\big)$	General	CPE-BL	Adaptive	Poly(d)
ICB (Kuroki et al. 2020b)	$\tilde{O}\big(\frac{d\xi_{\max}(M(\lambda)^{-1})\rho(\lambda)}{\Delta_{\min}^2}\log\frac{d\xi_{\max}(M(\lambda)^{-1})\rho(\lambda)}{\Delta_{\min}^2\delta}\big)$	General	CPE-BL	Static	Poly(d)
SAQM (Kuroki et al. 2020b)	$\tilde{O}\big(\frac{d^{1/4}k\xi_{\max}(M(\lambda)^{-1})\rho(\lambda)}{\Delta_{\min}^2}\log\frac{d^{1/4}k\xi_{\max}(M(\lambda)^{-1})\rho(\lambda)}{\Delta_{\min}^2\delta}\big)$	Top-k	CPE-BL	Static	Poly(d)
CSAR (Rejwan and Mansour 2020)	$\tilde{O}\big(\sum_{i=2}^{d}\frac{1}{\tilde{\Delta}_i^2}\log\frac{d}{\delta}\big)$	Top-k	CPE-BL	Adaptive	Poly(d)
$\mathcal{XY}$-static (Soare, Lazaric, and Munos 2014)	$O\big(\frac{d}{\Delta_{\min}^2}\log\frac{\|\mathcal{X}\|}{\delta\Delta_{\min}^2}+d^2\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Static	$\Omega(\|\mathcal{X}\|)$
Explore-Verify (Karnin 2016)	$O\big(\frac{d}{\Delta_{\min}^2}\log\frac{\|\mathcal{X}\|}{\delta\Delta_{\min}}+d\log\delta^{-1}\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Static	$\Omega(\|\mathcal{X}\|)$
LinGapE (Xu, Honda, and Sugiyama 2018)	$\tilde{O}\big(d\sum_{x\in\mathcal{X}}H_x\log\frac{d\|\mathcal{X}\|}{\delta}\cdot\sum_{x\in\mathcal{X}}H_x\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\|\mathcal{X}\|)$
$\mathcal{Y}$-ElimTil (Tao, Blanco, and Zhou 2018)	$\tilde{O}\big(\frac{d}{\Delta_{\min}^2}(\log\delta^{-1}+\log\|\mathcal{X}\|)\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\|\mathcal{X}\|)$
ALBA (Tao, Blanco, and Zhou 2018)	$\tilde{O}\big(\sum_{i=2}^{d}\frac{1}{\Delta_i^2}(\log\delta^{-1}+\log\|\mathcal{X}\|)\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\|\mathcal{X}\|)$
RAGE (Fiez et al. 2019)	$O\big(\sum_{t=1}^{\lfloor\log_2(4/\Delta_{\min})\rfloor}2(2^t)^2\tilde{\rho}(\mathcal{Y}(S_t))\log(t^2\|\mathcal{X}\|^2/\delta)\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\|\mathcal{X}\|)$
LinGame(-C) (Degenne et al. 2020)	$\limsup_{\delta\to 0}\frac{\mathbb{E}_\theta[\tau_\delta]}{\log(1/\delta)}\le\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{2\|\|x^*-x\|\|_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\|\mathcal{X}\|)$
Peace (Katz-Samuels et al. 2020)	$O\big(\big(\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{\|\|x^*-x\|\|_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}+\gamma^*\big)\log(1/\delta)\big)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	Adaptive	$\Omega(\|\mathcal{X}\|)$
Lower Bound (Fiez et al. 2019)	$\mathbb{E}_\theta[\tau_\delta]\ge\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{\|\|x^*-x\|\|_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}\log(1/2.4\delta)$	$\mathcal{X}\subseteq\mathbb{R}^d$	BAI-LB	-	-

at round t is $\theta+\eta_t=(0.2,0.3,0.4,0.6,\ldots)^\top\in\mathbb{R}^d$, and the learner will pull the action $x_t=(1,1,1,1,0,0,\ldots,0)^\top$. In this case, the examples of transformation matrix $M_x\in\mathbb{R}^{m_x\times d}$ and corresponding random feedback $y_t=M_{x_t}(\theta+\eta_t)\in\mathbb{R}^{m_{x_t}}$ can be written as follows.

If $M_{x_t}=(1,0,0,\ldots,0)$, the learner can observe only first base arm, that is,

$$M_{x_t}(\theta+\eta_t)=(1,0,0,\ldots,0)\begin{pmatrix}0.2\\0.3\\0.4\\0.6\\\vdots\end{pmatrix}=0.2=y_t.$$

If $M_{x_t}=\mathrm{diag}(x)$, the learner can obtain semi-bandit feedback, that is,

$$\begin{aligned}M_{x_t}(\theta+\eta_t)&=\begin{pmatrix}1&0&0&0&\cdots\\0&1&0&0&\cdots\\0&0&1&0&\cdots\\0&0&0&1&\cdots\end{pmatrix}\begin{pmatrix}0.2\\0.3\\0.4\\0.6\\\vdots\end{pmatrix}\\&=\begin{pmatrix}0.2\\0.3\\0.4\\0.6\end{pmatrix}\\&=y_t.\end{aligned}$$

If $M_{x_t}=x_t^\top$, the learner can only observe the sum of the rewards (i.e., full-bandit feedback), that is,

$$M_{x_t}(\theta+\eta_t)=(1,1,1,1,0,\ldots,0)\begin{pmatrix}0.2\\0.3\\0.4\\0.6\\\vdots\end{pmatrix}=1.5=y_t.$$

For other cases, the learner may obtain the sum of the rewards from two base arms as follows,

$$\begin{aligned}M_{x_t}(\theta+\eta_t)&=\begin{pmatrix}1&1&0&0&\cdots\\0&0&1&1&\cdots\end{pmatrix}\begin{pmatrix}0.2\\0.3\\0.4\\0.6\\\vdots\end{pmatrix}\\&=\begin{pmatrix}0.5\\1.0\end{pmatrix}\\&=y_t.\end{aligned}$$

Note that reward functions in CPE-PL can be nonlinear, while feedback model is linear as illustrated above.

C More Discussion on the Optimality of PolyALBA

For BAI-LB, Fiez et al. (2019) propose the first sample complexity lower bound

$$\mathbb{E}_\theta[\tau_\delta]\ge\min_{\lambda\in\triangle(\mathcal{X})}\max_{x\in\mathcal{X}\setminus\{x^*\}}\frac{\|\|x^*-x\|\|_{M(\lambda)^{-1}}^2}{((x^*-x)^\top\theta)^2}\log(1/2.4\delta),$$

and a nearly optimal algorithm RAGE with sample complexity

$$O\left(\sum_{t=1}^{\lfloor \log_2(4/\Delta_{\min})\rfloor} 2(2^t)^2 \tilde{\rho}(\mathcal{Y}(S_t)) \log(t^2|\mathcal{X}|^2/\delta)\right),$$

where $S_t = \{x \in \mathcal{X} : (x^* - x)^\top \theta \leq 4 \cdot 2^{-t}\}$, $\mathcal{Y}(S_t) = \{x - x' : \forall x, x' \in S_t, x \neq x'\}$ and $\tilde{\rho}(\mathcal{Y}(S_t)) = \min_{\lambda \in \triangle(\mathcal{X})} \max_{v \in \mathcal{Y}(S_t)} \|v\|_{M(\lambda)^{-1}}$. Fiez et al. (2019) prove that this upper bound for RAGE matches the lower bound with a logarithmic factor.

Below we show that for a family of instances, the sample complexity of PolyALBA matches that of RAGE and also achieves near-optimality. Consider a family of instances such that $\Delta_{\lfloor d/2^{(t-2)}\rfloor+1} = 4 \cdot 2^{-t},\ t = 2, 3, \dots, \log_2(\frac{4}{\Delta_{\min}})$, which can be easily found in practical sub-problems. For example, in the *Multi-Bandit* case (Gabillon et al. 2011), we are given base arms numbered by $1, \dots, 10$, and a feasible super arm consists of two base arms respectively from $\{1, \dots, 5\}$ and $\{6, \dots, 7\}$, where $d = 10$ and $|\mathcal{X}| = 25$. Let $\theta = (2.5, 2, 1.5, 1, 0.5, 0.625, 0.5, 0.375, 0.25, 0.125)^\top$. We use x_i and A_i to denote the indicator vector and the set of base arm indices for the i-th best super arm. Then, $A_1 = \{1, 6\}, x_1^\top \theta = 3.125, A_2 = \{1, 7\}, x_2^\top \theta = 3, A_3 = \{1, 8\}, x_3^\top \theta = 2.875, A_6 = \{2, 6\}, x_6^\top \theta = 2.625, A_{11} = \{3, 6\}, x_{11}^\top \theta = 2.125$, which belongs to the considered family of instances.

In such family of instances, our PolyALBA performs similar elimination procedure as RAGE. Specifically, since the sample complexity upper bound of PolyALBA guarantees that for epoch q, any action x with $\Delta_x \geq \Delta_{\lfloor d/2^q \rfloor+1}$ will be discarded, we have that $\forall x' \in S_q, \Delta_{x'} \leq \Delta_{\lfloor d/2^{q-1}\rfloor+1} = 2 \cdot 2^{-q},\ q = 1, 2, \dots, \log_2(\frac{4}{\Delta_{\min}})$, which is the same to the gap constraint for S_t in RAGE. Then, in epoch $q = 2, \dots, \log_2(\frac{4}{\Delta_{\min}})$, PolyALBA essentially performs the same procedure as RAGE, while in epoch $q = 1$, PolyALBA performs a higher samples (the $\xi_{\max}(\tilde{M}^{-1}(\lambda))$-related term) than RAGE. Formally, in such instances PolyALBA has sample complexity

$$O\Bigg(\sum_{t=2}^{\lfloor \log_2(4/\Delta_{\min})\rfloor} 2(2^t)^2 \tilde{\rho}(\mathcal{Y}(S_t)) \log(t^2|\mathcal{X}|^2/\delta) + md\xi_{\max}(\tilde{M}^{-1}(\lambda))\tilde{\rho}(\mathcal{Y}(S_1)) \log(|\mathcal{X}|^2/\delta)\Bigg).$$

For sufficiently small $\Delta_{\min}$ (increase d in the above Multi-Bandit case to obtain small $\Delta_{\min}$), the additional term related to $\xi_{\max}(\tilde{M}^{-1}(\lambda))$ is absorbed and the result is the same to that of RAGE, which matches the lower bound within a logarithmic factor. The sample complexity of PolyALBA shows superiority over other heavily $\Delta_{\min}$-dependent algorithms (Soare, Lazaric, and Munos 2014; Karnin 2016; Kuroki et al. 2020b), and the additional term related to $\xi_{\max}(\tilde{M}^{-1}(\lambda))$ can be viewed as a cost for achieving computational efficiency.

We also remark that PolyALBA is close to the lower bound in the worst-case instances and our sample complexity has an interesting relation with that of G-allocation strategy for BAI-LB (Soare, Lazaric, and Munos 2014). Soare, Lazaric, and Munos (2014) proved the following lemma to bound the *oracle complexity* H_{LB} defined as follows, which also appears in the information theoretic lower bound (Fiez et al. 2019).

$$H_{LB} = \min_{\lambda \in \triangle(\mathcal{X})} \max_{x \in \mathcal{X} \setminus \{x^*\}} \frac{\|x^* - x\|^2_{M(\lambda)^{-1}}}{((x^* - x)^\top \theta)^2}.$$

Lemma 2 (Lemma 2, Soare, Lazaric, and Munos (2014))**.** *Given an arm set $\mathcal{X} \subseteq \mathbb{R}^d$ and parameter θ, the complexity H_{LB} is such that*

$$\max_{x \in \mathcal{X} \setminus \{x^*\}} \frac{\|x^* - x\|_2^2}{L_x \Delta_{\min}^2} \leq H_{\mathrm{LB}} \leq \frac{4d}{\Delta_{\min}^2},$$

where L_x is the upper bound of the ℓ_2-norm of any $x \in \mathcal{X}$. Furthermore, if $\mathcal{X}$ is the canonical basis, the problem reduces to a MAB and $\sum_{i=1}^{|\mathcal{X}|} 1/\Delta_i \leq H_{\mathrm{LB}} \leq 2\sum_{i=1}^{|\mathcal{X}|} 1/\Delta_i$.

Soare, Lazaric, and Munos (2014) designed the G-allocation strategy and its sample complexity is $O\left(\frac{4d}{\Delta^2_{\min}} \log \frac{|\mathcal{X}|}{\delta}\right)$, which shows that G-allocation strategy is optimal within logarithmic terms for instances where the worst-case value of H_{LB} is given (See Theorem 1 in their paper). For instances with $\Delta_i = \Delta_{\min}, \forall i > 1$, PolyALBA has the sample complexity of $\tilde{O}\left(\frac{md^2\xi_{\max}(\tilde{M}^{-1}(\lambda))}{\Delta^2_{\min}} \log \frac{|\mathcal{X}|}{\delta}\right)$, which matches that of G-allocation strategy up to a factor of $md\xi_{\max}(\tilde{M}^{-1}(\lambda)))$ (when we ignore the $\log\log$ terms). PolyALBA also employs G-optimal design for its static phase but we restrict its support in order to make it running in polynomial time, which causes the additional term related to $\xi_{\max}(\tilde{M}^{-1}(\lambda))$. Note that the lower bound (Fiez et al. 2019) does not consider time complexity, and the gap in the additional term may be a price paid for achieving computational efficiency.

To sum up, to our best knowledge, PolyALBA is the first polynomial-time adaptive algorithm that works for CPE-BL with general combinatorial structures and achieves nearly optimal sample complexity for a family of problem instances. It is very interesting to study a lower bound for polynomial-time CPE-BL algorithms and it remains open to design efficient algorithms with instance-wise optimal sample complexity.

D Naive Reduction of Top-k and Analysis of a UCB-based Algorithm for CPE-BL

In this section, we briefly explain a naive reduction to the classic CPE-MB for CPE-BL in the top-k setting, and note that there is no simple reduction for general CPE-BL by showing an undesirable property of a UCB-based algorithm with a regularized least-square estimator.

We first remark that with only $O(k)$ more samples, the problem of top-k identification with full-bandit feedback can be solved by classic top-k algorithms in which base arms are queried. Suppose that an algorithm has a sample complexity of $C_{\delta,\Delta}$ in classic setting, it yields a complexity of $\tilde{O}(k \cdot C_{\delta,\Delta})$ for the full-bandit setting, where $\tilde{O}$ omits some log factors. This is due to the fact that the unbiased estimate

Algorithm 7: Combinatorial lower-upper naive confidence bound (CLUNCB)

```
   Input   : Accuracy $\epsilon > 0$, confidence level $\delta \in (0,1)$
1  Initialization For each $e \in [d]$, pull $x_e \in \mathcal{X}$ such that
   $e \in x_e$ once. Initialize $A^{\iota}_{\mathbf{x}_t}$ and $b_t$;
2  while $\widetilde{\theta}_t^{\top}\widetilde{x}_t - \widetilde{\theta}^{\top}\widehat{x}^*_t \le \epsilon$ is not true do
3      $t \leftarrow t+1$;
4      $\widehat{x}^*_t \leftarrow \mathrm{argmax}_{x\in\mathcal{X}}\, \widehat{\theta}_t^{\top} x$;
5      Set $\mathrm{rad}_t(e) = C_t\sqrt{(A^{\iota}_{\mathbf{x}_t})^{-1}(e,e)}$ for all $e \in [d]$;
6      for $e = 1,\dots,d$ do
7          if $e \in \widehat{x}^*_t$ then $\widetilde{\theta}_t(e) \leftarrow \widehat{\theta}_t(e) - \mathrm{rad}_t(e)$;
8          else $\widetilde{\theta}_t(e) \leftarrow \widehat{\theta}_t(e) + \mathrm{rad}_t(e)$;
9      $\widetilde{x}_t \leftarrow \mathrm{argmax}_{x\in\mathcal{X}}\, \widetilde{\theta}_t^{\top} x$;
10     $p_t \leftarrow \mathrm{argmax}_{e\in(\widetilde{x}_t\setminus\widehat{x}^*_t)\bigcup(\widehat{x}^*_t\setminus\widetilde{x}_t)}\, \mathrm{rad}_t(e)$;
11     Sample any $x_t \in \mathcal{X}$ such that $p_t \in x_t$;
12     Update $A^{\iota}_{\mathbf{x}_t}$, $b_t$ and $\widehat{\theta}_t$;
13 Return Out $\leftarrow \widehat{x}^*_t$
```

can be obtained for the difference between the two base arms by comparing two k-base arm queries with one base arm difference. Formally, with any fixed base arm i_0, one can get an unbiased estimate for the gap $\theta_j - \theta_{i_0}$ with $O(k)$ times larger variance by querying two super-arms $S \cup \{j\}$ and $S \cup \{i_0\}$ for $S \subseteq [d]$ such that $|S| = k-1$ and $j, i_0 \notin S$. Therefore, when $C_{\delta,\Delta}$ has dependence of $\sum_{i\in[d]} \Delta_i^{-2}$, we also have a sample complexity which has dependence of $\sum_{i\in[d]} \Delta_i^{-2}$for top-$k$ case with full-bandit feedback. However, for more complex cases such as matroid, matroid intersection, and s-t path, we cannot use such a reduction due to its combinatorial constraint.

We show that with a simple modification using the regularized least-square estimator, CLUCB algorithm proposed in (Chen et al. 2014) can work for CPE-BL for general constraints such as top-k, matroid, matroid intersection, s-t path, and it is a polynomial-time (ϵ,δ)-PAC algorithm. However, we prove that this naive adaption can be sub-optimal and the sample complexity depends on $\frac{1}{\Delta^2_{\min}}$ in the worst case.

D.1 Preliminary

We call an algorithm *fully adaptive* if it changes the arm-selection strategy based on the past observation at all rounds. For such an adaptive algorithm, we cannot use the ordinary least-square estimator for $\theta \in \mathbb{R}^d$ as an unbiased estimator. Instead we will use the regularized least-square estimator. If the sequence of super-arm selections $\mathbf{x}_t = (x_1,\dots,x_t)$ is adaptively determined based on the past observations, the regularized least-square estimator is given by

$$\widehat{\theta}_t = (A^{\iota}_{\mathbf{x}_t})^{-1} b_{\mathbf{x}_t}, \tag{1}$$

where $A^{\iota}_{\mathbf{x}_t}$ and $b_{\mathbf{x}_t}$ is defined by

$$A^{\iota}_{\mathbf{x}_t} = \iota I + \sum_{i=1}^{t} x_i x_i^{\top}, \text{ and } b_{\mathbf{x}_t} = \sum_{i=1}^{t} x_i r_i \in \mathbb{R}^n.$$

for regularization parameter $\iota > 0$ and the identity matrix I. If we set $\iota = 0$ and we are allowed to sample a base arm, i.e., unit vector at all rounds, it is easy to see that $A^{\iota}_{\mathbf{x}_t}(i,i) = T_i(t)$ for $i \in [d]$ and $A^{\iota}_{\mathbf{x}_t}(i,j) = 0$ for $i \neq j$, where $T_i(t)$ is the number of times that base arm i is sampled before round $t+1$.

Abbasi-Yaddkori et al. (2011) showed the high probability bound for the regularized least-squares estimator $\hat{\theta}$.

Proposition 1 (Theorem 2 in Abbasi-Yaddkori et al. (2011))**.** *Let $\hat{\theta}_t$ be the regularized least-squares estimator. Suppose that a noise η_t is κ-sub-Gaussian. If the ℓ_2-norm of parameter θ is less than L, then for all $i \in [d]$ and for every adaptive sequence $\mathbf{x_t}$,*

$$|x^{\top}\theta - x^{\top}\widehat{\theta}_t| \le C_t \|x\|_{(A^{\iota}_{\mathbf{x}_t})^{-1}}$$

holds for all $t \in \{1,2,\dots\}$ and $\forall x \in \mathbb{R}^n$ with probability at least $1-\delta$, where

$$C_t = \kappa\sqrt{2\log\frac{\det(A^{\iota}_{\mathbf{x}_t})^{\frac{1}{2}}}{\iota^{\frac{n}{2}}}} + \iota^{\frac{1}{2}} L. \tag{2}$$

Moreover, if $\|x\|_2 \le \sqrt{m}$ holds for all $t > 0$, then

$$C_t \le \kappa\sqrt{d\log\frac{1+tm/\iota}{\delta}} + \iota^{\frac{1}{2}} L. \tag{3}$$

We also introduce the notion of the *width* for a decision set $\mathcal{X}$ defined in Chen et al. (2014); $\mathrm{width}(\mathcal{X})$ prescribes the size of the thinnest exchange class (see Chen et al. (2014) for detailed definition). For example, if $\mathcal{X}$ are independent sets of ground set $[d]$, $\mathrm{width}(\mathcal{X}) \le 2$.

D.2 Analysis of CLUCB for CPE-BL

In the setting where each base arm is pulled at all rounds, the confidence radius is simply defined as $\mathrm{rad}_t(e) = \sqrt{\frac{2\log\left(\frac{4dt^3}{\delta}\right)}{T_e(t)}}$ for all $e \in [d]$. Since we are not allowed to pull each base arm, we cannot define such a radius as the above form. However, we have concentration inequalities for each unit vector of e, and thus we can construct the confidence radius in the full-bandit setting. From Proposition 1, we can construct the high probability confidence radius as follows.

Lemma 3. *Suppose that a reward from each base arm follows a 1-sub-Gaussian distribution for all $i \in [d]$. For all $t > 0$ and all $i \in [d]$, the confidence radius $\mathrm{rad}_t(i)$ is defined as*

$$\mathrm{rad}_t(i) = C_t\sqrt{(A^{\iota}_{\mathbf{x}_t})^{-1}(i,i)} \quad (\forall i \in [d]), \tag{4}$$

where C_t is given by (2)*. Let $\mathbf{rad}_t$ be an d-dimensional vector with nonnegative entries. For $\mathbf{rad}_t$, define random event $\mathcal{E}_t$ for all $t > 0$ as follows.*

$$\mathcal{E}_t = \{\forall i \in [d],\ |\theta(i) - \hat{\theta}_t(i)| \le \mathrm{rad}_t(i)\} \tag{5}$$

Then we have

$$\Pr\left[\bigcap_{t=1}^{\infty} \mathcal{E}_t\right] \ge 1-\delta. \tag{6}$$

The proof is omitted since it is straightforward from Proposition 1 and union bounds. Using the above confidence radius, we can design CLUCB-based algorithm for CPE-BL, which is detailed in Algorithm 7. We show that Algorithm 7 is (ϵ, δ)-PAC and its sample complexity bound is given in Corollary 1. As can be seen, the sample complexity depends on $\Delta_{\min}^{-2}$ in the worst case. Also, since CLUNCB is fully adaptive, we cannot completely control λ_{C} beforehand and thus $M(\lambda_{\mathrm{C}})^{-1}(e,e) \leq \lambda_{\max}(M(\lambda_{\mathrm{C}})^{-1})$ can be large.

Corollary 1. *Let $\lambda_{\mathrm{C}} \in \triangle(\mathcal{X})$ be a distribution in which $\lambda_{\mathrm{C}}(x)$ represents the ratio that x is pulled by CLUNCB. The total number of samples T is bounded as*

$$T = O\left(d\kappa^2 \tilde{H} \log\left(\frac{d\kappa^2 m\tilde{H}/\iota + \log\delta^{-1}}{\delta}\right)\right),$$

where $\tilde{H}$ is defined as

$$\tilde{H} = \max_{e\in[d]}\left(M(\lambda_{\mathrm{C}})^{-1}(e,e)\min\left\{\frac{9\mathrm{width}(\mathcal{X})^2}{\Delta_e^2}, \frac{4m^2}{\epsilon d^2}\right\}\right).$$

Proof. First, we state the following two lemmas in Chen et al. (2014); Lemma 4 (Lemma 12 in Chen et al. (2014)) shows that if the confidence radius is valid, then CLUCB always outputs ϵ-optimal set, and Lemma 5 (Lemma 13 in Chen et al. (2014)) implies that if the confidence radius of an arm is sufficiently small, then the arm will not be chosen as p_t. Note that we have the lemmas since Lemmas 3, 5, 7, and 10 in Chen et al. (2014) also hold for our setting where $\mathbf{rad}_t$ is given by (4) and the empirical mean is replaced with the least square estimator $\hat{\theta}_t$ in (1).

Lemma 4 (Lemma 12 in Chen et al. (2014))**.** *If CLUCB stops on round t and suppose that $\mathcal{E}_t$ occurs. Then, we have $\theta^\top x^* - \theta^\top x_{\mathit{Out}} \leq \epsilon$.*

Lemma 5 (Lemma 13 in Chen et al. (2014))**.** *Given any t and suppose that event $\mathcal{E}_t$ occurs. For any $e \in [d]$, if $\mathrm{rad}_t(e) < \max\left\{\frac{\Delta_e}{3\mathrm{width}(\mathcal{X})}, \frac{\epsilon}{2m}\right\}$, then $p_t \neq e$.*

The random event $\bigcap_{t=1}^{\infty}\mathcal{E}_t$ occurs with probability at least $1-\delta$ from Lemma 3. From Lemma 4, under the event $\bigcap_{t=1}^{\infty}\mathcal{E}_t$, CLUNCB returns an ϵ-optimal set. Therefore, in the rest of part, we shall assume this event holds.

Fix any arm $e \in [d]$ and let τ_e be the last round which arm e is chosen as p_t. From (3) and for a small $\iota \leq \frac{\kappa^2}{L^2}\log\delta^{-1}$, we have

$$\begin{aligned} C_{\tau_e} \leq& \kappa\sqrt{n\log\frac{1+\tau_e m/\iota}{\delta}} + \iota^{\frac{1}{2}}L \\ \leq& 2\kappa\sqrt{d\log\left(\frac{1+\tau_e m/\iota}{\delta}\right)}. \end{aligned}$$

By Lemma 5, we have $\mathrm{rad}_{\tau_e}(e) \geq \max\left\{\frac{\Delta_e}{3\mathrm{width}(\mathcal{X})}, \frac{\epsilon}{2m}\right\}$. We define $\Lambda_{\mathbf{x}_{\tau_e}} = \frac{A^{\iota}_{\mathbf{x}_t}}{\tau_e}$. Then, we have

$$\begin{aligned} &\max\left\{\frac{\Delta_e^2}{9\mathrm{width}(\mathcal{X})^2}, \frac{\epsilon^2}{4m^2}\right\} \leq \mathrm{rad}_t^2(e) \\ &= C_{\tau_e}^2 (A^{\lambda}_{\mathbf{x}_{\tau_e}})^{-1}(e,e) \\ &\leq 4\kappa^2 d\log\left(\frac{1+\tau_e m/\iota}{\delta}\right)(A^{\lambda}_{\mathbf{x}_{\tau_e}})^{-1}(e,e) \\ &= 4\kappa^2 d\log\left(\frac{1+\tau_e m/\iota}{\delta}\right)\frac{(\Lambda^{\iota}_{\mathbf{x}_{\tau_e}})^{-1}(e,e)}{\tau_e}. \end{aligned}$$

That is, we obtain

$$\tau_e \leq H_e \log\left(\frac{1+\tau_e m/\iota}{\delta}\right),$$

where we define $H_e = 4\kappa^2 d(\Lambda^{\iota}_{\mathbf{x}_{\tau_e}})^{-1}(e,e)\min\left\{\frac{9\mathrm{width}(\mathcal{X})^2}{\Delta_e^2}, \frac{4m^2}{\epsilon^2}\right\}$. Let $\tau'(\leq \tau_e)$ satisfying

$$\tau_e = H_e\left(\log\left(1+\frac{\tau' m}{\iota}\right) + \log\frac{1}{\delta}\right). \tag{7}$$

Then, we have

$$\begin{aligned} \tau' \leq \tau_e &= H_e\left(\log\left(1+\frac{\tau' m}{\iota}\right)+\log\frac{1}{\delta}\right) \\ &\leq H_e\left(\sqrt{\frac{\tau' m}{\iota}} + \log\frac{1}{\delta}\right). \end{aligned} \tag{8}$$

Solving (8) for $\sqrt{\tau'}$, we obtain

$$\begin{aligned} \sqrt{\tau'} \leq& \frac{1}{2}\left(H_e\sqrt{\frac{m}{\iota}} + \sqrt{H_e^2\frac{m}{\iota} + 4H_e\log\frac{1}{\delta}}\right) \\ \leq& 2\sqrt{H_e^2\frac{m}{4\iota} + H_e\log\frac{1}{\delta}}. \end{aligned}$$

That is, we see that $\tau' = O\left(H_e^2\frac{m}{\iota} + H_e\log\frac{1}{\delta}\right)$, which shows that

$$\log\left(\frac{1+\tau' m}{\iota}\right) = O\left(\log\left(\frac{mH_e}{\iota} + \log\frac{1}{\delta}\right)\right). \tag{9}$$

Combining (9) into (7), we obtain

$$\tau_e = O\left(H_e\log\left(\frac{mH_e/\iota + \log\delta^{-1}}{\delta}\right)\right).$$

The number of samples used by CLUNCB is $T = \max_{e\in[d]}\tau_e$.

Recall that $\lambda_{\mathrm{C}} \in \triangle(\mathcal{X})$ is a distribution in which $\lambda_{\mathrm{C}}(x)$ represents the ratio that x was pulled by CLUNCB. Suppose that T is sufficiently large such that $\Lambda_{\mathbf{x_T}} = \frac{A^{\iota}_{\mathbf{x}_t}}{T} \approx M(\lambda_{\mathrm{C}})$. Define

$$\tilde{H} = \max_{e\in[d]}\left((M(\lambda_{\mathrm{C}})^{-1}(e,e)\min\left\{\frac{9\mathrm{width}(\mathcal{X})^2}{\Delta_e^2}, \frac{4m^2}{\epsilon d^2}\right\}\right)$$

Then, we have

$$T = \max_{e\in[d]}\tau_e = O\left(d\kappa^2\tilde{H}\log\left(\frac{d\kappa^2 m\tilde{H}/\iota + \log\delta^{-1}}{\delta}\right)\right).$$

□

E Entropic Mirror Descent Algorithm for Computing the Distribution $\lambda^*_{\mathcal{X}_\sigma}$

For completeness, we provide the algorithm for computing the optimal distribution $\lambda^*_{\mathcal{X}_\sigma} = \min_{\lambda\in\triangle(\mathcal{X}_\sigma)} \max_{x\in\mathcal{X}_\sigma} x^\top M(\lambda)^{-1}x$ for a given set of actions $\mathcal{X}_\sigma \subseteq \mathcal{X}$, which is used in both ALBA (Tao, Blanco, and Zhou 2018) and our method PolyALBA. Algorithm 8 details the entropic mirror descent algorithm proposed in Tao, Blanco, and Zhou (2018).

Algorithm 8: The entropic mirror descent for computing $\lambda^*_{\mathcal{X}_\sigma}$ (Tao, Blanco, and Zhou 2018)

```
Input :d-set of base arms $[d]$, a set of super arms
       $\mathcal{X}_\sigma \subseteq \mathcal{X}$, Lipschitz constant $L_f$ of function
       $\log\det M(\lambda)$ and tolerance $\epsilon$
1 Choose , such that $\mathrm{rank}(X) = d$ where
  $X = (b_1, \dots, b_d)$;
2 Initialize $t \leftarrow 1$ and $\lambda^{(1)} \leftarrow (1/|\mathcal{X}_\sigma|, \dots, 1/|\mathcal{X}_\sigma|)$;
3 while $|\max_{x\in\mathcal{X}_\sigma} x^\top M(\lambda^{(t)})^{-1}x| - d \geq \epsilon$ do
4     $a_t \leftarrow \frac{\sqrt{2\ln|\mathcal{X}_\sigma|}}{L_f\sqrt{t}}$ ;
5     Compute gradient
      $G_i^{(t)} \leftarrow \mathrm{Tr}(M(\lambda^{(t)})^{-1}(x_i x_i^\top))$;
6     Update $\lambda_i^{(t+1)} \leftarrow \frac{\lambda_i^{(t)}\exp(a_t G_i^{(t)})}{\sum_{i=1}^{|\mathcal{X}_\sigma|}\lambda_i^{(t)}\exp(a_t G_i^{(t)})}$;
7     $t \leftarrow t + 1$;
8 $\lambda^*_{\mathcal{X}_\sigma} \leftarrow \lambda^{(t)}$;
  Output : $\lambda^*_{\mathcal{X}_\sigma}$
```

F Discussion on Improving α

In this section, we further discuss the approximation algorithm for computing $\lambda \in \triangle(\mathcal{X})$ and provide Algorithm 9, which can be one alternative of Algorithm 5 in PolyALBA. Lemma 1 indicates that choosing $\mathrm{supp}(\lambda)$ that maximizes $\xi_{\min}(\widetilde{M}(\lambda))$ gives the better bound. Such a design is so called the *E-optimal design*, i.e., the goal is to minimize the maximum eigenvalue of the error covariance (Pukelsheim 2006). As noted in the main paper, if we are allowed to pull unit vectors, we have $\xi_{\min}(\widetilde{M}(\lambda)) \geq 1$. For general cases, more sophisticated algorithm by the ellipsoid method (Grötschel, Lovász, and Schrijver 1981) is considered in this section as one alternative of Algorithm 5. This approach provides the better choice of λ (or equivalently α) than Algorithm 5 in terms of the expectation value.

Let $\triangle(\mathcal{X})_{\mathrm{poly}}$ be the subset of probability distributions over $\mathcal{X}$ with polynomial-size support. Our task is to find an approximate solution $\tilde{\lambda} \in \triangle(\mathcal{X})_{\mathrm{poly}}$ to the following minmax optimization:

$$\min_{\lambda\in\triangle(\mathcal{X})} \max_{x\in\mathcal{X}} \|x\|_{M(\lambda)^{-1}}.$$

We denote the exact G-optimal design by $\lambda^* = \mathrm{argmin}_{\lambda\in\triangle(\mathcal{X})} \max_{x\in\mathcal{X}} \|x\|_{M(\lambda)^{-1}}$. The above minmax optimization is computationally intractable in combinatorial settings, while many existing methods in linear bandits involved

Algorithm 9: Approximation algorithm for G-optimal design by the ellipsoid method

```
Input :d-set of base arms $[d]$, $n \in \mathbb{Z}_+$, $\tilde{n} \in \mathbb{Z}_+$.
1 for $i = 1, \dots, n$ do
2     Choose $\mathcal{X}_{\sigma_i} \leftarrow$ any $\tilde{n}$-super arms;
3     Compute $\lambda_i \leftarrow \lambda^*_{\mathcal{X}_{\sigma_i}}$ by Algorithm 8 for $\mathcal{X}_{\sigma_i}$;
4     Compute $w_{\lambda_i} \in \mathbb{R}^d_+$ by setting
      $w_{\lambda_i} = (\sum_{j\in[d]} |M(\lambda_i)^{-1/2}_{e,j}|)_{e\in[d]} \in \mathbb{R}^d_+$ ;
5 Perform the ellipsoid method to solve $\mathrm{LP_{primal}}$:
    $\mathrm{LP_{primal}}$ : min. $\nu$                                   (10)
        s.t. $\nu \geq \sum_{i\in[n]} h_i \sum_{e\in[d]} w_{\lambda_i,e} x_e,\ (\forall x \in \mathcal{X})$
6            $h \in \triangle([n])$.
7 $h^* \leftarrow$ optimal solution to $\mathrm{LP_{primal}}$;
8 $\nu^* \leftarrow$ optimal value of $\mathrm{LP_{primal}}$
9 Sample $i^* \in [n]$ from $h^* \in \triangle([n])$;
10 $\alpha \leftarrow \min\left\{\frac{\nu^*}{\sqrt{d}}, \min_{i\in[n]} \sqrt{\frac{md}{\xi_{\min}(\widetilde{M}(\lambda_i))}}\right\}$ ;
  Output : $\lambda_{i^*}$ and $\alpha$
```

the brute force to solve G-optimal design problem (Fiez et al. 2019; Soare, Lazaric, and Munos 2014; Tao, Blanco, and Zhou 2018). To avoid a intractable brute force, we address a relaxation problem for the minmax optimization, i.e., a randomized mixed strategy for the robust combinatorial optimization. However, since the ellipsoidal norm $\|x\|_{M(\lambda)^{-1}}$ has the quadratic form, such a relaxation problem is still hard to compute. To overcome this challenge, we consider a simpler norm instead of the ellipsoidal norm. In higher level, this idea is similar to that of Dani, Hayes, and Kakade (2008); they use skewed octahedron called $\mathrm{ConfidenceBall}_1$ as its confidence region rather than the ellipsoid. The radius of $\mathrm{ConfidenceBall}_1$ has been set large enough such that it contains the confidence ellipsoid as an inscribed subset. Whereas they use 1-norm $\|M(\lambda)^{1/2}x\|_1$ to define $\mathrm{ConfidenceBall}_1$, however, $\max_{x\in\mathcal{X}} \|M(\lambda)^{1/2}x\|_1$ is still intractable in the combinatorial action space. To avoid the computational hardness, we introduce a linear function $g_\lambda : \{0,1\}^d \to \mathbb{R}_+$ in order to utilize the underlying combinatorial structure. We define $w_\lambda = (\sum_{j\in[d]} |M(\lambda)^{-1/2}_{i,j}|)_{i\in[d]} \in \mathbb{R}^d_+$ for $\lambda \in \triangle(\mathcal{X})$. For $\lambda \in \triangle(\mathcal{X})$, a linear function $g_\lambda : \{0,1\}^d \to \mathbb{R}_+$ is represented as $g_\lambda(x) = \sum_{e\in[d]} w_{\lambda,e} x_e$. We shall assume that the Ellipsoid method computes the optimal solution for linear programmings by enough iterations (Grötschel, Lovász, and Schrijver 1981).

We show that the optimal value ν^* in Algorithm 9 gives an upper bound of the confidence ellipsoidal norm.

Lemma 6. *Let $\mathcal{X}$ be a family of super arms satisfying given constraints such as the matroid, matroid intersection, and s-t path. Let $\lambda^{h*} \in \triangle(\mathcal{X})_{\mathrm{poly}}$ be an output by Algorithm 9. Let h^* be an optimal solution and ν^* be the optimal value for* $\mathrm{LP_{primal}}$*. Then, λ^{h*} satisfies*

$$\max_{x\in\mathcal{X}} \mathbb{E}[\|x\|_{M(\lambda^{h*})^{-1}}] \leq \nu^*.$$

Proof. By the definition of $w_\lambda = (\sum_{j\in[d]} |M(\lambda)^{-1/2}_{i,j}|)_{i\in[d]}$ and definitions of the quadratic norm and 1-norm, we have

$$\begin{aligned}\|x\|_{M(\lambda)^{-1}} &= \|M(\lambda)^{-1/2}x\|_2\\ &\leq ||M(\lambda)^{-1/2}x||_1\\ &\leq \sum_{e\in[d]} w_{\lambda,e}x_e\\ &= g_\lambda(x) \ (\forall x \in \{0,1\}^d). \end{aligned} \tag{11}$$

Let $y^* = \mathrm{argmax}_{x\in\mathcal{X}} \|x\|_{M(\lambda)^{-1}}$. From Eq. (11), it holds that

$$\max_{x\in\mathcal{X}} \|x\|_{M(\lambda)^{-1}} = \|y^*\|_{M(\lambda)^{-1}} \leq g_\lambda(y^*). \tag{12}$$

Recall that $\lambda^{h*} = \lambda_{i^*}$ where i^* was sampled from $h^* \in \triangle([n])$ in Algorithm 9; we have $\mathbb{E}[w_{\lambda^{h*},e}] = \sum_{i\in[n]} h^*_i w_{\lambda_i,e}$ for all $e \in [d]$. Thus, for any $x \in \mathcal{X}$, we see that

$$\mathbb{E}[g_{\lambda^{h*}}(x)] = \sum_{i\in[n]} h^*_i g_{\lambda_i}(x).$$

From the above, we have that

$$\begin{aligned}\max_{x\in\mathcal{X}} \mathbb{E}[\|x\|_{M(\lambda^{h*})^{-1}}] &\leq \max_{x\in\mathcal{X}} \mathbb{E}[g_{\lambda^{h*}}(x)]\\ &= \max_{x\in\mathcal{X}} \sum_{i\in[n]} h^*_i g_{\lambda_i}(x).\end{aligned} \tag{13}$$

Thus, we obtain

$$\begin{aligned}\max_{x\in\mathcal{X}} \mathbb{E}[\|x\|_{M(\lambda^{h*})^{-1}}] &\leq \max_{x\in\mathcal{X}} \sum_{i\in[n]} h^*_i g_{\lambda_i}(x)\\ &= \min_{h\in\triangle([n])} \max_{x\in\mathcal{X}} \sum_{i\in[n]} h_i g_{\lambda_i}(x)\\ &= \nu^*\end{aligned}$$

where the first inequality follows by Eq. (13) and the equations follow from the fact that h^* is an optimal solution for $\mathrm{LP}_{\mathrm{primal}}$. □

Using the above property, α can be replaced with $\alpha = \min\left\{\frac{\nu^*}{\sqrt{d}}, \min_{i\in[n]} \sqrt{\frac{md}{\xi_{\min}(\tilde{M}(\lambda_i))}}\right\}$ in the expected sample complexity given in Theorem 1.

Corollary 2. *Let ν^* be the optimal value for* $\mathrm{LP}_{\mathrm{primal}}$ *obtained in Algorithm 9. With probability at least $1-\delta$, the PolyALBA algorithm (Algorithm 4) will return the best super arm x^* with the expected sample complexity*

$$O\left(\frac{c_0 d(\alpha\sqrt{m}+\alpha^2)}{\Delta^2_{d+1}}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta^{-1}_{d+1}\right) + \sum_{i=2}^{\lfloor\frac{d}{2}\rfloor} \frac{c_0}{\Delta^2_i}(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta^{-1}_i)\right),$$

where $\alpha = \min\left\{\frac{\nu^*}{\sqrt{d}}, \min_{i\in[n]} \sqrt{\frac{md}{\xi_{\min}(\tilde{M}(\lambda_i))}}\right\}$.

Note that Algorithm 9 runs in polynomial time as long as $g_\lambda(x)$ is linear and not necessarily $g_\lambda(x) = \sum_{e\in[d]} w_{\lambda,e}x_e$ defined in this section.

LP-based algorithm for combinatorial robust optimization. We briefly explain polynomial-time solvability of $\mathrm{LP}_{\mathrm{primal}}$ in Algorithm 9 by the Ellipsoid method. Given a family $\mathcal{X}$ satisfying a combinatorial constraint and n-set function $g_{\lambda_1}, \dots, g_{\lambda_n} : 2^{[d]} \to \mathbb{R}_+$, we describe how to solve the following combinatorial robust optimization:

$$\min_{h\in\triangle([n])} \max_{x\in\mathcal{X}} \sum_{i\in[n]} h_i g_{\lambda_i}(x).$$

By von Neumann's minimax theorem, it holds that

$$\min_{h\in\triangle([n])} \max_{x\in\mathcal{X}} \sum_{i\in[n]} h_i g_{\lambda_i}(x) = \max_{p\in\triangle(\mathcal{X})} \min_{i\in[n]} \sum_{x\in\mathcal{X}} p_x g_{\lambda_i}(x).$$

A *polytope* of $\mathcal{X}$ is defined as $P(\mathcal{X}) = \mathrm{conv}\{x : x \in \mathcal{X}\}$. For a vector $x \in \mathcal{X}$ we define S^x to be the corresponding subset form, i.e. $S^x = \{i \in [n] : x_i = 1\}$. The key observation is that for any distribution $p \in \triangle(\mathcal{X})$, we can obtain a point $y \in P(\mathcal{X})$ by $y = p^\top \cdot x$, and thus for every $e \in [d]$, $y_e = \sum_{x\in\mathcal{X}: e\in S^x} p_x$. This means that y_e is the marginal probability of seeing an included dimension (a.k.a. a base arm e) when selecting vector x (a.k.a. super arm S^x) according to distribution p.

Then, the optimal value of the above problems is equal to the value of the following LP:

$$\begin{aligned}\mathrm{LP}_{\mathrm{dual}} : \max. \quad & s \\ \text{s.t.} \quad & s \leq \sum_{e\in[d]} w_{\lambda_i,e} y_e, \ (\forall i \in [n])\\ & y \in P(\mathcal{X}).\end{aligned} \tag{14}$$

If $\mathcal{X}$ is a matroid, matroid intersection, or the set of s-t paths, there exists an efficient *separation oracle* for $P(\mathcal{X})$. The separation problem for these constraints can be solved in polynomial time as long as $g_{\lambda_1}, \dots, g_{\lambda_n}$ are linear functions. Therefore, due to the theorem of Grötschel et al. (Grötschel, Lovász, and Schrijver 1981), we can solve the LP in polynomial time in d and n. Note that the Ellipsoid method can find an optimal solution to the dual problem of $\mathrm{LP}_{\mathrm{dual}}$, i.e., $\mathrm{LP}_{\mathrm{primal}}$ in (10). Therefore, we can obtain $h^* = \mathrm{argmin}_{h\in\triangle([n])} \max_{x\in\mathcal{X}} \sum_{i\in[n]} h_i g_{\lambda_i}(x)$. For the knapsack constraint and the $r(>2)$-matroid intersection constraint, the corresponding separation problems are NP-hard. Kawase and Sumita (2019) proposed approximation schemes by solving a separation problem for a relaxation of the polytope, which gives PTAS for the knapsack constraint and $2/(er)$-approximate solution for r-matroid intersection constraint.

G Equivalence Theorem for Optimal Experimental Design

We introduce the following equivalence theorem in Kiefer and Wolfowitz (1960) adopted our setting of CPE-BL, which will be used in our analysis.

Proposition 2 (Kiefer and Wolfowitz (1960))**.** *Define $M(\lambda) = \mathbb{E}_{z\sim\lambda}[zz^\top]$ for any distribution λ supported on $\mathcal{X} \subseteq \mathbb{R}^d$. We consider two extremum problems.*

The first is to choose λ so that

$(1)\lambda$ *maximizes* $\det M(\lambda)$ (*D-optimal design*)

The second one is to choose λ so that

$$(2)\lambda \text{ minimizes } \max_{x\in\mathcal{X}} x^\top M(\lambda)^{-1}x \qquad (\textit{G-optimal design})$$

We note that $\mathbb{E}_{x\sim\lambda}[x^\top M(\lambda)^{-1}x]$ is d, hence, $\max_{x\in\mathcal{X}} x^\top M(\lambda)^{-1}x \geq d$, and thus a sufficient condition for λ to satisfy (2) is

$$(3)\max_{x\in\mathcal{X}} x^\top M(\lambda)^{-1}x = d.$$

Statements (1), (2) and (3) are equivalent.

H Missing proofs

H.1 Proof of $\|\hat{\theta}_n - \theta\|_2 \leq \beta_\sigma$ in Section 4.1

Proof. We prove inequality $\|\hat{\theta}_n - \theta\|_2 \leq \beta_\sigma$ in Section 4.1 using similar techniques in (Lin et al. 2014).

Recall that in the GCB-PE algorithm (Algorithm 6), $\hat{\theta}_n$ is the estimate of the environment vector θ in the n-th exploration round. For any n,

$$\begin{aligned}
&\left\|\hat{\theta}_n - \theta\right\|_2 \\
=&\left\|M_\sigma^+\vec{y}_n - M_\sigma^+ M_\sigma\theta\right\|_2 \\
=&\left\|M_\sigma^+ \cdot [M_{x_1}\eta_1; \cdots ; M_{x_{|\sigma|}}\eta_{|\sigma|}]\right\|_2 \\
=&\left\|(M_\sigma^\top M_\sigma)^{-1}\sum_{i=1}^{|\sigma|} M_{x_i}^\top M_{x_i}\eta_i\right\|_2 \\
\leq& \max_{\eta_1,\cdots,\eta_{|\sigma|}\in[-1,1]^d}\left\|(M_\sigma^\top M_\sigma)^{-1}\sum_{i=1}^{|\sigma|} M_{x_i}^\top M_{x_i}\eta_i\right\|_2 \\
=&\beta_\sigma.
\end{aligned}$$

□

H.2 Proof of Theorem 2

In order to prove Theorem 2, we first present the following three lemmas, Lemma 7-9.

Lemma 7. *For Algorithm 6, after n exploration rounds,*

$$\Pr[\|\theta - \hat{\theta}(n)\|_2 \geq \text{rad}_n] \leq \frac{\delta}{2n^2}$$

Proof. In Lemma A.3 in (Lin et al. 2014), let $\gamma = \text{rad}_n$. Then, we have

$$\begin{aligned}
\Pr[\|\theta - \hat{\theta}(n)\|_2 \geq \text{rad}_n] \leq& 2e^2\exp\left\{-\frac{n}{2\beta_\sigma^2}\cdot\frac{2\beta_\sigma^2\log(\frac{4n^2e^2}{\delta})}{n}\right\} \\
=&\frac{\delta}{2n^2}.
\end{aligned}$$

□

Define the following events

$$\mathcal{E}_n := \{\forall x\in\mathcal{X}, |\bar{r}(x,\theta) - \bar{r}(x,\hat{\theta}(n))| < L_p\cdot\text{rad}_n\}, n\geq 1$$

$$\mathcal{E} := \bigcap_{n=1}^{\infty}\mathcal{E}_n.$$

Lemma 8. *It hols that $\Pr[\mathcal{E}] \geq 1-\delta$.*

Proof. From the continuity of the expected reward function,

$$\bar{r}(x,\theta) - \bar{r}(x,\hat{\theta}(n)) < L_p\cdot\|\theta - \hat{\theta}(n)\|_2.$$

From Lemma 7, we have that with probability at least $1-\frac{\delta}{2n^2}$,

$$\|\theta - \hat{\theta}(n)\|_2 < \text{rad}_n.$$

Thus, with probability at least $1-\frac{\delta}{2n^2}$,

$$\bar{r}(x,\theta) - \bar{r}(x,\hat{\theta}(n)) < L_p\cdot\text{rad}_n.$$

In other words,

$$\Pr[\mathcal{E}_n] \geq 1 - \frac{\delta}{2n^2}.$$

Thus, we have

$$\begin{aligned}
\Pr[\mathcal{E}] &= 1 - \Pr[\bar{\mathcal{E}}] \\
&\geq 1 - \sum_{n=1}^{\infty}\Pr[\bar{\mathcal{E}}_j] \\
&\geq 1 - \sum_{n=1}^{\infty}\frac{\delta}{2n^2} \\
&\geq 1-\delta.
\end{aligned}$$

□

Lemma 9. *Suppose that $\mathcal{E}$ occurs. If $\text{rad}_n < \frac{\Delta_{\min}}{4L_p}$, Algorithm 6 will terminate.*

Proof. Suppose that $\mathcal{E}$ occurs. From the definition of $\mathcal{E}$, we have

$$\begin{aligned}
\bar{r}(\hat{x},\hat{\theta}(n)) - \bar{r}(\hat{x}^-,\hat{\theta}(n)) &> \bar{r}(\hat{x},\theta) - \bar{r}(\hat{x}^-,\theta) - 2L_p\cdot\text{rad}_n \\
&= \Delta_{\min} - 2L_p\cdot\text{rad}_n \\
&> 2L_p\cdot\text{rad}_n
\end{aligned}$$

Thus, the stop condition holds, and then Algorithm 6 will terminate. □

Now we prove Theorem 2.

Proof. First, we prove the correctness of Algorithm 6. From the stop condition, we have that when Algorithm 6 terminates, for all $x\in\mathcal{X}\setminus\{\hat{x}\}$,

$$\bar{r}(\hat{x},\hat{\theta}(n)) - \bar{r}(x,\hat{\theta}(n)) > 2L_p\cdot\text{rad}_n.$$

Then, conditioning on $\mathcal{E}$, when Algorithm 6 terminates, for all $x\in\mathcal{X}\setminus\{\hat{x}\}$,

$$\begin{aligned}
\bar{r}(\hat{x},\theta) &> \bar{r}(\hat{x},\hat{\theta}(n)) - L_p\cdot\text{rad}_n \\
&> \bar{r}(x,\hat{\theta}(n)) + L_p\cdot\text{rad}_n \\
&> \bar{r}(x,\theta),
\end{aligned}$$

which complete the proof of correctness.

Next, we prove the sample complexity of Algorithm 6. Let N denote the total number of the exploration rounds. If $N=1$, Theorem 2 trivially holds. In the If $N>1$, from

Lemma 9, we have that after $N-1$ exploration rounds,

$$\sqrt{\frac{2\beta_\sigma^2\log(\frac{4(N-1)^2e^2}{\delta})}{N-1}} \geq \frac{\Delta_{\min}}{4L_p}$$

$$N \leq \frac{32\beta_\sigma^2 L_p^2}{\Delta_{\min}^2}\log\left(\frac{4N^2e^2}{\delta}\right)+1$$

Let $\tilde{H} := \frac{\beta_\sigma^2 L_p^2}{\Delta_{\min}^2}$. In the following, we prove $N \leq 655\tilde{H}\log(\frac{\tilde{H}}{\delta})$. We can write $N = C\tilde{H}\log(\frac{\tilde{H}}{\delta})$ for some $C>0$. In order to prove the theorem, it suffices to prove $C \leq 655$. Suppose, on the contrary, that $C>655$. Then, we have

$$\begin{aligned}
N \leq& 32\tilde{H}\log\left(\frac{4N^2e^2}{\delta}\right)+1\\
=&64\tilde{H}\log\left(\frac{2eC\tilde{H}\log\frac{\tilde{H}}{\delta}}{\delta}\right)+1\\
\leq&64\tilde{H}\log(2eC)+64\tilde{H}\log\frac{\tilde{H}}{\delta}+64\tilde{H}\log\left(\log\frac{\tilde{H}}{\delta}\right)\\
&+\tilde{H}\log\frac{\tilde{H}}{\delta}\\
\leq&64\tilde{H}\log(2eC)+129\tilde{H}\log\frac{\tilde{H}}{\delta}\\
<&C\tilde{H}\log\frac{\tilde{H}}{\delta}\\
=&N,
\end{aligned}$$

which makes a contradiction. Thus,

$$N \leq 655\tilde{H}\log(\frac{\tilde{H}}{\delta}).$$

Since an exploration round contains $|\sigma| \leq n$ actions, the total number of samples

$$T = |\sigma|\cdot N \leq \frac{655\beta_\sigma^2 L_p^2}{\Delta_{\min}^2}\log\left(\frac{\beta_\sigma^2 L_p^2}{\Delta_{\min}^2\delta}\right).$$

□

H.3 Proof of Lemma 1

Proof. Recall that in Algorithm 5, we choose d super arms $\mathcal{X}_\sigma = \{x_1,\dots,x_d\}$ from $\mathcal{X}$, such that $\mathrm{rank}(X)=d$ where $X=(x_1,\dots,x_d)$. Then, for any super arm $z \in \mathcal{X}$, z can be written as a linear combination of $x_1,x_2,\dots,x_d$, i.e.,

$$z = Xw,$$

where $w \in \mathbb{R}^d$ is the vector of coefficients. Let $\xi_{\min}(A)$ denote the smallest eigenvalue of matrix A. Then, we have

$$\begin{aligned}
\sum_{k=1}^d |w_k| \leq& \sqrt{d\left(\sum_{k=1}^d w_k^2\right)}\\
=&\sqrt{dw^\top w}\\
\leq&\sqrt{dw^\top X^\top Xw}\cdot\max_{w'\in\mathbb{R}^d}\sqrt{\frac{w'^\top w'}{w'^\top X^\top Xw'}}\\
\leq&\sqrt{dw^\top X^\top Xw}\cdot\sqrt{\frac{1}{\xi_{\min}(X^\top X)}}\\
=&\sqrt{\frac{d}{\xi_{\min}(X^\top X)}}\|z\|_2\\
\leq&\sqrt{\frac{md}{\xi_{\min}(X^\top X)}}\\
=&\sqrt{\frac{md}{\xi_{\min}(XX^\top)}}\\
=&\alpha.
\end{aligned}$$

Note that in Algorithm 5, we compute $\lambda^*_{\mathcal{X}_\sigma} = \mathrm{argmin}_{\lambda\in\triangle(\mathcal{X}_\sigma)}\max_{x\in\mathcal{X}_\sigma} x^\top M(\lambda)^{-1}x$ by the entropic mirror descent (Algorithm 8 in Appendix E). $\lambda^*_{\mathcal{X}_\sigma}$ is the solution to Proposition 2 and satisfies $\max_{x\in\mathcal{X}_\sigma} x^\top M(\lambda^*_{\mathcal{X}_\sigma})^{-1}x = d$. Thus, $\max_{x\in\mathcal{X}_\sigma}\|x\|_{M(\lambda^*_{\mathcal{X}_\sigma})^{-1}} = \sqrt{d}$. Thus, for any $z\in\mathcal{X}$ we have

$$\begin{aligned}
\|z\|_{M(\lambda^*_{\mathcal{X}_\sigma})^{-1}} =&\|w_1x_1+\cdots+w_dx_d\|_{M(\lambda^*_{\mathcal{X}_\sigma})^{-1}}\\
\leq&|w_1|\cdot\|x_1\|_{M(\lambda^*_{\mathcal{X}_\sigma})^{-1}}+\dots\\
&+|w_d|\cdot\|x_d\|_{M(\lambda^*_{\mathcal{X}_\sigma})^{-1}}\\
\leq&|w_1|\cdot\sqrt{d}+\cdots+|w_d|\cdot\sqrt{d}\\
=&(|w_1|+\cdots+|w_d|)\sqrt{d}\\
\leq&\alpha\sqrt{d}.
\end{aligned}$$

□

H.4 Proof of Theorem 1

Technical lemmas for Theorem 1

Lemma 10. *When $n \geq c_0\ell(\varepsilon)\ln(\frac{5|\mathcal{X}|}{\delta_r})$ where $\varepsilon \leq 3$, we have*

$$\Pr[|x^\top\theta - x^\top\hat{\theta}| \leq \varepsilon, \forall x\in\mathcal{X}] \geq 1-\delta.$$

Proof. We introduce the high probability bound for the estimator $\hat{\theta}$ as follows.

Proposition 3 (Lemma 10 Tao, Blanco, and Zhou (2018)). *Let $c_0 = \max\{4L^2,3\}$. Let $n \geq \ell\ln(5/8)$ where $\ell \geq d$. For any fixed $x\in\mathcal{X}$, with probability at least $1-\delta$, we have*

$$|x^\top(\theta-\hat{\theta})| \leq$$

$$\sqrt{\frac{2\|x\|_2^2+2\sqrt{d}\|x\|_2\|x\|_{M(\lambda)^{-1}}+(4+2\sqrt{d/\ell})\|x\|^2_{M(\lambda)^{-1}}}{\ell}}.$$

Since $\|x\|_2 \leq \sqrt{m}$ and $\|x\|_{M(\lambda)^{-1}} \leq \alpha\sqrt{d}$ from Lemma 1, applying Proposition 3 for every super arm in $\mathcal{X}$ and via a union bound, we have that when $n \geq c_0 \ell \ln(\frac{5|\mathcal{X}|}{\delta_r})$ where $\ell \geq d$,

$$\Pr\left[|x^\top\theta - x^\top\hat{\theta}| \leq \sqrt{\frac{2m + 2\alpha\sqrt{m}d + (4 + 2\sqrt{d/\ell})\alpha^2 d}{\ell}}, \forall x \in \mathcal{X}\right] \geq 1 - \delta.$$

Setting ℓ as $\ell(\varepsilon) := \frac{2m+2\alpha\sqrt{m}d+4\alpha^2 d+\alpha\varepsilon d}{\varepsilon^2}$, we have that with probability at least $1-\delta$, $\forall x \in \mathcal{X}$,

$$\begin{aligned}
|x^\top\theta - x^\top\hat{\theta}| \leq& \sqrt{\frac{2m + 2\alpha\sqrt{m}d + (4 + 2\sqrt{d/\ell})\alpha^2 d}{\ell}} \\
=& \left(\frac{2m + 2\alpha\sqrt{m}d + 4\alpha^2 d}{2m + 2\alpha\sqrt{m}d + 4\alpha^2 d + \alpha\varepsilon d}\varepsilon^2 \right. \\
&\left. + \frac{2\alpha^2 d^{\frac{3}{2}}}{(2m + 2\alpha\sqrt{m}d + 4\alpha^2 d + \alpha\varepsilon d)^{\frac{3}{2}}}\varepsilon^3\right)^{\frac{1}{2}} \\
\leq& \varepsilon\left(1 - \frac{\alpha\varepsilon d}{2m + 2\alpha\sqrt{m}d + 4\alpha^2 d + \alpha\varepsilon d}\right. \\
&+ \frac{2\alpha\sqrt{d}}{\sqrt{2m + 2\alpha\sqrt{m}d + 4\alpha^2 d + \alpha\varepsilon d}} \cdot \\
&\left.\frac{\alpha\varepsilon d}{(2m + 2\alpha\sqrt{m}d + 4\alpha^2 d + \alpha\varepsilon d)}\right)^{\frac{1}{2}} \\
\leq& \varepsilon,
\end{aligned}$$

which completes the proof. □

Next, we show the sample complexity bound for the epoch $q = 0$.

Lemma 11. *With probability at least $1-\delta_0$, the first epoch $q = 0$ in Algorithm 4 satisfies the following properties: (i) epoch $q = 0$ ends with $x^* \in S_1$; and (ii) the sample complexity is bounded by*

$$O\left(\frac{c_0(\alpha\sqrt{m}d + \alpha^2 d)}{\Delta_{d+1}^2}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta_{d+1}^{-1}\right)\right).$$

Proof. For round r of the first epoch, define event $\mathcal{F}_r := \{|x^\top\theta - x^\top\hat{\theta}_r| \leq \frac{\varepsilon}{2}, \forall x \in \mathcal{X}\}$. Applying Lemma 10, we have $\Pr[\mathcal{F}_r] \leq 1 - \delta_r$. Define event $\mathcal{F} := \bigcap_{r=1}^{\infty}\mathcal{F}_r$. By a union bound, we have $\Pr[\mathcal{F}] \geq 1 - \sum_{r=1}^{\infty}\delta_r = 1 - \sum_{r=1}^{\infty}\frac{6}{\pi^2}\frac{\delta_q}{r^2} \geq 1 - \delta_q$. We condition the remaining proof on event $\mathcal{F}$.

(i) First, we show that the first epoch $q = 0$ will end with $x^* \in S_1$. Let $x_1, x_2, \ldots$ denote the super arms ranked by $x^\top\theta$ for $x \in \mathcal{X}$ (i.e., $x_1^\top\theta \geq x_2^\top\theta, \ldots$), and we use x^* and x_1 interchangeably.

For any round $r \geq 1$, $x_1^\top\hat{\theta}_r \geq x_1^\top\theta - \frac{\varepsilon_r}{2} \geq \hat{x}_1^\top\theta - \frac{\varepsilon_r}{2} \geq \hat{x}_1^\top\hat{\theta}_r - \varepsilon_r$. Rearranging the terms, we have $\hat{x}_1^\top\hat{\theta}_r - x_1^\top\hat{\theta}_r \leq \varepsilon_r$, which implies that x_1 will never be discarded. Thus, when the first epoch $q = 0$ ends, $x_1 \in S_1$.

Let r^* be the smallest round such that $\varepsilon_{r^*} < \frac{\Delta_{d+1}}{2}$. In round r^*, for x_i s.t. $i \geq d+1$, $x_1^\top\hat{\theta}_{r^*} - x_i^\top\hat{\theta}_{r^*} \geq (x_1^\top\theta - \frac{\varepsilon_{r^*}}{2}) - (x_i^\top\theta + \frac{\varepsilon_{r^*}}{2}) \geq \Delta_i - \varepsilon_{r^*} \geq \Delta_{d+1} - \varepsilon_{r^*} > \varepsilon_{r^*}$. Then, the first epoch $q = 0$ will end.

(ii) Since the first epoch $q = 0$ will end in (or before) round r^*, which is the smallest round such that $\varepsilon_{r^*} < \frac{\Delta_{d+1}}{2}$, then the sample complexity of the first epoch $q = 0$ is bounded by

$$\begin{aligned}
&O\left(\frac{c_0(\alpha\sqrt{m}d + \alpha^2 d)}{\varepsilon_{r^*}^2}\ln\left(\frac{|\mathcal{X}|}{\delta_{r^*}}\right)\right) \\
=&O\left(\frac{c_0(\alpha\sqrt{m}d + \alpha^2 d)}{\Delta_{d+1}^2}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta_{d+1}^{-1}\right)\right).
\end{aligned}$$

□

Proof of Theorem 1

Proof. Define $q^* = \lfloor\log_2 d\rfloor$. For epoch $q = 0$, applying Lemma 11, the sample complexity is bounded by

$$O\left(\frac{c_0(\alpha\sqrt{m}d + \alpha^2 d)}{\Delta_{d+1}^2}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta_{d+1}^{-1}\right)\right).$$

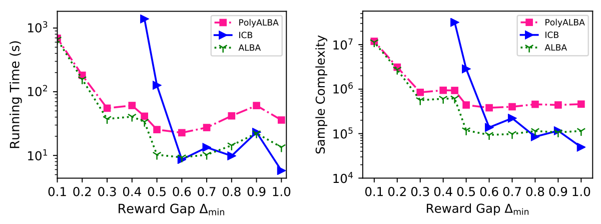
For epoch $q \geq 1$, the PolyALBA algorithm directly calls subroutine ALBA. Applying Lemma 17 in (Tao, Blanco, and Zhou 2018), we can bound the sample complexity for epoch $q \geq 1$ by

$$\begin{aligned}
&\sum_{q=1}^{q^*} O\left(\frac{c_0\left\lfloor\frac{d}{2^{q-1}}\right\rfloor}{\Delta^2_{\left\lfloor\frac{d}{2^q}\right\rfloor+1}}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta^{-1}_{\left\lfloor\frac{d}{2^q}\right\rfloor+1}\right)\right) \\
=&\sum_{q=1}^{q^*} O\left(\frac{c_0\left\lfloor\frac{d}{2^q}\right\rfloor}{\Delta^2_{\left\lfloor\frac{d}{2^q}\right\rfloor+1}}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta^{-1}_{\left\lfloor\frac{d}{2^q}\right\rfloor+1}\right)\right) \\
=&\sum_{q=1}^{q^*} O\left(\frac{c_0(\left\lfloor\frac{d}{2^q}\right\rfloor - \left\lfloor\frac{d}{2^{q+1}}\right\rfloor)}{\Delta^2_{\left\lfloor\frac{d}{2^q}\right\rfloor+1}}\cdot\right. \\
&\left.\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta^{-1}_{\left\lfloor\frac{d}{2^q}\right\rfloor+1}\right)\right) \\
=&\sum_{q=1}^{q^*-1}\sum_{\left\lfloor\frac{d}{2^{q+1}}\right\rfloor+1}^{\left\lfloor\frac{d}{2^q}\right\rfloor} O\left(\frac{c_0}{\Delta_i^2}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta_i^{-1}\right)\right) \\
=&O\left(\sum_{i=2}^{\left\lfloor\frac{d}{2}\right\rfloor}\frac{c_0}{\Delta_i^2}\left(\ln\delta^{-1} + \ln|\mathcal{X}| + \ln\ln\Delta_i^{-1}\right)\right).
\end{aligned}$$

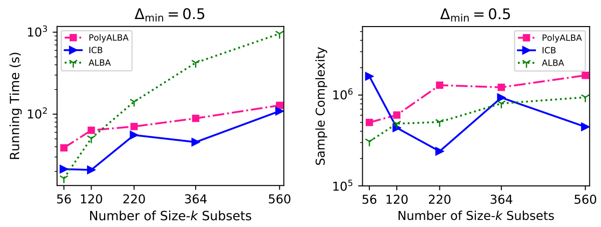
Summing the sample complexity for epoch $q = 0$ and $q \geq 1$, we obtain the theorem. □

I Experiments on the Top-k Instances

The main purpose of the experiments in this section is to see the dependence of the performance on the minimum gap $\Delta_{\min}$. We empirically demonstrate that PolyALBA is robust across different $\Delta_{\min}$ settings and runs much faster than existing BAI-LB algorithms as reported in Figure 3.



(a) The number of samples and running time with $(d, k) = (8, 3)$. We vary the minimum gap $\Delta_{\min}$ from 0.1 to 1.0. Each point is an average over 10 realizations.



(b) The number of samples and running time with $k = 3$, $d = 8, 10, 12, 14, 16$, and $\Delta_{\min} = 0.5$. Each point is an average over 10 realizations.

Figure 3: Experimental results of running time and sample complexity for full-bandit top-k instances.

Experimental settings. As a polynomial-time baseline algorithm, we implement ICB (Kuroki et al. 2020b), whose sample complexity heavily depends on $\Delta_{\min}^{-2}$. We evaluate PolyALBA, ALBA, and ICB for the top-k case of CPE-BL on Intel Xeon E5-2640 v3 CPU at 2.60GHz with 132GB RAM. We set the expected rewards for the top-k base arms uniformly at random from $[0.5, 1.0]$. Let $\theta_{\text{min-}k}$ be the minimum expected reward in the top-k base arms. We set the expected reward of the top $(k+1)$-th base arm to $\theta_{\text{min-}k} - \Delta_{\min}$ for the predetermined parameter $\Delta_{\min} \in [0.1, 1.0]$. Then, we generate the expected rewards of the rest of base arms by uniform samples from $[-1.0, \theta_{\text{min-}k} - \Delta_{\min}]$ so that expected rewards of the best super-arm is larger than those of the rest of super arms by at least $\Delta_{\min}$. We set the additive noise distribution $\mathcal{N}(0, 1)$. In all instances we set $\delta = 0.05$. In order to perform the exponential-time algorithms ALBA in reasonable time, we run the experiments with $d = 8$, $k = 3$, and thus $|\mathcal{X}| = 56$. Note that since RAGE is already prohibitive in these instances due to its memory error, we exclude it here. The result is shown in Figure 3(a). In the second experiment, we evaluate PolyALBA, ALBA, and ICB on the synthetic instances with varying $|\mathcal{X}|$ and fixed $\Delta_{\min}$. To perform ICB with a reasonable sample complexity, we set $\Delta_{\min} = 0.5$ in all instances. We vary the number of base arms $d = [8, 10, 12, 14, 16]$ while $k = 3$ is fixed. Thus, $|\mathcal{X}| \in [56, 120, 220, 364, 560]$ in this experiment. The result is shown in Figure 3(b).

Results. As can be seen in Figure 3(a), PolyALBA performs well in all instances, while a polynomial-time baseline ICB is sensitive to the value of $\Delta_{\min}$, which matches our theoretical analysis. Indeed, ICB cannot stop even after several days because it requires at least more than 10^9 samples when $\Delta_{\min} = 0.4$. On the other hand, PolyALBA is still competitive with the exponential-time baseline ALBA. Figure 3(b) demonstrates that ALBA is too slow in larger sized instances; we report that ALBA cannot work in the large instance with $d = 18$ and $|\mathcal{X}| = 816$ due to its memory issue in our environment. From the results, PolyALBA is the only practical algorithm that is efficient in terms of both time complexity and sample complexity. We validate that the sample complexity of PolyALBA is robust against the minimum gap, which well suits the real-world applications.