JMLR: Workshop and Conference Proceedings vol 65:1{55, 2017 30th Annual Conference on Learning Theory

Nearly Optimal Sampling Algorithms for Combinatorial Pure Exploration

Editor: Under Review for COLT 2017

Abstract

We study the combinatorial pure exploration problem Best-Set in a stochastic multi-armed bandit game. In an Best-Set instance, we are given n stochastic arms with unknown reward distributions, as well as a family $\mathcal{F}$ of feasible subsets over the arms. Let the weight of an arm be the mean of its reward distribution. Our goal is to identify the feasible subset in $\mathcal{F}$ with the maximum total weight, using as few samples as possible. The problem generalizes the classical best arm identi cation problem and the top-k arm identi cation problem, both of which have attracted signi cant attention in recent years. We provide a novel $instance\text{-}wise$ lower bound for the sample complexity of the problem, as well as a nontrivial sampling algorithm, matching the lower bound up to a factor of $\ln|\mathcal{F}|$. For an important class of combinatorial families (including spanning trees, matchings, and path constraints), we also provide polynomial time implementation of the sampling algorithm, using the equivalence of separation and optimization for convex program, and the notion of approximate Pareto curves in multi-objective optimization (note that $|\mathcal{F}|$ can be exponential in n). We also show that the $\ln|\mathcal{F}|$ factor is inevitable in general, through a nontrivial lower bound construction utilizing a combinatorial structure resembling the Nisan-Wigderson design. Our results signi cantly improve several previous results for several important combinatorial constraints, and provide a tighter understanding of the general Best-Set problem.

We further introduce an even more general problem, formulated in geometric terms. We are given n Gaussian arms with unknown means and unit variance. Consider the n-dimensional Euclidean space $\mathbb{R}^n$, and a collection $\mathcal{O}$ of disjoint subsets. Our goal is to determine the subset in $\mathcal{O}$ that contains the mean pro le (which is the n-dimensional vector of the means), using as few samples as possible. The problem generalizes most pure exploration bandit problems studied in the literature. We provide the rst nearly optimal sample complexity upper and lower bounds for the problem.

1. Introduction

The stochastic multi-armed bandit model is a classical model for characterizing the exploration-exploitation tradeo in a variety of application elds with stochastic environments. In this model, we are given a set of n stochastic arms, each associated with an unknown reward distribution. Upon each play of an arm, we can get a reward sampled from the corresponding distribution. The most well studied objectives include maximizing the cumulative sum of rewards, or minimizing the cumulative regret (see e.g., Cesa-Bianchi and Lugosi (2006); Bubeck and Cesa-Bianchi (2012)). Another popular objective is to identify the optimal solution (which can either be a single arm, or a set of arms, depending on the problem) with high con dence, using as few samples as possible. This problem is called the $pure\ exploration$ version of the multi-armed bandit problem, and has attracted signi cant attention

due to applications in domains like medical trials, crowdsourcing, communication network, databases and online advertising Chen et al. (2014); Zhou et al. (2014); Cao et al. (2015).

The problems of identifying the best arm (i.e., the arm with maximum expected reward) and the top-k arms have been studied extensively (see e.g., Mannor and Tsitsiklis (2004); Even-Dar et al. (2006); Audibert and Bubeck (2010); Kalyanakrishnan and Stone (2010); Gabillon et al. (2012); Kalyanakrishnan et al. (2012); Karnin et al. (2013); Jamieson et al. (2014); Zhou et al. (2014); Chen and Li (2015); Cao et al. (2015); Carpentier and Locatelli (2016); Garivier and Kaufmann (2016); Chen et al. (2016b); Chen et al. (2017)). Chen et al. (2014) proposed the following signi cant generalization, in which the cardinality constraint is replaced by a general combinatorial constraint and the goal is to identify the best subset (in terms of the total mean) satisfying the constraint.

De nition 1.1 (Best-Set)

In a Best-Set *instance* $\mathcal{C} = (S, \mathcal{F})$*, we are given a set* S *of* n *arms. Each arm* $a \in S$ *is associated with a Gaussian reward distribution with unit variance and an unknown mean* μ_a*. We are also given a family of subsets* $\mathcal{F}$ *with ground set identified with the set* S *of arms. Our goal is to find with probability at least* $1 - \delta$*, a subset in* $\mathcal{F}$ *with the maximum total mean using as few samples as possible. We assume that there is a unique subset with the maximum total mean.*

In the above de nition, the set family $\mathcal{F}$ may be given explicitly (i.e., the list of all subsets in $\mathcal{F}$ is given as input), or implicitly in the form of some combinatorial constraint (e.g., matroids, matchings, paths). Note that in the latter case, $|\mathcal{F}|$ may be exponential in the input size; we would additionally like to design sampling algorithms that run in polynomial time. Some common combinatorial constraints are the following:

1. (Matroids) $(S, \mathcal{F})$ is a matroid, where S is the ground set and $\mathcal{F}$ is the family of independent set of the matroid. The problem already captures a number of interesting applications. See Chen et al. (2016a) for more details and the state-of-the-art sample complexity bounds.

2. (Paths) Consider a network G, in which the latency of each edge is stochastic. However, the distributions of the latencies are unknown and we can only take samples. We would like to choose a path from node s to node t such that the expected latency is minimized. Here S is the set of edges of a given undirected graph G, and $\mathcal{F}$ the set of s-t paths in G.

3. (Matchings) There are n workers and n types of jobs. Each job type must be assigned to exactly one worker, and each worker can only handle one type of job. Jobs of the same type may not be exactly the same, hence may have di erent pro t. For simplicity, we assume that for worker i, the pro t of nishing a random job in type j follows an unknown distribution D_{ij}. Our goal is to nd an assignment of types to workers, such that the expected total reward is maximized (assuming each worker gets a random job from the type assigned to them). This problem has potential applications to crowdsourcing. Here, S is the set of edges in the worker-job-type bipartite graph G, and $\mathcal{F}$ the set of perfect matchings in G.

4. (Tree-Planning) We are given a tree, where each arm i corresponds to an edge of the tree. The family $\mathcal{F}$ is the set of paths from the root to the leaves. The goal is to find the root-leaf path with the maximum weight. This setting corresponds to the open-loop planning problem of maximizing the expected sum of rewards over consecutive actions from a starting state (i.e., the root) when the state dynamics is deterministic and the reward distributions are unknown (see e.g., Munos et al. (2014); Gabillon et al. (2016)).

While these examples show that the Best-Set problem is quite general, there are other interesting pure exploration bandit problems that cannot be captured by such combinatorial constraints over the arm set. For example, suppose there are n Gaussian arms with unknown means and unit variance. We know there is exactly one special arm with mean in the interval $(0.4, 0.6)$; all other arms have means either strictly larger than 0.6, or strictly less than 0.4. We would like to identify this special arm. To this end, we define a general sampling problem, which captures such problems, as follows.

Definition 1.2 (General-Samp) *An instance of the general sampling problem is a pair $\mathcal{I} = (S, \mathcal{O})$, where $S = (A_1, A_2, \ldots, A_n)$ is a sequence of n Gaussian arms each with unit variance. $\mathcal{O}$ is a collection of answer sets, each of which is a subset of $\mathbb{R}^n$. Let μ_i denote the mean of arm A_i. The vector μ is called the mean profile of the instance. In each round, we choose one of the arms and obtain an independent sample drawn from its reward distribution. The goal is to identify with probability $1-\delta$ the unique set in $\mathcal{O}$ that contains μ, while using as few samples as possible. It is guaranteed that $\mu \in \bigcup_{O\in\mathcal{O}} O$, and for each $O \in \mathcal{O}$, the closure of $\bigcup_{O'\in\mathcal{O}\setminus\{O\}} O'$ is disjoint from O.*

This definition of the general sampling problem captures well-studied bandit problems (having unique solutions) in the pure-exploration setting:

1. In the best arm identification problem, $\mathcal{O}$ contains exactly n answer sets, where the i-th answer set is given by $\{\mu \in \mathbb{R}^n : \mu_i > \max_{j\neq i} \mu_j\}$.

2. In the Best-Set problem (Definition 1.1), $\mathcal{O}$ contains exactly $|\mathcal{F}|$ answer sets, where each answer set corresponds to a set $S \in \mathcal{F}$, and is given by $\{\mu \in \mathbb{R}^n : \sum_{i\in S} \mu_i > \sum_{i\in T} \mu_i, \forall T \in \mathcal{F}, T \neq S\}$.

3. There are n Gaussian arms with unknown means and unit variance. We would like to find how many arms have mean larger than a given threshold θ. This is a variant of the threshold bandit problem (see e.g., Locatelli et al. (2016)). $\mathcal{O}$ contains exactly n answer sets, where the j-th answer set is given by $\{\mu \in \mathbb{R}^n : \sum_{i\in[n]} \mathbb{I}\{\mu_i > \theta\} = j\}$. We assume that no arm has mean exactly θ (to guarantee disjointness).

4. There are n Gaussian arms with unknown means and unit variance. Given a threshold θ, we want to determine whether the span (i.e., the difference between the largest and smallest means) is greater than θ. We assume that no difference of two arms is exactly θ (to guarantee disjointness).

Remark 1.3 *The disjointness requirement, that the closure of $\bigcup_{O'\in\mathcal{O}} O'$ is disjoint from O for all $O \in \mathcal{O}$, is crucial to the solvability of the instance. For example, no δ-correct algorithm (for some $\delta < 0.5$) can solve the instance with a single arm with zero mean and*

$\mathcal{O} = \{(-\infty, 0), [0, +\infty)\}$ *within a finite number of samples in expectation. Furthermore, the disjointness condition guarantees that the correct solution is unique. Hence, our problem cannot capture some PAC problems in which there may be many approximate solutions.*

Remark 1.4 *Our problem is closely related to the active multiple hypothesis testing problem. See the related work section for more discussions.*

1.1. Our Results

In order to formally state our results, we first define the notion of δ-correct algorithms.

Definition 1.5 (δ-correct algorithms) *We say Algorithm $\mathbb{A}$ is a δ-correct algorithm for* Best-Set *if on every instance $\mathcal{C} = (S, \mathcal{F})$, algorithm $\mathbb{A}$ identifies the set in $\mathcal{F}$ with the largest total mean with probability at least $1 - \delta$.*

Similarly, we say Algorithm $\mathbb{A}$ is a δ-correct algorithm for General-Samp *if on every instance $\mathcal{I} = (S, \mathcal{O})$, algorithm $\mathbb{A}$ identifies the set in $\mathcal{O}$ which contains the mean profile of the instance with probability at least $1 - \delta$.*

1.1.1. Instance Lower Bound via Convex Program

Garivier and Kaufmann (2016) obtained a strong lower bound for the sample complexity of Best-1-Arm, based on the change of distribution. Their lower bound is in fact the solution of a mathematical program. They show that for Best-1-Arm, one can derive the explicit solution for several distributions.

Garivier and Kaufmann's approach is general and can be applied to Best-Set and General-Samp as well. However, the resulting mathematical program is not easy to work with. Unlike Best-1-Arm, we cannot hope for an explicit solution for the general Best-Set problem: the program has an infinite number of constraints, and it is unclear how to solve it computationally. Instead, we adopt their framework and derive an equivalent convex program for Best-Set (in Section 3.1). Using this, we obtain the following result.

Theorem 1.6 *Let $\mathcal{C} = (S, \mathcal{F})$ be an instance of* Best-Set*. Let $\mathit{Low}(\mathcal{C})$ be the optimal value of the convex program (1) (see Section 3.1). Then for any $\delta \in (0, 0.1)$ and δ-correct algorithm $\mathbb{A}$ for* Best-Set*, $\mathbb{A}$ takes $(\mathit{Low}(\mathcal{C}) \ln \delta^{-1})$ samples in expectation on $\mathcal{C}$.*

Our new lower bound has the following computational advantage. First, it is a solution of a convex program with a finite number of constraints. Hence, one can solve it in time polynomial in n and the number of constraints (note that there may be exponential many of them, if $|\mathcal{F}|$ is exponentially large). Moreover, for some important classes of $\mathcal{F}$, we can approximate the optimal value of the convex program within constant factors in polynomial time (see Section 5.4 for more details).

Comparison with the Lower Bound in Chen et al. (2014). Let $\mathcal{C} = (S, \mathcal{F})$ be an instance of Best-Set with the optimal set $O \in \mathcal{F}$. Assume that all arms are Gaussian with unit variance. It was proved in Chen et al. (2014) that for any $\delta \in (0, e^{-16}/4)$ and any δ-correct algorithm $\mathbb{A}$, $\mathbb{A}$ takes $(H_C(\mathcal{C}) \ln \delta^{-1})$ samples in expectation. Here $H_C(\mathcal{C}) = \sum_{i \in S} \Delta_i^{-2}$ is the *hardness* of the instance $\mathcal{C}$, where Δ_i, the *gap* of arm $i \in S$, is defined as

$$\Delta_i = \begin{cases} \mu(O) - \max_{O' \in \mathcal{F}, i \notin O'} \mu(O'), & i \in O, \\ \mu(O) - \max_{O' \in \mathcal{F}, i \in O'} \mu(O'), & i \notin O. \end{cases}$$

We can show that our lower bound is no weaker than the lower bound in Chen et al. (2014).

Theorem 1.7 *Let* $\mathcal{C} = (S, \mathcal{F})$ *be an instance of* Best-Set*,*

$$\mathit{Low}(\mathcal{C}) \geq H_C(\mathcal{C}).$$

The proof of Theorem 1.7 can be found in Section 3.2.

Furthermore, we note that for certain instances of Matchings and Paths, our lower bound can be stronger than the $H_C(\mathcal{C}) \ln \delta^{-1}$ bound by an (n) factor. We consider the following simple instance $\mathcal{C}_{\text{disj-sets}}$ of Best-Set that consist of $n = 2k$ arms numbered 1 through n. The only two feasible sets are $A = \{1, 2, \ldots, k\}$ and $B = \{k + 1, k + 2, \ldots, 2k\}$ (i.e., $\mathcal{F} = \{A, B\}$). The mean of each arm in A is $\epsilon > 0$, while each arm in B has a mean of 0. A simple calculation shows that $\mathrm{Low}(\mathcal{C}_{\text{disj-sets}}) = (\epsilon^{-2})$, while $H_C(\mathcal{C}_{\text{disj-sets}}) = O\left(\epsilon^{-2}/n\right)$. This establishes an (n) factor separation.

Indeed, $\mathcal{C}_{\text{disj-sets}}$ is a special case of many Best-Set instances, including Matchings (consider a cycle with length $2k$; there are two perfect matchings that are disjoint) and Paths (consider a graph with only two s-t paths, each with length k). Thus, understanding the complexity of $\mathcal{C}_{\text{disj-sets}}$ is crucial for understanding more complicated instances.

1.1.2. Positive Result I: A Nearly Optimal Algorithm for Best-Set

Our first positive result is a nearly optimal algorithm for Best-Set.

Theorem 1.8 *There is a* δ*-correct algorithm for* Best-Set *that takes*

$$O\left(\mathit{Low}(\mathcal{C}) \ln \delta^{-1} + \mathit{Low}(\mathcal{C}) \ln \ ^{-1}\left(\ln \ln \ ^{-1} + \ln |\mathcal{F}|\right)\right)$$

samples on an instance $\mathcal{C} = (S, \mathcal{F})$ *in expectation, where* $\ $ *denote the gap between the set with the second largest total mean in* $\mathcal{F}$ *and the optimal set* O*.*

Comparison with Previous Algorithms. Again, consider the instance $\mathcal{C}_{\text{disj-sets}}$, which consists of k arms with mean $\epsilon > 0$ and another k arms with mean zero. A straightforward strategy to determine whether A or B has a larger total mean is to sample each arm $\tau = 8/(k\epsilon^2) \ln(2/\delta)$ times, and determine the answer based on the sign of $\hat{\mu}(A) - \hat{\mu}(B)$. Lemma 2.1 implies that with probability at least $1 - \delta$, $\hat{\mu}(A) - \hat{\mu}(B)$ lies within an additive error $k\epsilon/2$ to $\mu(A) - \mu(B) = k\epsilon$. Hence, we can identify A as the correct answer using $n\tau = O(\epsilon^{-2} \ln \delta^{-1})$ samples.

Chen et al. (2014) developed a δ-correct algorithm CLUCB for Best-Set with sample complexity

$$O\left(\mathrm{width}(\mathcal{C})^2 H_C(\mathcal{C}) \ln(nH_C(\mathcal{C})/\delta)\right),$$

where $\mathrm{width}(\mathcal{C})$ is the *width* of the underlying combinatorial structure $\mathcal{F}$ as defined in Chen et al. (2014). Hence, roughly speaking, ignoring logarithmic factors, their upper bound is a $\mathrm{width}(\mathcal{C})^2$-factor (which is at most n^2 factor) away from the complexity term $H_C(\mathcal{C})$ they define,[1] while our upper bound is at most $\ln |\mathcal{F}|$ (which is at most n) factor away from our lower bound. Theorem 1.9 shows that the $\ln |\mathcal{F}|$ term is also inevitable in the worst case.

1. In view of Theorem 1.7 and Theorem 1.6, $H_C(\mathcal{C}) \ln(1/\delta)$ is indeed a lower bound.

In fact, consider the simple instance $\mathcal{C}_{\text{disj-sets}}$ we de ned earlier. A simple calculation shows that width($\mathcal{C}_{\text{disj-sets}}$) = (n) and $H_C(\mathcal{C}_{\text{disj-sets}}) = (\epsilon^{-2}/n)$, so CLUCB requires $(n\epsilon^{-2} \ln \delta^{-1})$ samples in total on the simple instance $\mathcal{C}_{\text{disj-sets}}$ we de ned earlier. Moreover, a recent algorithm proposed in Gabillon et al. (2016) also takes $(n\epsilon^{-2} \ln \delta^{-1})$ samples. In comparison, our algorithm achieves a sample complexity of $O(\epsilon^{-2}(\ln \delta^{-1} + \ln \epsilon^{-1} \ln \ln \epsilon^{-1}))$ on $\mathcal{C}_{\text{disj-sets}}$, which is nearly optimal. Therefore, our algorithm obtain a signi cant speed up of order (n) on certain cases comparing to all previous algorithms.

In fact, both these previous algorithms for Best-Set are UCB-based, i.e., they maintain a *confidence bound* for each *individual arm*. We observe that such UCB-based algorithms are inherently inadequate to achieve the optimal sample complexity, even for the simple instance $\mathcal{C}_{\text{disj-sets}}$. Note that in order for a UCB-based algorithm to identify A as the correct answer for $\mathcal{C}_{\text{disj-sets}}$, it requires an $O(\epsilon)$ estimation of the mean of each arm in S, which requires $(n\epsilon^{-2} \ln \delta^{-1})$ samples in total. Thus, the sample complexity of previous algorithms based on maintaining con dence bounds for individual arms is at least a factor of n away from the optimal sample complexity, even for very simple instances such as $\mathcal{C}_{\text{disj-sets}}$.

The $\ln|\mathcal{F}|$ term is necessary in the worst case: Note that the sample complexity of our algorithm involves a $\ln|\mathcal{F}|$ term, which could be large when $\mathcal{F}$ is exponential in n. Hence, it is natural to ask whether one can get rid of it. We show that this is impossible by proving a worst-case lower bound for Best-Set in which the factor $\ln|\mathcal{F}|$ is necessary.

Theorem 1.9 *(i) For $\delta \in (0, 0.1)$, two positive integers n and $m \le 2^{cn}$ where c is a universal constant,, and every δ-correct algorithm $\mathbb{A}$ for* Best-Set*, there exists an infinite sequence of n-arm instances $\mathcal{C}_1 = (S_1, \mathcal{F}_1), \mathcal{C}_2 = (S_2, \mathcal{F}_2), \ldots$, such that $\mathbb{A}$ takes at least*

$$(\mathit{Low}(\mathcal{C}_k) \cdot (\ln \delta^{-1} + \ln|\mathcal{F}_k|))$$

samples in expectation on each $\mathcal{C}_k$, $|\mathcal{F}_k| = m$ for all k, and $\mathit{Low}(\mathcal{C}_k)$ approaches to infinity as $k \to +\infty$.

(ii) Moreover, for each $\mathcal{C}_k$ defined above, there exists a δ-correct algorithm $\mathbb{A}_k$ for Best-Set *such that $\mathbb{A}_k$ takes*

$$O(\mathit{Low}(\mathcal{C}_k) \cdot \text{poly}(\log n, \ln \delta^{-1}))$$

samples in expectation on it. (The constants in and O do not depend on n, m, δ and k.)

The second part of the above theorem implies that $\text{Low}(\mathcal{C}_k) \cdot \ln \delta^{-1}$ is achievable by some speci c δ-correct algorithms for Best-Set (up to polylog factors). However, the rst part states that no matter what algorithm to use, one has to pay such a $\ln|\mathcal{F}|$ factor, which can be as large as n, for in nitely many instances. Therefore, Theorem 1.9 indeed captures a huge separation between the instance-wise lower bound and the worst-case lower bound: they may di er by a large factor of $\ln|\mathcal{F}|$.

Remark 1.10 *Such a separation is a delicate issue in pure exploration multi-armed bandit problems. Even in the* Best-1-Arm *problem with only two arms, such a separation of $\ln \ln \ ^{-1}$ (is the gap of the two arms) factor is known (see Chen and Li (2015) for more details).*

We note that Garivier and Kaufmann (2016) obtained an algorithm for Best-1-Arm that matches the instance lower bound as δ approaches zero. Such algorithms are called *asymptotically optimal* in the sequential hypothesis testing literature. Their algorithm can potentially be adapted to obtain asymptotically optimal algorithms for Best-Set (and even General-Samp) as well. From both Theorem 1.8 and 1.9, we can see the sample complexity typically consists of two terms, one depending on $\ln 1/\delta$ and one independent of δ (this is true for many other pure exploration bandit problems). In Garivier and Kaufmann (2016), the authors did not investigate the the second term in the sample complexity (which does not depend on δ), since it is treated as a \constant" (only δ is treated as a variable and approaches to 0). However, as Theorem 1.9 indicates, the \constant" term can be quite large (comparing with the rst term for moderate δ values, say $\delta = 10^{-10}$) and cannot be ignored, especially in our general Best-Set problem. Hence, in this paper, we explictly pin down the \constant" terms for our algorithms, and make progress towards tighter bounds on these terms (Theorems 1.9 and 1.14).

1.1.3. Positive Result II: An Efficient Implementation for Implicit $\mathcal{F}$

One drawback for our rst algorithm is that it needs to take full description of $\mathcal{F}$. So it would become computationally intractable if $\mathcal{F}$ is given *implicitly*, and of exponential size. Our second algorithm addresses this computational problem in several important special cases, assuming that the underlying combinatorial structure, $\mathcal{F}$, admits an *efficient maximization oracle* and a *pseudo-polynomial algorithm for the exact version*, which we explain next.

We say that a family of Best-Set instances $\{\mathcal{C}_k\} = \{(S_k, \mathcal{F}_k)\}$ has an e cient maximization oracle, if there is an algorithm that, given a weight function w de ned on S_k, identi es the maximum weight set in $\mathcal{F}_k$ (i.e., $\operatorname{argmax}_{A\in\mathcal{F}_k} \sum_{i\in A} w_i$), and runs in polynomial time (with respect to $|S_k|$). Moreover, a family of Best-Set instances $\{\mathcal{C}_k\}$ admits a pseudo-polynomial algorithm for the exact version, if an algorithm, given an integer weight function w on S_k together with a target value V, decides whether $\mathcal{F}_k$ contains a set with total weight *exactly* V, and runs in pseudo-polynomial time (i.e., polynomial with respect to $|S_k|$ and V).

Remark 1.11 *The following problems admit efficient maximization oracles and pseudo-polynomial algorithms for the exact version:*

1. *Maximum weight spanning tree.*
2. *Maximum weight bipartite matching.*
3. *Shortest s-t path with non-negative weights.* [2]

Theorem 1.12 *For any family of* Best-Set *instances that admits efficient maximization oracles and pseudo-polynomial algorithms, there is a δ-correct algorithm for this instance family that takes*

$$O\left(\mathit{Low}(\mathcal{C}) \ln \delta^{-1} + \mathit{Low}(\mathcal{C}) \ln^2 \quad ^{-1} \left(\ln \ln \quad ^{-1} + \ln |\mathcal{F}|\right)\right)$$

samples on an instance $\mathcal{C} = (S, \mathcal{F})$ in expectation, where denote the gap between the set with the second largest total mean in $\mathcal{F}$ and the optimal set O. Moreover, the algorithm runs in polynomial time with respect to the expected sample complexity.

2. For the shortest path problem, we need an efficient minimization oracle, which is equivalent.

1.1.4. Positive Result III: Nearly Optimal Algorithm for General-Samp

Last but not the least, we present an nearly optimal algorithm for General-Samp.

Theorem 1.13 *There is a δ-correct algorithm for* General-Samp *that takes*

$$O(\mathit{Low}(\mathcal{I})\ (\ln \delta^{-1} + n^3 + n \ln\ \ ^{-1}))$$

samples on any instance $\mathcal{I} = (S, \mathcal{O})$ in expectation, where

$$= \inf_{\nu \in \mathsf{Alt}(O)} \|\mu - \nu\|_2$$

is defined as the minimum Euclidean distance between the mean profile μ and an alternative mean profile $\nu \in \mathsf{Alt}(O)$ with an answer other than O.

Another worst-case lower bound with a factor of n. Note that the sample complexity of our algorithm above involves a term depending on n. Note that Best-Set is a special case of General-Samp, and setting $m = 2^{(n)}$ in Theorem 1.9, we obtain an $O(\mathsf{Low}(\mathcal{I}) \cdot (n + \ln \delta^{-1}))$ worst-case lower bound for General-Samp. [3] Therefore, at least we cannot get rid of the dependence on n. Moreover, the instance behind the lower bound in Theorem 1.9 has an exponentially large $|\mathcal{O}|$. So one may wonder if it is possible to reduce the factor n to $\ln |\mathcal{O}|$. [4] We present another lower bound showing it is also impossible, by constructing hard instances with $|\mathcal{O}| = O(1)$. Our lower bound instances reveal another source of hardness, di erent from the combinatorics used in Theorem 1.9.

Theorem 1.14 *For $\delta \in (0, 0.1)$, a positive integer n and every δ-correct algorithm $\mathbb{A}$ for the general sampling problem, there exists an infinite sequence of n-arm instances $\mathcal{I}_1 = (S_1, \mathcal{O}_1), \mathcal{I}_2 = (S_2, \mathcal{O}_2), \ldots$, such that $\mathbb{A}$ takes at least*

$$(\mathit{Low}(\mathcal{I}_k) \cdot (\ln \delta^{-1} + n))$$

samples in expectation on each $\mathcal{I}_k$, $|\mathcal{O}_k| = O(1)$ for all k, and $\mathit{Low}(\mathcal{I}_k)$ goes to infinity. Moreover, for each $\mathcal{I}_k$, there exists a δ-correct algorithm $\mathbb{A}_k$ for General-Samp *such that $\mathbb{A}_k$ takes*

$$O(\mathit{Low}(\mathcal{I}_k) \cdot \ln \delta^{-1})$$

samples in expectation on it. (The constants in and O do not depend on n, m, δ and k.)

1.2. Our Techniques

1.2.1. Overview of Our Best-Set Algorithm

Our algorithm is based on a process of successive elimination. However, unlike previous approaches Karnin et al. (2013); Chen and Li (2015); Chen et al. (2016a); Gabillon et al. (2016) which maintained a set of arms, our algorithm maintains a collection of candidate sets and performs the eliminations on them. The goal of the r-th round is to eliminate those sets with a optimality gap[5] of at least (2^{-r}).

3. We observe that $\mathsf{Low}(\mathcal{I})$ is essentially equivalent to $\mathsf{Low}(\mathcal{C})$ when $\mathcal{I}$ is identical to a Best-Set instance $\mathcal{C}$.
4. This is possible for Best-Set, because of Theorem 1.8.
5. The optimality gap for a set $A \in \mathcal{F}$ is simply $\mu(O) - \mu(A)$ where O is the optimal set.

In order to implement the above elimination, we adopt a mathematical program, which is a nontrivial modi cation of the one in Theorem 1.6 and sample the arms accordingly to obtain an 2^{-r} approximation of the gap between *every pair* of sets that are still under consideration. If there is a set pair (A, B) such that the empirical mean of B exceeds that of A by 2^{-r}, we are certain that A is not the optimal set, and thus we stop considering A as a candidate answer. This process is repeated until only one set remains.

1.2.2. Overview of Our Best-Set Algorithm with Efficient Computation

The previous algorithm maintains the sets still under consideration at the beginning of each round r. As the number of feasible sets is typically exponential in the number of arms, it may be computationally expensive to maintain these sets explicitly. The key to computational e ciency is to nd a compact representation of the sets still under consideration. Here, we represent these sets by using the empirical means and some carefully chosen threshold.

To e ciently solve the mathematical program (which is actually a *convex program*) mentioned above, we apply the Ellipsoid method and use the ε-approximate Pareto curve framework of Papadimitriou and Yannakakis (2000) to design an e cient separation oracle. This technique allows us to approximately solve the convex program in polynomial-time with respect to the input size and the sample complexity of our algorithm.

1.2.3. Overview of Our General-Samp Algorithm

Our General-Samp algorithm follows a \explore-verify" approach. In the rst stage of algorithm (exploration stage), we sample each arm repeatedly in round-robin fashion, until the con dence region of the mean pro le μ intersects exactly one answer set, which we identify as the candidate answer. The second stage (veri cation stage) is devoted to verifying the candidate answer. To this end, we formulate the optimal sampling pro le as a linear program. Then, we verify the candidate answer by sampling the arms according to the sampling pro le.

Note that in the exploration stage, the arms are sampled in an ine cient round-robin fashion, while the candidate answer is verifed using the optimal sampling pro le in the second stage. Hence, we use a less stringent con dence in Stage 1, and then adopts the required con dence level δ in the second stage.

1.2.4. Other Technical Highlights

The factor $\ln|\mathcal{F}|$ is necessary for Best-Set. In order to establish the worst-case $\mathrm{Low}(\mathcal{C}_k) \cdot \ln|\mathcal{F}|$ lower bound (the rst part of Theorem 1.9), we construct a family $\mathcal{F}$ of subsets of $[n]$ satisfying the following two important properties [6]:

- (Sets in $\mathcal{F}$ are large) Each subset $A \in \mathcal{F}$ is of the same size $\ell =$ (n).
- (Intersections are small) For every two di erent subsets $A, B \in \mathcal{F}$, $|A \cap B| \leq \ell/2$.

For each $A \in \mathcal{F}$, we construct an instance $\mathcal{C}_A$ in which the i-th arm has mean if $i \in A$ and mean 0 otherwise, where is a small real number. Clearly, a δ-correct algorithm must

6. These two properties resemble the well-known set system of Nisan and Wigderson (1994).

output the subset A on instance $\mathcal{C}_A$ with probability at least $1-\delta$. Intuitively speaking, our lower bound works by showing that if an algorithm $\mathbb{A}$ can correctly solve all $\mathcal{C}_A$'s, then there must exist two di erent instances $\mathcal{C}_A$ and $\mathcal{C}_B$, such that $\mathbb{A}$ can distinguish these two instance with a much smaller con dence of $\delta/|\mathcal{F}|$. It is easy to see the later task requires ($^{-2}\cdot\ln|\mathcal{F}|$) samples by the change of distribution method. Then, by a simple calculation, one can show that Low($\mathcal{C}_A$) = ($^{-2}$) for all $A\in\mathcal{F}$.

An $O(\textbf{Low}(\mathcal{C}_B)\cdot\text{polylog}(\log n,\ln\delta^{-1}))$ Upper Bound. To prove Theorem 1.9 (ii), for each instance of the form $\mathcal{C}_B$ constructed above, we need to design a δ-correct algorithm $\mathbb{A}_B$ which is particularly fast on that instance. To this end, we provide a surprisingly fast testing algorithm $\mathbb{A}_{\text{help}}$ to distinguish between two hypotheses $x=0$ and $\|x\|_2\geq 1$ with sample complexity $O(n)$ (see Theorem C.2 for the details). This is somewhat surprising, considering the following argument: for this problem, uniform sampling seems to be a good method as the problem is completely symmetric (no arm is more special than others). If we sample every arm once, we actually get a sample (which is an n-dimensional vector) from a multivariate Gaussian $N(x,I_{n\times n})$. To decide whether the mean x satis es $x=0$ or $\|x\|_2\geq 1$ with con dence 0.99, we need $O(n)$ samples from multi-variate Gaussian $N(x,I_{n\times n})$, by a simple calculation. This argument suggests that we need $O(n^2)$ arm pulls. However, we show that an interesting randomized sampling method only needs $O(n)$ arm pulls.

Worst Case Lower Bound for General-Samp. We consider the following special case of General-Samp, which behaves like an OR function: namely, each arm has mean either 0 or , and the goal is to nd out whether there is an arm with mean , where is a small real number.

Let $\mathcal{I}_k$ be an instance in which the k-th arm has mean , while other arms have mean 0. On one hand, it is not hard to see that Low($\mathcal{I}_k$) = ($^{-2}$) via a simple calculation. On the other hand, we show that in order to solve all $\mathcal{I}_k$'s correctly, an algorithm must in a sense nd the -mean arm itself, and hence are forced to spend (n $^{-2}$) total samples in the worst-case. While the high level idea is simple, the real proof is a bit technical and can be found in Section D.

1.3. Other Related Work

An important and well-studied special case of Best-Set and General-Samp is Best-1-Arm, in which we would like identify the best single arm. For the PAC version of the problem, [7] Even-Dar et al. (2002) obtained an algorithm with sample complexity $O(n\varepsilon^{-2}\cdot\ln\delta^{-1})$, which is also optimal in worse cases. For the exact version of Best-1-Arm, a lower bound of ($\sum_{i=2}^{n}$ $_{[i]}^{-2}\ln\delta^{-1}$), where $_{[i]}$ denotes the gap between the i-th largest arm and the largest arm, has been proved by Mannor and Tsitsiklis (2004). In a very early work, Farrell (1964) established a worst-case lower bound of ($_{[2]}^{-2}\ln\ln$ $_{[2]}^{-1}$) even if there are only two arms. An upper bound of $O(\sum_{i=2}^{n}$ $_{[i]}^{-2}(\ln\ln$ $_{[i]}^{-1}+\ln\delta^{-1}))$ was achieved by the Exponential-Gap Elimination algorithm by Karnin et al. (2013), matching Farrell's lower bound for two arms. Later, Jamieson et al. (2014) obtained the a more practical algorithm with the same theoretical sample complexity, based the con dence bounds derived from the law of iterative logarithm. Very Recently, in Chen et al. (2016b); Chen and Li (2016), the

7. In the PAC version, our goal is to identify an ε-approximate optimal arm.

authors proposed an intriguing gap-entropy conjecture stating that the optimal instance-wise sample complexity of Best-1-Arm is related to the entropy of a certain distribution of the arm gaps. On a different line, Garivier and Kaufmann (2016) proposed an algorithm which is asymptotically optimal. The high level ideas of our algorithms for Best-Set and General-Samp are similar to the approach in Garivier and Kaufmann (2016), which also computes the allocation of samples using a mathematical program, and then take samples according to it. In a high level, our algorithm are also similar to the "explore-verify" approach used in Karnin (2016).

The natural generalization of Best-1-Arm is Best-k-Arm, in which we would like to identify the top k arms. The problem and its PAC versions have been also studied extensively in the past few years Kalyanakrishnan and Stone (2010); Gabillon et al. (2012, 2011); Kalyanakrishnan et al. (2012); Bubeck et al. (2012); Kaufmann and Kalyanakrishnan (2013); Zhou et al. (2014); Kaufmann et al. (2014); Chen et al. (2016a); Chen et al. (2017).

All aforementioned results are in the *fixed confidence* setting, where we need to output the correct answer with probability at least $1 - \delta$, where δ is the given confidence level. Another popular setting is called the *fixed budget* setting, in which one aims to minimize the failure probability, subject to a fixed budget constraint on the total number of samples. (see e.g., Gabillon et al. (2012); Karnin et al. (2013); Chen et al. (2014); Carpentier and Locatelli (2016)).

Our problems are related to the classic sequential hypothesis testing framework, which is pioneered by Wald (1945). In fact, they are closely related to the *active hypothesis testing* problem first studied by Chernoff (1959). In Wald's setting, the observations are predetermined, and we only need to design the stopping time. In Chernoff's setting, the decision maker can choose different experiments to conduct, which result in different observations about the underlying model. Chernoff focused on the case of binary hypotheses and obtained an asymptotically optimal testing algorithm as the error probability approaches to 0. Chernoff's seminal result has been extended to more than two hypothesis (see e.g., Draglia et al. (1999); Naghshvar et al. (2013)). In fact, the active multiple hypothesis testing is already quite general, and includes the bandit model as a special case. Our work differs from the above line of work in the following aspects: First, most of the work following Chernoff's approach (which extends Wald's approach) uses different variants of the SPRT (sequential probability ratio test). It is unclear how to compute such ratios (efficiently) for our combinatorial pure exploration problem. However, the computation problem is a major focus of our work (we devote Section 5 to discuss how to solve the computation problem efficiently). Second, the optimality of their results are in the asymptotic sense (when $\delta \to 0$). In the high dimension (i.e., n is large) but moderate δ regime, the additive term, which is independent of δ but dependent on n (see Theorem 1.9 and 1.14), may become the dominate term. Our work makes this term explicit and aims at minimizing it as well.

2. Preliminaries

Some naming conventions first. Typically, we use a lowercase letter to denote an *element*, e.g., an arm a or an index i, uppercase letter to denote a set of elements, e.g., a set of arms S, and a letter in calligraphic font to denote a family of sets, e.g., a set of sets of arms $\mathcal{S}$. Given a Best-Set instance $\mathcal{C} = (S, \mathcal{F})$, μ_i denotes the mean of arm $i \in S$. For subset $A \subseteq S$, $\mu(A)$ denotes $\sum_{i \in A} \mu_i$.

For two sets A, B, we use $A \triangle B$ to denote the *symmetry difference* of A and B. Namely,

$$A \triangle B = (A \backslash B) \cup (B \backslash A).$$

We need the following important lemma for calculating the con dence level of a subset of arms.

Lemma 2.1 *Given a set of Gaussian arms $a_1, a_2, \ldots, a_k$ with unit variance and means $\mu_1, \mu_2, \ldots, \mu_k$, suppose we take τ_i samples in the i^{th} arm, and let X_i be its empirical mean. Then we have*

$$\Pr\left[\left|\sum_{i=1}^{k} X_i - \sum_{i}^{k} \mu_i\right| \geq \epsilon\right] \leq 2\exp\left\{-\frac{\epsilon^2}{2\sum_{i=1}^{k} 1/\tau_i}\right\}.$$

Proof [Proof of Lemma 2.1] By assumption, $\sum_{i=1}^{k} X_i - \sum_{i=1}^{k} \mu_i$ follows the Gaussian distribution with mean 0 and variance $\sum_{i=1}^{k} 1/\tau_i$. The lemma hence follows from the tail bound of Gaussian distributions. ■

Tail bound of the χ^2 distribution. A χ^2 distribution with n degrees of freedom is the distribution of a sum of the squares of n random variables drawn independently from the standard Gaussian distribution $\mathcal{N}(0, 1)$. The following lemma, as a special case of (Laurent and Massart, 2000, Lemma 1), proves an exponential tail probability bound for χ^2 distributions.

Lemma 2.2 *Let X be a χ^2 random variable with n degrees of freedom. For any $x > 0$, it holds that*

$$\Pr[X \geq 2n + 3x] \leq e^{-x}.$$

Let $\mathrm{KL}(a_1, a_2)$ denote the Kullback-Leibler divergence from the distribution of arm a_2 to that of arm a_1. For two Gaussian arms a_1 and a_2 with means μ_1 and μ_2 respectively, it holds that

$$\mathrm{KL}(a_1, a_2) = \frac{1}{2}(\mu_1 - \mu_2)^2.$$

Moreover, let

$$d(x, y) = x \ln(x/y) + (1 - x) \ln[(1 - x)/(1 - y)]$$

denote the binary relative entropy function.

Change of Distribution. The following \Change of Distribution" lemma, formulated by Kaufmann et al. (2014), characterizes the behavior of an algorithm when underlying distributions of the arms are slightly altered, and is thus useful for proving sample complexity lower bounds. Similar bounds are known in the sequential hypothesis testing literature (see e.g., (Ghosh, 1970, p.283)) In the following, $\Pr_{\mathbb{A},\mathcal{C}}$ and $\mathbb{E}_{\mathbb{A},\mathcal{C}}$ denote the probability and expectation when algorithm $\mathbb{A}$ runs on instance $\mathcal{C}$.

Lemma 2.3 (Change of Distribution) *Let $\mathbb{A}$ be an algorithm that runs on n arms, and let $\mathcal{C} = (a_1, a_2, \ldots, a_n)$ and $\mathcal{C}' = (a'_1, a'_2, \ldots, a'_n)$ be two sequences of n arms. Let random*

variable τ_i denote the number of samples taken from the i-th arm. For any event $\mathcal{E}$ in $\mathcal{F}_\tau$, where τ is a stopping time with respect to the filtration $\{\mathcal{F}_t\}_{t\geq 0}$, it holds that

$$\sum_{i=1}^{n} \mathbb{E}_{\mathbb{A},\mathcal{C}}[\tau_i]\,\mathrm{KL}\left(a_i, a_i'\right) \geq d\left(\Pr_{\mathbb{A},\mathcal{C}}[\mathcal{E}], \Pr_{\mathbb{A},\mathcal{C}'}[\mathcal{E}]\right).$$

3. Instance Lower Bound

3.1. Instance Lower Bound for Best-Set

Given a Best-Set instance $\mathcal{C} = (S, \mathcal{F})$, let O denote the optimal set in $\mathcal{F}$ (i.e., $O = \mathrm{argmax}_{A\in\mathcal{F}} \mu(A)$). We de ne Low($\mathcal{C}$) as the optimal value of the following mathematical program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i\in S} \tau_i \\ \text{subject to} \quad & \sum_{i\in O\triangle A} 1/\tau_i \leq [\mu(O) - \mu(A)]^2 \qquad \forall A \in \mathcal{F} \\ & \tau_i > 0, \ \forall i \in S. \end{aligned} \tag{1}$$

We prove Theorem 1.6, which we restate in the following for convenience.

Theorem 1.6 (restated) *Let $\mathcal{C} = (S, \mathcal{F})$ be an instance of* Best-Set*. For any $\delta \in (0, 0.1)$ and δ-correct algorithm $\mathbb{A}$ for* Best-Set*, $\mathbb{A}$ takes $(\mathit{Low}(\mathcal{C}) \ln \delta^{-1})$ samples in expectation on $\mathcal{C}$.*

Proof [Proof of Theorem 1.6] Fix $\delta \in (0, 0.1)$, instance $\mathcal{C}$ and δ-correct algorithm $\mathbb{A}$. Let n_i be the expected number of samples drawn from the i-th arm when $\mathbb{A}$ runs on instance $\mathcal{C}$. Let $\alpha = d(1-\delta, \delta)/2$ and $\tau_i = n_i/\alpha$. It su ces to show that τ is a feasible solution for the program in (1), as it directly follows that

$$\sum_{i=1}^{n} n_i = \alpha \sum_{i=1}^{n} \tau_i \geq \alpha \mathrm{Low}(\mathcal{C}) = \quad (\mathrm{Low}(\mathcal{C}) \ln \delta^{-1}).$$

Here the last step holds since for all $\delta \in (0, 0.1)$,

$$d(1-\delta, \delta) = (1-2\delta) \ln \frac{1-\delta}{\delta} \geq 0.8 \ln \frac{1}{\sqrt{\delta}} = 0.4 \ln \delta^{-1}.$$

To show that τ is a feasible solution, we x $A \in \mathcal{F}$. Let $_i = c/n_i$, where

$$c = \frac{2[\mu(O) - \mu(A)]}{\sum_{i\in O\triangle A} 1/n_i}.$$

We consider the following alternative instance $\mathcal{C}'$: the mean of each arm i in $O \setminus A$ is decreased by $_i$, while the mean of each arm $i \in A \setminus O$ is increased by $_i$; the collection of feasible sets are identical to that in $\mathcal{C}$. Note that in $\mathcal{C}'$, the di erence between the weights of O and A is given by

$$\left(\mu(O) - \sum_{i\in O\setminus A} {}_i\right) - \left(\mu(A) + \sum_{i\in A\setminus O} {}_i\right) = \mu(O) - \mu(A) - c \sum_{i\in O\triangle A} 1/n_i = -(\mu(O) - \mu(A)) < 0.$$

In other words, O is no longer optimal in $\mathcal{C}'$.

Let $\mathcal{E}$ denote the event that algorithm $\mathbb{A}$ returns O as the optimal set. Note that since $\mathbb{A}$ is δ-correct, $\Pr_{\mathbb{A},\mathcal{C}}[\mathcal{E}] \geq 1-\delta$ and $\Pr_{\mathbb{A},\mathcal{C}'}[\mathcal{E}] \leq \delta$. Therefore, by Lemma 2.3,

$$\sum_{i \in O \triangle A} n_i \cdot \frac{1}{2} \quad {}_i^2 \geq d\left(\Pr_{\mathbb{A},\mathcal{C}}[\mathcal{E}], \Pr_{\mathbb{A},\mathcal{C}'}[\mathcal{E}]\right) \geq d(1-\delta, \delta).$$

Plugging in the values of ${}_i$'s yields

$$2d(1-\delta,\delta) \leq \sum_{i \in O \triangle A} n_i \cdot \frac{c^2}{n_i^2} = \frac{4[\mu(O)-\mu(A)]^2}{\left(\sum_{i \in O \triangle A} 1/n_i\right)^2} \sum_{i \in O \triangle A} 1/n_i = \frac{4[\mu(O)-\mu(A)]^2}{\sum_{i \in O \triangle A} 1/n_i},$$

and it follows that

$$\sum_{i \in O \triangle A} 1/\tau_i \leq [\mu(O)-\mu(A)]^2.$$

■

3.2. Comparison with Previous Lower Bound

In this section, we show that our lower bound is no weaker than the lower bound in Chen et al. (2014). Formally, we prove Theorem 1.7. We restate it here for convenience.

Theorem 1.7 (restated) *Let $\mathcal{C} = (S, \mathcal{F})$ be an instance of* Best-Set*,*

$$\mathit{Low}(\mathcal{C}) \geq H_C(\mathcal{C}).$$

Proof We start with the concept of *gap*, which was defined as follows in Chen et al. (2014). Let $\mathcal{C} = (S, \mathcal{F})$ be an instance of Best-Set with the optimal set $O \in \mathcal{F}$. For each arm i, its gap ${}_i$ is defined as

$${}_i = \begin{cases} \mu(O) - \max_{O' \in \mathcal{F}, i \notin O'} \mu(O'), & i \in O, \\ \mu(O) - \max_{O' \in \mathcal{F}, i \in O'} \mu(O'), & i \notin O. \end{cases}$$

The *hardness* of the instance $\mathcal{C}$, $H_C(\mathcal{C})$, is defined as $H_C(\mathcal{C}) = \sum_{i \in S} m_i'$, where $m_i' = \quad {}_i^{-2}$.

Consider the following mathematical program, which is essentially the same as Program (1), except for replacing summation with maximization in the constraints.

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in S} \tau_i' \\ \text{subject to} \quad & \max_{i \in O \triangle A} 1/\tau_i' \leq [\mu(O)-\mu(A)]^2 \qquad \forall A \in \mathcal{F} \\ & \tau_i' > 0, \ \forall i \in S. \end{aligned} \tag{2}$$

A simple observation is that, the optimal solution of Program (2) can be achieved by setting

$$\tau_i' = \max_{i \in O \triangle A} [\mu(O)-\mu(A)]^{-2} = m_i'.$$

Furthermore, every feasible solution of Program (1) is also a feasible solution of Program (2). Theorem 1.7 hence holds. ■

3.3. Instance Lower Bound for General-Samp

For a General-Samp instance $\mathcal{I} = (S, \mathcal{O})$ with mean pro le $\mu \in O$, we de ne Low($\mathcal{I}$) as the optimal value of the following linear program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} \tau_i \\ \text{subject to} \quad & \sum_{i=1}^{n} (\nu_i - \mu_i)^2 \tau_i \geq 1, \ \forall \nu \in \mathrm{Alt}(O), \\ & \tau_i \geq 0. \end{aligned} \tag{3}$$

Here Alt(O) = $\bigcup_{O' \in \mathcal{O} \setminus \{O\}} O'$.

We prove that (Low($\mathcal{I}$) ln δ^{-1}) is an instance-wise sample complexity lower bound for instance $\mathcal{I}$. The proof is very similar to that in Garivier and Kaufmann (2016), and we provide one for completeness.

Theorem 3.1 *Suppose $\delta \in (0, 0.1)$ and $\mathcal{I}$ is an instance of* General-Samp. *Any δ-correct algorithm for* General-Samp *takes* (*Low*($\mathcal{I}$) ln δ^{-1}) *samples in expectation on $\mathcal{I}$.*

Proof Fix $\delta \in (0, 0.1)$, instance $\mathcal{I}$ and a δ-correct algorithm $\mathbb{A}$. Let n_i be the expected number of samples drawn from the i-th arm when $\mathbb{A}$ runs on instance $\mathcal{I}$. Let $\alpha = 2d(1-\delta, \delta)$ and $\tau_i = n_i/\alpha$. It su ces to show that τ is a feasible solution for the program in (3), as it directly follows that the expected sample complexity of $\mathbb{A}$ is lower bounded by

$$\sum_{i=1}^{n} n_i = \alpha \sum_{i=1}^{n} \tau_i \geq \alpha \mathrm{Low}(\mathcal{I}) = \quad (\mathrm{Low}(\mathcal{I}) \ln \delta^{-1}).$$

Here the last step holds since for all $\delta \in (0, 0.1)$,

$$d(1-\delta, \delta) = (1-2\delta) \ln \frac{1-\delta}{\delta} \geq 0.8 \ln \frac{1}{\sqrt{\delta}} = 0.4 \ln \delta^{-1}.$$

To show that τ is a feasible solution, we x $\nu \in \mathrm{Alt}(O)$. Let $\mathcal{I}'$ be an alternative instance obtained by changing the mean pro le in $\mathcal{I}$ from μ to ν. Let $\mathcal{E}$ denote the event that algorithm $\mathbb{A}$ returns O as the optimal set. The δ-correctness of $\mathbb{A}$ guarantees that $\Pr_{\mathbb{A},\mathcal{I}}[\mathcal{E}] \geq 1 - \delta$ and $\Pr_{\mathbb{A},\mathcal{I}'}[\mathcal{E}] \leq \delta$. By Lemma 2.3,

$$\sum_{i=1}^{n} n_i \cdot \mathrm{KL}(\mathcal{N}(\mu_i, 1), \mathcal{N}(\nu_i, 1)) \geq d\left(\Pr_{\mathbb{A},\mathcal{I}}[\mathcal{E}], \Pr_{\mathbb{A},\mathcal{I}'}[\mathcal{E}]\right) \geq d(1-\delta, \delta).$$

Plugging in $\alpha = 2d(1-\delta, \delta)$ and $\mathrm{KL}(\mathcal{N}(\mu_i, 1), \mathcal{N}(\nu_i, 1)) = \frac{1}{2}(\nu_i - \mu_i)^2$ yields

$$\alpha = 2d(1-\delta, \delta) \leq \sum_{i=1}^{n} n_i \cdot (\nu_i - \mu_i)^2 = \alpha \sum_{i=1}^{n} (\nu_i - \mu_i)^2 \tau_i,$$

and it follows that $\sum_{i=1}^{n}(\nu_i - \mu_i)^2\tau_i \geq 1$. ■

4. Optimal Algorithm for Combinatorial Bandit

In this section, we present an algorithm for Best-Set that nearly achieves the optimal sample complexity. We postpone the computationally e cient implementation of the algorithm to the next section.

4.1. Overview

Our algorithm is based on a process of successive elimination. However, unlike the previous approaches Karnin et al. (2013); Chen and Li (2015); Chen et al. (2016a); Gabillon et al. (2016) which maintained a set of arms, our algorithm maintains a collection of candidate sets and performs the eliminations on them directly. Speci cally, at the r-th round of the algorithm, we adopt a precision level $\epsilon_r := 2^{-r}$, and maintain a set of candidates sets $\mathcal{F}_r \subseteq \mathcal{F}$; and the goal of the r-th round is to eliminate those sets in $\mathcal{F}_r$ with a optimality gap[8] of at least (ϵ_r).

A crucial di culty in implementing the above elimination is that it seems we have to compute the optimal set itself in order to approximate the optimality gaps. To circumvent this problem, we instead approximate the gaps between $every\ pair$ of sets in $\mathcal{F}_r$, which certainly include the optimality gaps. Roughly speaking, we solve an optimization similar to that in (1), and sample the arms accordingly to obtain an $O(\epsilon_r)$ approximation of the gap between $every\ pair$ of sets that are still under consideration. Now if there is a set pair (A, B) such that the empirical mean of B exceeds that of A by ϵ_r, we are certain that A is not the optimal set, and thus we stop considering A as a candidate answer. This process is repeated until only one set remains.

4.2. Algorithm

We rst de ne a few useful subroutines that play important roles in the algorithm. Procedure SimultEst takes as its input a set $\mathcal{U} \subseteq \mathcal{F}$ together with an accuracy parameter ϵ and a con dence level δ. It outputs a vector $\{m_i\}_{i\in S}$ over S that speci es the number of samples

8. The optimality gap for a set $A \in \mathcal{F}$ is simply $\mu(O) - \mu(A)$ where O is the optimal set.

that should be taken from each arm, so that the difference between any two sets in $\mathcal{U}$ can be estimated to an accuracy of ϵ with probability $1-\delta$.

Algorithm 1: SimultEst($\mathcal{U}, \epsilon, \delta$)

Input: $\mathcal{U}$, accuracy parameter ϵ, and confidence level δ.

Output: A vector m, indicating the number of samples to be taken from each arm.

1 Let $\{m_i\}_{i\in S}$ be the optimal solution of the following program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i\in S} m_i \\ \text{subject to} \quad & \sum_{i\in A\triangle B} \frac{1}{m_i} \le \frac{\epsilon^2}{2\ln(2/\delta)}, \ \forall A, B \in \mathcal{U} \\ & m_i > 0, \ \forall i \in S \end{aligned}$$

2 return m;

Procedure Verify takes a sequence $\mathcal{F}_1, \ldots, \mathcal{F}_r$ of subsets of $\mathcal{F}$ together with a confidence parameter δ. Similar to SimultEst, it returns a vector $\{m_i\}_{i\in S}$ of the number of samples from each arm, so that the gap between the conjectured answer $\widehat{O}$, the only set in $\mathcal{F}_r$, and each set in $\mathcal{F}_k$ can be estimated to $O(\epsilon_k)$ accuracy. (Recall that $\epsilon_k = 2^{-k}$.) Notice that in general, the solutions returned by SimultEst and Verify are real-valued. We can simply get an integer-valued solution by rounding up without affecting the asymptotical performance.

Algorithm 2: Verify($\{\mathcal{F}_k\}, \delta$)

Input: $\{\mathcal{F}_k\}_{k\in[r]}$ and confidence level δ. It is guaranteed that $|\mathcal{F}_r| = 1$.

Output: A vector m, indicating the number of samples to be taken from each arm.

1 $\hat{O} \leftarrow$ the only set in $\mathcal{F}_r$; $\lambda \leftarrow 10$;

2 Let $\{m_i\}_{i\in S}$ be the optimal solution of the following program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i\in S} m_i \\ \text{subject to} \quad & \sum_{i\in \hat{O}\triangle A} \frac{1}{m_i} \le \frac{(\epsilon_k/\lambda)^2}{2\ln(2/\delta)}, \ \forall k \in [r], A \in \mathcal{F}_k \\ & m_i > 0, \ \forall i \in S \end{aligned}$$

3 return m;

Sample is a straightforward sampling procedure: it takes a vector m, samples each arm $i \in S$ exactly m_i times, and returns the empirical means. We finish the description of all subroutines.

Remark 4.1 *In this section, we focus on the sample complexity and do not worry too much about the computation complexity of our algorithm. It would be convenient to think that the subsets in $\mathcal{F}$ are given explicitly as input. In this case, the convex programs in the subroutines can be solved in time polynomial in n and $|\mathcal{F}|$. We will consider computational complexity issues when $\mathcal{F}$ is given implicitly in Section 5.*

Now, we describe our algorithm NaiveGapElim for Best-Set. NaiveGapElim proceeds in rounds. In each round r, it calls the SimultEst procedure and samples the arms in S accordingly, and then removes the sets with (ϵ_r) gaps from $\mathcal{F}_r$. When exactly one set $\hat{O}$ remains in $\mathcal{F}_r$ (i.e., the condition at line 3 is met), the algorithm calls Verify and Sample, and veri es that the conjectured answer $\hat{O}$ is indeed optimal.

Algorithm 3: NaiveGapElim$(\mathcal{C}, \delta)$

Input: Best-Set instance $\mathcal{C} = (S, \mathcal{F})$ and con dence level δ.
Output: The answer.

1 $\mathcal{F}_1 \leftarrow \mathcal{F}$; $\delta_0 \leftarrow 0.01$; $\lambda \leftarrow 10$;
2 for $r = 1$ to ∞ do
3 if $|\mathcal{F}_r| = 1$ then
4 $m \leftarrow \text{Verify}(\{\mathcal{F}_k\}_{k=1}^{r}, \delta/(r|\mathcal{F}|))$;
5 $\hat{\mu} \leftarrow \text{Sample}(m)$;
6 $\hat{O} \leftarrow$ the only set in $\mathcal{F}_r$;
7 if $\hat{\mu}(\hat{O}) - \hat{\mu}(A) \geq \epsilon_k/\lambda$ for all $A \in \mathcal{F} \setminus \mathcal{F}_k$ and all $k \in [r]$ then
8 return $\hat{O}$;
9 else
10 return error;
11 $\epsilon_r \leftarrow 2^{-r}$; $\delta_r \leftarrow \delta_0/(10r^2|\mathcal{F}|^2)$;
12 $m^{(r)} \leftarrow \text{SimultEst}(\mathcal{F}_r, \epsilon_r/\lambda, \delta_r)$;
13 $\hat{\mu}^{(r)} \leftarrow \text{Sample}(m^{(r)})$;
14 $\text{opt}_r \leftarrow \max_{A \in \mathcal{F}_r} \hat{\mu}^{(r)}(A)$;
15 $\mathcal{F}_{r+1} \leftarrow \{A \in \mathcal{F}_r : \hat{\mu}^{(r)}(A) \geq \text{opt}_r - \epsilon_r/2 - 2\epsilon_r/\lambda\}$;

4.3. Correctness of **NaiveGapElim**

We formally state the correctness guarantee of NaiveGapElim in the following lemma.

Lemma 4.2 *For any $\delta \in (0, 0.01)$ and* Best-Set *instance $\mathcal{C}$, NaiveGapElim$(\mathcal{C}, \delta)$ returns the correct answer with probability $1-\delta_0-\delta$, and returns an incorrect answer w.p. at most δ.*

The proof proceeds as follows. We rst de ne two \good events" $\mathcal{E}_0^{\text{good}}$ and $\mathcal{E}^{\text{good}}$, which happen with probability at least $1-\delta_0$ and $1-\delta$, respectively. Our algorithm always returns the correct answer conditioned on $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$, and it either returns the correct answer or reports an error conditioned on $\mathcal{E}^{\text{good}}$. This implies that NaiveGapElim is $(\delta + \delta_0)$-correct. This is not ideal since δ_0 is a xed constant, so in Section 4.5 we use a parallel simulation construction to boost its success probability to $1 - \delta$, while retaining the same sample complexity.

Good events. De ne $\mathcal{E}_{0,r}^{\text{good}}$ as the event that either the algorithm terminates before round r, or for all $A, B \in \mathcal{F}_r$,

$$\left|(\hat{\mu}^{(r)}(A) - \hat{\mu}^{(r)}(B)) - (\mu(A) - \mu(B))\right| < \epsilon_r/\lambda.$$

Here λ is the constant in NaiveGapElim. Moreover, we define $\mathcal{E}_0^{\text{good}}$ as the intersection of $\{\mathcal{E}_{0,r}^{\text{good}}\}$, i.e.,

$$\mathcal{E}_0^{\text{good}} := \bigcap_{r=1}^{\infty} \mathcal{E}_{0,r}^{\text{good}}.$$

$\mathcal{E}^{\text{good}}$ is defined as the event that for all $A \in \mathcal{F}_r$,

$$\left|(\hat{\mu}(\hat{O}) - \hat{\mu}(A)) - (\mu(\hat{O}) - \mu(A))\right| < \epsilon_r/\lambda.$$

Here $\hat{\mu}$ and $\hat{O}$ are defined at lines 5 and 6 in NaiveGapElim.

Lemma 4.3 $\Pr\left[\mathcal{E}_0^{\text{good}}\right] \geq 1-\delta_0$ *and* $\Pr\left[\mathcal{E}^{\text{good}}\right] \geq 1-\delta$.

Proof [Proof of Lemma 4.3] Since $m^{(r)}$ is a feasible solution of the program in SimultEst$(\mathcal{F}_r, \epsilon_r/\lambda, \delta_r)$, it holds for all $A, B \in \mathcal{F}_r$ that

$$\sum_{i \in A \triangle B} \frac{1}{m_i^{(r)}} \leq \frac{(\epsilon_r/\lambda)^2}{2\ln(2/\delta_r)}.$$

By Lemma 2.1,

$$\begin{aligned}
&\Pr\left[\left|(\hat{\mu}^{(r)}(A) - \hat{\mu}^{(r)}(B)) - (\mu(A) - \mu(B))\right| \geq \epsilon_r/\lambda\right] \\
&= \Pr\left[\left|(\hat{\mu}^{(r)}(A \setminus B) - \hat{\mu}^{(r)}(B \setminus A)) - (\mu(A \setminus B) - \mu(B \setminus A))\right| \geq \epsilon_r/\lambda\right] \\
&\leq 2\exp\left\{-\frac{(\epsilon_r/\lambda)^2}{2\sum_{i \in A \triangle B} 1/m_i^{(r)}}\right\} \\
&\leq 2\exp\left(-\ln(2/\delta_r)\right) = \delta_r.
\end{aligned}$$

By a union bound over all possible $A, B \in \mathcal{F}_r$, we have $\Pr\left[\overline{\mathcal{E}_{0,r}^{\text{good}}}\right] \leq |\mathcal{F}|^2 \delta_r = \delta_0/(10r^2)$. It follows from another union bound that

$$\Pr\left[\mathcal{E}_0^{\text{good}}\right] \geq 1 - \sum_{r=1}^{\infty} \Pr\left[\overline{\mathcal{E}_{0,r}^{\text{good}}}\right] \geq 1 - \sum_{r=1}^{\infty} \frac{\delta_0}{10r^2} \geq 1 - \delta_0.$$

A similar union bound argument over all $k \in [r]$ and $A \in \mathcal{F}_r$ yields that $\Pr\left[\mathcal{E}^{\text{good}}\right] \geq 1-\delta$. ■

Implications of good events. Let O denote the optimal set in $\mathcal{F}$, i.e., $O = \text{argmax}_{A \in \mathcal{F}} \mu(A)$. Throughout the analysis of our algorithm, it is useful to group the sets in $\mathcal{F}$ based on the gaps between their weights and $\mu(O)$. Formally, we define G_r as

$$G_r := \{A \in \mathcal{F} : \mu(O) - \mu(A) \in (\epsilon_{r+1}, \epsilon_r]\}. \tag{4}$$

We also adopt the shorthand notation $G_{\geq r} = \{O\} \cup \bigcup_{k=r}^{\infty} G_k$.

Lemma 4.4 *Conditioning on* $\mathcal{E}_0^{\text{good}}$, $O \in \mathcal{F}_r$ *for all* $r \geq 1$.

Proof [Proof of Lemma 4.4] Suppose for a contradiction that $O \in \mathcal{F}_r \setminus \mathcal{F}_{r+1}$ for some r. By definition of $\mathcal{F}_{r+1}$, it holds that

$$\hat{\mu}^{(r)}(O) < \mathsf{opt}_r - \epsilon_r/2 - 2\epsilon_r/\lambda.$$

Observe that $\mathsf{opt}_r = \hat{\mu}^{(r)}(A)$ for some $A \in \mathcal{F}_r$. Therefore, $\hat{\mu}^{(r)}(O) - \hat{\mu}^{(r)}(A) < -(1/2 + 2/\lambda)\epsilon_r$. Since O is the maximum-weight set with respect to μ, $\mu(O) - \mu(A) \geq 0$. These two inequalities imply that

$$\left|(\hat{\mu}^{(r)}(O) - \hat{\mu}^{(r)}(A)) - (\mu(O) - \mu(A))\right| > 0 - [-(1/2 + 2/\lambda)\epsilon_r] > \epsilon_r/\lambda,$$

which happens with probability zero conditioning on event $\mathcal{E}_0^{\mathsf{good}}$, since $O, A \in \mathcal{F}_r$. ■

The following lemma, as a generalization of Lemma 4.4 to all sets in $\mathcal{F}$, states that the sequence $\{\mathcal{F}_r\}$ is an approximation of $\{G_{\geq r}\}$ conditioning on event $\mathcal{E}_0^{\mathsf{good}}$.

Lemma 4.5 *Conditioning on $\mathcal{E}_0^{\mathsf{good}}$, $G_{\geq r} \supseteq \mathcal{F}_{r+1} \supseteq G_{\geq r+1}$ for all $r \geq 1$.*

Proof [Proof of Lemma 4.5] We first prove the left inclusion. Suppose that $A \in \mathcal{F}_r$ and $A \notin G_{\geq r}$. By definition of $G_{\geq r}$, $\mu(O) - \mu(A) > \epsilon_r$. Conditioning on $\mathcal{E}_0^{\mathsf{good}}$, we have $O \in \mathcal{F}_r$ by Lemma 4.4, and thus

$$\hat{\mu}^{(r)}(O) - \hat{\mu}^{(r)}(A) > \mu(O) - \mu(A) - \epsilon_r/\lambda > (1 - 1/\lambda)\epsilon_r.$$

Recall that $\mathsf{opt}_r = \max_{A \in \mathcal{F}_r} \hat{\mu}^{(r)}(A) \geq \hat{\mu}^{(r)}(O)$, and $1 - 1/\lambda > 1/2 + 2/\lambda$ by our choice of λ. It follows that

$$\hat{\mu}^{(r)}(A) < \hat{\mu}^{(r)}(O) - (1 - 1/\lambda)\epsilon_r < \mathsf{opt}_r - \epsilon_r/2 - 2\epsilon_r/\lambda,$$

and thus $A \notin \mathcal{F}_{r+1}$.

Then we show that $A \in G_{\geq r+1}$ implies $A \in \mathcal{F}_{r+1}$. Note that $A \in G_{\geq r+1}$ implies $\mu(O) - \mu(A) \leq \epsilon_{r+1} = \epsilon_r/2$, and thus,

$$\hat{\mu}^{(r)}(O) - \hat{\mu}^{(r)}(A) \leq \mu(O) - \mu(A) + \epsilon_r/\lambda \leq (1/2 + 1/\lambda)\epsilon_r.$$

Moreover, since $\mathsf{opt}_r = \hat{\mu}^{(r)}(B)$ for some $B \in \mathcal{F}_r$, it holds that

$$\mathsf{opt}_r - \hat{\mu}^{(r)}(O) = \hat{\mu}^{(r)}(B) - \hat{\mu}^{(r)}(O) \leq \mu(B) - \mu(O) + \epsilon_r/\lambda \leq \epsilon_r/\lambda.$$

Adding the two inequalities above yields

$$\hat{\mu}^{(r)}(A) \geq \mathsf{opt}_r - (1/2 + 2/\lambda)\epsilon_r.$$

By definition of $\mathcal{F}_{r+1}$ in NaiveGapElim, $A \in \mathcal{F}_{r+1}$, which completes the proof. ■

Correctness conditioning on $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$. By Lemma 4.4, conditioning on $\mathcal{E}_0^{\text{good}}$, the correct answer is in $\mathcal{F}_r$ for every r. This guarantees that whenever the algorithm enters the if-statement (i.e., when $|\mathcal{F}_r| = 1$), it holds that $\mathcal{F}_r = \{O\}$. Moreover, let r^* be a sufficiently large integer such that $G_{\geq r^*} = G_{\geq r^*+1} = \{O\}$. Then Lemma 4.5 implies that $\mathcal{F}_{r^*+1} = \{O\}$, and consequently the algorithm eventually enters the if-statement, either before or at round $r^* + 1$.

Now we show that the algorithm always returns the correct answer O instead of reporting an error, conditioning on $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$. Fix $A \in \mathcal{F} \setminus \{O\}$. Let r be the largest integer such that $A \in \mathcal{F}_r$, i.e., $A \in \mathcal{F}_r \setminus \mathcal{F}_{r+1}$. By Lemma 4.5, we have $\mathcal{F}_{r+1} \supseteq G_{\geq r+1}$. It follows that $A \notin G_{\geq r+1}$, and thus $\mu(O) - \mu(A) > \epsilon_{r+1} = \epsilon_r/2$.

Recall that since $O, A \in \mathcal{F}_r$, conditioning on event $\mathcal{E}^{\text{good}}$, $\hat{\mu}(O) - \hat{\mu}(A)$ is within an additive error of ϵ_r/λ to $\mu(O) - \mu(A)$. Therefore,

$$\hat{\mu}(O) - \hat{\mu}(A) > \mu(O) - \mu(A) - \epsilon_r/\lambda > (1/2 - 1/\lambda)\epsilon_r > \epsilon_r/\lambda.$$

Here the last step follows from our choice of parameter λ.

Consequently, the condition at line 7 of NaiveGapElim is always met conditioning on $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$, and NaiveGapElim returns the correct answer O.

Soundness conditioning on $\mathcal{E}^{\text{good}}$. Finally, we show that conditioning on $\mathcal{E}^{\text{good}}$, NaiveGapElim either returns the correct answer or reports an error. Suppose that when the algorithm enters the if-statement at line 3, $\mathcal{F}_r$ is equal to $\{\hat{O}\}$ for some $\hat{O} \neq O$. Let r be the unique integer that satisfies $O \in \mathcal{F}_r \setminus \mathcal{F}_{r+1}$. Recall that since $O, \hat{O} \in \mathcal{F}_r$, conditioning on event $\mathcal{E}^{\text{good}}$,

$$\left|(\hat{\mu}(\hat{O}) - \hat{\mu}(O)) - (\mu(\hat{O}) - \mu(O))\right| < \epsilon_r/\lambda.$$

By definition, $\mu(\hat{O}) - \mu(O) < 0$, and it follows that

$$\hat{\mu}(\hat{O}) - \hat{\mu}(O)) < \mu(\hat{O}) - \mu(O) + \epsilon_r/\lambda < \epsilon_r/\lambda.$$

This guarantees that the condition at line 7 is not met when $\hat{O} \neq O$, and thus the algorithm does not incorrectly return $\hat{O}$.

4.4. Sample Complexity

We analyze the sample complexity of the NaiveGapElim algorithm conditioning on event $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$. Let $\Delta = \mu(O) - \max_{A \in \mathcal{F} \setminus \{O\}} \mu(A)$ denote the gap between the set with the second largest weight in $\mathcal{F}$ and the weight of O.

Lemma 4.6 *For any $\delta \in (0, 0.01)$ and* Best-Set *instance $\mathcal{C}$, NaiveGapElim$(\mathcal{C}, \delta)$ takes*

$$O\left(\mathit{Low}(\mathcal{C}) \ln \delta^{-1} + \mathit{Low}(\mathcal{C}) \ln \Delta^{-1} \left(\ln \ln \Delta^{-1} + \ln |\mathcal{F}|\right)\right)$$

samples conditioning on event $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$.

Proof Recall that for a Best-Set instance $\mathcal{C} = (S, \mathcal{F})$, Low$(\mathcal{C})$ is defined as

$$\text{Low}(\mathcal{C}) := \sum_{i \in S} \tau_i^*,$$

where τ^* denotes the optimal solution to the following program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in S} \tau_i \\ \text{subject to} \quad & \sum_{i \in O \triangle A} 1/\tau_i \le [\mu(O) - \mu(A)]^2, \ \forall A \in \mathcal{F} \\ & \tau_i > 0, \ \forall i \in S. \end{aligned} \tag{5}$$

For each r, we construct a feasible solution to the corresponding program in SimultEst($\mathcal{F}_r, \epsilon_r/\lambda, \delta_r$), thereby proving an upper bound on the number of samples taken in round r. Let $\alpha = 16\lambda^2 \ln(2/\delta_r)$ and $m_i = \alpha\tau_i^*$. Fix $A, B \in \mathcal{F}_r$. By Lemma 4.5, we have $A, B \in G_{\ge r-1}$, and thus $\mu(O) - \mu(A) \le \epsilon_{r-1}$ and $\mu(O) - \mu(B) \le \epsilon_{r-1}$. Therefore,

$$\begin{aligned} \sum_{i \in A \triangle B} 1/m_i &\le \alpha^{-1} \left(\sum_{i \in O \triangle A} 1/\tau_i^* + \sum_{i \in O \triangle B} 1/\tau_i^* \right) \\ &\le \alpha^{-1} \left[[\mu(O) - \mu(A)]^2 + [\mu(O) - \mu(B)]^2 \right] \\ &\le 2\alpha^{-1}\epsilon_{r-1}^2 = \frac{(\epsilon_r/\lambda)^2}{2\ln(2/\delta_r)}. \end{aligned}$$

Here the second step holds since τ^* is a feasible solution to the program in (1). The third step follows from $\mu(O) - \mu(A) \le \epsilon_{r-1}$ and $\mu(O) - \mu(B) \le \epsilon_{r-1}$. Finally, the last step applies $\alpha = 16\lambda^2 \ln(2/\delta_r)$.

Therefore, $\{m_i\}$ is a valid solution to the program in SimultEst, and then the number of samples taken in round r is upper bounded by

$$\sum_{i \in S} m_i = \alpha \sum_{i \in S} \tau_i^* = O(\text{Low}(\mathcal{C}) \ln \delta_r^{-1}) = O\left(\text{Low}(\mathcal{C}) \left(\ln r + \ln |\mathcal{F}|\right)\right).$$

The last step holds due to

$$\ln \delta_r^{-1} = \ln(10r^2|\mathcal{F}|^2/\delta_0) = O(\ln r + \ln |\mathcal{F}|).$$

Recall that $\quad = \mu(O) - \max_{A \in \mathcal{F} \setminus \{O\}} \mu(A)$. Let $r^* = \lfloor \log_2 \quad ^{-1} \rfloor + 1$ be the smallest integer such that $\epsilon_{r^*} < \quad$. As shown in the proof of correctness, the algorithm terminates before round $r^* + 1$. Summing over all r between 1 and r^* yields

$$\begin{aligned} O\left(\text{Low}(\mathcal{C}) \sum_{r=1}^{r^*} (\ln r + \ln |\mathcal{F}|) \right) &= O\left(r^* \cdot \text{Low}(\mathcal{C}) \left(\ln r^* + \ln |\mathcal{F}|\right)\right) \\ &= O\left(\ln \quad ^{-1} \cdot \text{Low}(\mathcal{C}) \left(\ln \ln \quad ^{-1} + \ln |\mathcal{F}|\right)\right). \end{aligned}$$

It remains to upper bound the number of samples taken in the last round, denoted by round r. Let $\beta = 8\lambda^2 \ln(2r|\mathcal{F}|/\delta)$, and $m_i = \beta\tau_i^*$. Fix $k \in [r]$ and $A \in \mathcal{F}_k$. By Lemma 4.5, we have $A \in G_{\ge k-1}$, which implies that $\mu(O) - \mu(A) \le \epsilon_{k-1}$. It also follows from Lemma

4.4 that $\hat{O} = O$. Thus we have

$$\begin{aligned}\sum_{i \in \hat{O} \triangle A} 1/m_i &= \beta^{-1} \sum_{i \in O \triangle A} 1/\tau_i^* \\ &\leq \beta^{-1}[\mu(O) - \mu(A)]^2 \\ &\leq 4\beta^{-1}\epsilon_k^2 = \frac{(\epsilon_k/\lambda)^2}{2\ln(2r|\mathcal{F}|/\delta)}.\end{aligned}$$

In other words, $\{m_i\}$ is a feasible solution to the program in Verify($\{\mathcal{F}_k\}, \delta/(r|\mathcal{F}|)$). Therefore, the number of samples taken in the last round r is upper bounded by

$$\sum_{i \in S} m_i = \beta \sum_{i \in S} \tau_i^* = O\left(\mathsf{Low}(\mathcal{C})\left(\ln \delta^{-1} + \ln\ln \quad^{-1} + \ln|\mathcal{F}|\right)\right).$$

In sum, the number of samples taken by NaiveGapElim conditioning on $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$ is

$$O\left(\mathsf{Low}(\mathcal{C}) \ln \delta^{-1} + \mathsf{Low}(\mathcal{C}) \ln \quad^{-1}\left(\ln\ln \quad^{-1} + \ln|\mathcal{F}|\right)\right).$$

■

4.5. Parallel Simulation

In the above sections, we showed that conditioning on the \good" events we had low sample complexity and returned correct answers. We now show how to remove the conditioning and get a δ-correct algorithm with the same sample complexity in expectation (which is nearly optimal), using a \parallel simulation" idea. The idea was rst used in the Best-1-Arm problem in Chen and Li (2015).

De nition 4.7 *An algorithm $\mathbb{A}$ is (δ_0, δ, A, B)-correct if there exist two events $\mathcal{E}_0$ and $\mathcal{E}_1$ satisfying the following three conditions:*

1. $\Pr[\mathcal{E}_0] \geq 1 - \delta_0 - \delta$ *and* $\Pr[\mathcal{E}_1] \geq 1 - \delta$.
2. *Conditioning on $\mathcal{E}_0$, $\mathbb{A}$ returns the correct answer, and takes $O(A \ln \delta^{-1} + B)$ samples.*
3. *Conditioning on $\mathcal{E}_1$, $\mathbb{A}$ either returns the correct answer or terminates with an error.*

By Lemma 4.2 and Lemma 4.6, NaiveGapElim is a (δ_0, δ, A, B)-correct algorithm for Best-Set, where $\mathcal{E}_0 = \mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$ and $\mathcal{E}_1 = \mathcal{E}^{\text{good}}$, $\delta_0 = 0.01$, $A = \mathsf{Low}(\mathcal{C})$ and $B = \mathsf{Low}(\mathcal{C}) \ln \quad^{-1}(\ln\ln \quad^{-1} + \ln|\mathcal{F}|)$. The following lemma shows that we can obtain a δ-correct algorithm with the same $O(A \ln \delta^{-1} + B)$ sample complexity, thus proving Theorem 1.8.

Lemma 4.8 (Parallel Simulation) *If there is a (δ_0, δ, A, B) algorithm for a sampling problem for $\delta_0 = 0.01$ and any $\delta < 0.01$, there is also a δ-correct algorithm for any $\delta < 0.01$ that takes $O(A \ln \delta^{-1} + B)$ samples in expectation.*

We postpone the proof of Lemma 4.8 to Appendix A.

5. Optimal Algorithm for Combinatorial Bandit with E cient Computation

In this section, we present a computationally e cient implementation of the NaiveGapElim algorithm. Recall that NaiveGapElim maintains a sequence of set families $\{\mathcal{F}_r\}$, which contain the sets still under consideration at the beginning of each round r. As $|\mathcal{F}|$, the number of feasible sets, is typically exponential in the number of arms, it may be computationally expensive to compute $\{\mathcal{F}_r\}$ explicitly. The key to computational e ciency is to nd a compact representation of $\{\mathcal{F}_r\}$. In this paper, we represent $\mathcal{F}_{r+1}$ using the empirical means $\hat{\mu}^{(r)}$ and some carefully chosen threshold θ_r:

$$\mathcal{F}_{r+1} = \{A \in \mathcal{F} : \hat{\mu}^{(r)}(A) \geq \theta_r\}.$$

Consequently, we have to adapt the procedures in NaiveGapElim, including SimultEst and Verify, so that they work with this implicit representation of set families. To this end, we use the ε-approximate Pareto curve framework of Papadimitriou and Yannakakis (2000). This technique allows us to implement our subroutines in polynomial-time with respect to the input size and $1/\epsilon$, if we relax the constraints in the subroutines by an multiplicative factor of $1 + \epsilon$. In particular, if $1/\epsilon$ is upper bounded by the sample complexity of the instance, we would obtain a computationally e cient implementation of the algorithm.

5.1. Algorithm

We give a simpli ed version of the algorithm, and then later boost its probability of success by a parallel simulation (Lemma 4.8). The algorithm relies on computationally e cient implementations of the subroutines SimultEst and Verify, as well as three new procedures Unique, Check and OPT. We start by introducing the syntax and performance guarantees of these procedures, and postpone their e cient implementation to Section 5.4.

Procedure SimultEst takes as its input a weight μ on S, two thresholds θ^{high} and θ^{low}, together with an accuracy parameter ϵ and a con dence level δ, and outputs a vector $\{m_i\}_{i \in S}$ indicating the number of samples to be taken from each arm in S to estimate the di erence between any two sets in $\{A \in \mathcal{F} \mid \mu(A) \geq \theta^{\text{high}}\}$ to an accuracy of ϵ with con dence $1 - \delta$. This new procedure is akin to the version in Section 4, where we set $\mathcal{U} = \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$. While the lower threshold θ^{low} is not explicitly used, it gives us the approximation guarantee of the procedure: indeed, while SimultEst will be guaranteed to output a feasible solution to the original program, the resulting objective will be a constant approximation of the *tightened* program obtained by replacing $\{A' \in \mathcal{F} :$

$\mu(A') \geq \theta^{\text{high}}\}$ with $\{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{low}}\}$ in the constraints. A detailed specification of SimultEst appears in Section 5.2.

Algorithm 4: SimultEst($\mu, \theta^{\text{high}}, \theta^{\text{low}}, \epsilon, \delta$)

Input: Mean vector μ, thresholds θ^{high} and θ^{low}, accuracy parameter ϵ, confidence level δ.
Output: A vector m, indicating the number of samples to be taken from each arm.

1 Let $\{m_i\}_{i \in S}$ be an approximate solution to the following program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in S} m_i \\ \text{subject to} \quad & \sum_{i \in A \triangle B} \frac{1}{m_i} \leq \frac{\epsilon^2}{2\ln(2/\delta)}, \ \forall A, B \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\} \\ & m_i > 0, \ \forall i \in S \end{aligned}$$

2 return m;

Similarly, procedure Verify takes a sequence of means $\{\hat{\mu}^{(k)}\}$, two threshold sequences $\{\theta_k^{\text{high}}\}$ and $\{\theta_k^{\text{low}}\}$, together with a confidence parameter δ. It returns a vector $\{m_i\}$, indicating the number of samples from each arm, so that the gap between the conjectured answer $\hat{O}$ and each set in $\{A \in \mathcal{F} : \hat{\mu}^{(k-1)}(A) \geq \theta_{k-1}^{\text{high}}\}$ can be estimated to $O(\epsilon_k)$ accuracy. As in SimultEst, the resulting objective value is guaranteed to be bounded by a constant times the optimal value of the tightened program, obtained by replacing $\theta_{k-1}^{\text{high}}$ with $\theta_{k-1}^{\text{low}}$ in the contraint.

Algorithm 5: Verify($\{\hat{\mu}^{(k)}\}, \{\theta_k^{\text{high}}\}, \{\theta_k^{\text{low}}\}, \delta$)

Input: A sequence $\{\hat{\mu}^{(k)}\}_{k=0}^{r-1}$ of empirical means, threshold sequences $\{\theta_k^{\text{high}}\}_{k=0}^{r-1}$ and $\{\theta_k^{\text{low}}\}_{k=0}^{r-1}$, together with a confidence level δ.
Output: A vector m, indicating the number of samples to be taken from each arm.

1 $\lambda \leftarrow 10$; $\hat{O} \leftarrow \text{argmax}_{A \in \mathcal{F}} \hat{\mu}^{(r)}(A)$;

2 Let $\{m_i\}_{i \in S}$ be an approximate solution to the following program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in S} m_i \\ \text{subject to} \quad & \sum_{i \in \hat{O} \triangle A} \frac{1}{m_i} \leq \frac{(\epsilon_k/\lambda)^2}{2\ln(2/\delta)}, \ \forall k \in [r], A \in \{A' \in \mathcal{F} : \hat{\mu}^{(k-1)}(A') \geq \theta_{k-1}^{\text{high}}\} \\ & m_i > 0, \ \forall i \in S \end{aligned}$$

3 return m;

The EfficientGapElim algorithm (Algorithm 6) proceeds in rounds. At round r, EfficientGapElim first calls the subroutine Unique to determine whether exactly one set survives (i.e., has a weight greater than $\theta_{r-1} - \epsilon_{r-1}/\lambda$ with respect to $\hat{\mu}^{(r-1)}$. If so, the algorithm invokes Verify, Sample (a straightforward sampling procedure) and Check (a procedure analogous to Line 7 in NaiveGapElim), in order to verify that the conjectured answer $\hat{O}$ is indeed optimal. The algorithm terminates and depending on these tests, returns either $\hat{O}$ or an error.

Otherwise, EfficientGapElim calls SimultEst and Sample to estimate the means to su - cient accuracy. After that, OPT is called to compute the approximately optimal set among the sets under consideration. Finally, the algorithm computes the threshold for the next round based on opt_r.

Algorithm 6: EfficientGapElim$(\mathcal{C}, \delta)$

Input: Best-Set instance $\mathcal{C} = (S, \mathcal{F})$ and con dence level δ.
Output: The answer.

$\hat{\mu}^{(0)} \leftarrow 0$; $\theta_0 \leftarrow 0$;
$\delta_0 \leftarrow 0.01$; $\lambda \leftarrow 20$;
for $r = 1$ to ∞ do
  if *Unique*$(\hat{\mu}^{(r-1)}, \theta_{r-1} - \epsilon_{r-1}/\lambda)$ then
    $m \leftarrow \text{Verify}(\{\hat{\mu}^{(k)}\}_{k=0}^{r-1}, \{\theta_k - \epsilon_k/\lambda\}_{k=0}^{r-1}, \{\theta_k - 2\epsilon_k/\lambda\}_{k=0}^{r-1}, \delta/(r|\mathcal{F}|))$;
    $\hat{\mu} \leftarrow \text{Sample}(m)$;
    $\hat{O} \leftarrow \text{argmax}_{A \in \mathcal{F}} \hat{\mu}^{(r-1)}(A)$;
    if Check$(\hat{O}, \hat{\mu}^{(k)}, \hat{\mu}, \theta_k, \epsilon_k/\lambda)$ for all $k \in [r-1]$ then
      return $\hat{O}$;
    else
      return error;
  $\epsilon_r \leftarrow 2^{-r}$; $\delta_r \leftarrow \delta_0/(10r^3|\mathcal{F}|^2)$;
  $m^{(r)} \leftarrow \sum_{k=1}^{r} \text{SimultEst}(\hat{\mu}^{(k-1)}, \theta_{k-1} - \epsilon_{k-1}/\lambda, \theta_{k-1} - 2\epsilon_{k-1}/\lambda, \epsilon_k/\lambda, \delta_r)$;
  $\hat{\mu}^{(r)} \leftarrow \text{Sample}(m^{(r)})$;
  $\text{opt}_r \leftarrow \text{OPT}(\hat{\mu}^{(r-1)}, \theta_{r-1}, \hat{\mu}^{(r)}, \epsilon_{r-1}/\lambda)$;
  $\theta_r \leftarrow \text{opt}_r - (1/2 + 2/\lambda)\epsilon_r$;

5.2. Speci cation

We formally state the performance guarantees of the subroutines in EfficientGapElim, which are crucial to the analysis of the algorithm. In Section 5.4 we discuss implementations that meet these speci cations.

1. Given weights μ and threshold θ, Unique(μ, θ) correctly decides whether there is exactly one set $A \in \mathcal{F}$ such that $\mu(A) \geq \theta$.

2. Both SimultEst and Verify return *feasible* solutions to the programs de ned in the procedures. Moreover, the resulting objective function should be at most a constant times the optimal value of the tightened programs obtained by replacing θ^{high} with θ^{low} (or replacing $\{\theta_k^{\text{high}}\}$ with $\{\theta_k^{\text{low}}\}$) in the constraints.

3. Given empirical means μ, threshold θ, weight w, and accuracy level ϵ, OPT$(\mu, \theta, w, \epsilon)$ returns a set $A \in \mathcal{F}$ such that: (a) $\mu(A) \geq \theta - \epsilon$ (i.e., A is approximately feasible); (b) $w(A) \geq \max_{B \in \mathcal{F}, \mu(B) \geq \theta} w(B) - \epsilon$ (i.e., A is approximately optimal).

4. When Check$(\hat{O}, \hat{\mu}^{(k)}, \hat{\mu}, \theta, \epsilon)$ is called, and it holds that $\hat{\mu}(\hat{O}) - \hat{\mu}(A) \geq 2\epsilon$ for all $A \in \mathcal{F}$ such that $\hat{\mu}^{(k)}(A) < \theta$, the procedure returns \true". If $\hat{\mu}(\hat{O}) - \hat{\mu}(A) \leq \epsilon$ for *some*

$A \in \mathcal{F}$ such that $\hat{\mu}^{(k)}(A) < \theta - \epsilon$, the procedure always returns \false". In other cases, the procedure may return arbitrarily.

5.3. Analysis of **EfficientGapElim**

We state the performance guarantees of algorithm EfficientGapElim in the following two lemmas. The proofs are essentially identical to those in Sections 4.3 and 4.4, and are therefore postponed to Appendix B.

Lemma 5.1 *For any* $\delta \in (0, 0.01)$ *and* Best-Set *instance* $\mathcal{C}$, *EfficientGapElim*$(\mathcal{C}, \delta)$ *returns the correct answer with probability* $1 - \delta_0 - \delta$, *and returns an incorrect answer w.p. at most* δ.

Recall that $= \mu(O) - \max_{A \in \mathcal{F} \setminus \{O\}} \mu(A)$ is the gap between the set with the second largest weight in $\mathcal{F}$ and the weight of O.

Lemma 5.2 *For any* $\delta \in (0, 0.01)$ *and* Best-Set *instance* $\mathcal{C}$, *EfficientGapElim*$(\mathcal{C}, \delta)$ *takes*

$$O\left(\mathit{Low}(\mathcal{C}) \ln \delta^{-1} + \mathit{Low}(\mathcal{C}) \ln^2 \quad ^{-1} \left(\ln \ln \quad ^{-1} + \ln |\mathcal{F}|\right)\right)$$

samples conditioning on event $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$.

Lemmas 5.1, 5.2 and 4.8 imply that there is a δ-correct algorithm that matches the sample complexity stated in Theorem 1.12. It remains to implement the subroutines speci ed in Section 5.2 e ciently.

5.4. E cient Computation via ε-approximate Pareto Curve

In this section, we propose a general framework for e ciently implementing the subroutines speci ed in Section 5.2, thus proving Theorem 1.12. Here, by \e cient", we mean the time complexity of the algorithm is bounded by a function polynomial both in n and the *sample complexity* of the algorithm. Indeed, for any natural algorithm, the *time complexity* is at least the same as the *sample complexity*. We use the concept of *ε-approximate Pareto curve*, a general framework for multi-objective optimization, which was rst introduced by Papadimitriou and Yannakakis (2000).

In this section, we only need *bi-objective optimization problems*, i.e., problems with two objective functions. For a bi-objective optimization problem, for each instance x, we denote $F(x)$ to be its feasible solution space. For each feasible solution $s \in F(x)$, two objective functions $f_1(x, s)$ and $f_2(x, s)$ will be used to evaluate the *quality* of the solution s. The goal here is to *maximize* the objective functions. Meanwhile, as shown in Papadimitriou and Yannakakis (2000), minimization problems can be treated similarly.

The *Pareto curve* of an instance x, denoted by $P(x)$, is a set of 2-dimension points. For each $v \in P(x)$,

(1) There exists some $s \in F(x)$ such that $f_i(x, s) = v_i$, for $i = 1$ and $i = 2$.

(2) There is no feasible solution s' such that $f_i(x, s) \geq v_i$ for $i = 1$ and $i = 2$, with at least one inequality holding strictly.

The Pareto curve naturally provides a trade-o between the two objective functions. However, the Pareto curve is exponentially large in size in general and cannot be e ciently computed. Thus, Papadimitriou and Yannakakis (2000) considered the approximate version of Pareto curves. The ε-*approximate Pareto curve* of an instance x, denoted by $P_\varepsilon(x)$, is a set of feasible solutions, such that for each feasible solution $s' \in F(x)$, there exists some $s \in P_\varepsilon(x)$ such that $f_i(x,s') \leq (1+\varepsilon) f_i(x,s)$ for $i = 1$ and $i = 2$. For a problem A where the objective functions are linear, Papadimitriou and Yannakakis (2000) give an FPTAS for constructing the approximate Pareto curve, given a pseudopolynomial algorithm for the *exact version* of A. The exact version of A is one where, given an instance x and a value B, we have to decide if there exists a feasible solution with cost *exactly* B.

Many combinatorial problems admit pseudopolynomial algorithms for the exact version, including the shortest path problem, the minimum spanning tree problem and the matching problem, as noted in Papadimitriou and Yannakakis (2000). In the following sections, we will show how to e ciently implement the algorithm descried in previous sections, when the approximate Pareto curve of the underlying combinatorial problem of the Best-Set instance can be computed by an FPTAS. We also assume that the single-objective maximization version of the underlying combinatorial problem can be solved in polynomial time, i.e., given a weight vector w, there exists an algorithm that runs in polynomial time that can calculate $\text{argmax}_{A \in \mathcal{F}} w(A)$. [9]

5.4.1. Efficient Implementation of OPT, Check and Unique

We begin with the implementation of OPT. Notice that OPT is actually a bi-objective optimization problem, by setting $f_1(\cdot)$ to be $\mu(\cdot)$ and $f_2(\cdot)$ to be $w(\cdot)$. We can e ciently implement OPT by listing all points in the approximate Pareto curve. Notice that it is required that OPT outputs a solution with *additive* approximation term, while the FPTAS presented in Papadimitriou and Yannakakis (2000) can only be used to generate approximate Pareto curve with multiplicative approximation ratio. An important observation is that, for any $S \in \mathcal{F}$, $\mu(S)$ is bounded by $O(n)$ and $w(S)$ is bounded by a function polynomial in the sample complexity of our algorithm. Thus, to calculate an additive ε-approximate Pareto curve, it su ces to set ε' to be a value polynomial in n, ε and the sample complexity of our algorithm, and calculate the multiplicative ε'-approximate Pareto curve.

Similarly, Check is also a bi-objective optimization problem, by setting $f_1(\cdot)$ to be $\mu^{(k)}(\cdot)$ and $f_2(\cdot)$ to be $\mu(\cdot)$. We can still e ciently implement Check by listing all points in the approximate Pareto curve. We omit implementation details due to the similarity.

Given a polynomial-time algorithm $\mathbb{A}$ for the single-objective maximization version of the underlying combinatorial problem, it will be straightforward to implement Unique. One possible way is to calculate the subset with second largest objective value, which is given as follows. We rst call $\mathbb{A}$ to nd a subset A with maximum $\mu(A)$. Then we enumerate every element $a \in A$, set $\mu(a)$ to $-\infty$ and call $\mathbb{A}$ again. By doing so, we will be able to nd the subset A' with second largest objective value. We can then decide whether there is exactly one subset A such that $\mu(A) \geq \theta$ by comparing $\mu(A')$ with θ.

9. Such an algorithm has already been used implicitly in Line 7 of algorithm EfficientGapElim.

5.4.2. Efficient Implementation of SimultEst and Verify

Now we present our implementation for SimultEst. To solve the convex program described in Algorithm 4, we apply the *Ellipsoid method*. It suffices to devise a polynomial time *separation oracle* [10] (see e.g., Schrijver (2002)). Concretely, we need to solve the following separation problem.

Definition 5.3 (Separation problem of SimultEst) *Given* $(\mu, \theta^{\mathsf{high}}, \theta^{\mathsf{low}}, \varepsilon, \delta)$ *and vector* $m^\star$*, the goal of the separation problem of* SimultEst$(\mu, \theta^{\mathsf{high}}, \theta^{\mathsf{low}}, \varepsilon, \delta)$ *is to decide whether there exists two subsets* $A, B \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\mathsf{high}}\}$ *such that*

$$\sum_{i \in A \triangle B} \frac{1}{m_i^\star} \geq \frac{\varepsilon^2}{2\ln(2/\delta)}.$$

Notice that we do not need to solve the separation problem exactly: a constant approximation suffices, as this would only increase a constant factor hidden in the big-O notation of the sample complexity. (This trick is often used in the approximation algorithms literature; see, e.g., Carr et al. (2000).) Specifically, it is sufficient to find two subsets $A, B \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\mathsf{high}}\}$ such that

$$\sum_{i \in A \triangle B} \frac{1}{m_i^\star} \geq C \cdot \frac{\varepsilon^2}{2\ln(2/\delta)}$$

for some constant C (assuming there are subsets A', B' satisfying $\sum_{i \in A' \triangle B'} \frac{1}{m_i^\star} \geq \frac{\varepsilon^2}{2\ln(2/\delta)}$). Moreover, as noted in the previous section, the constraint $A, B \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\mathsf{high}}\}$ can also be relaxed to allow an additive approximate term of $\theta^{\mathsf{high}} - \theta^{\mathsf{low}}$.

To decide whether such a pair of subset (A, B) exists or not, we first arbitrarily choose a subset O in $\{A' \in \mathcal{F} : \mu(A') \geq \theta^{\mathsf{high}}\}$ and then find a subset $O' \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\mathsf{high}}\}$ such that

$$\sum_{i \in O \triangle O'} \frac{1}{m_i^\star}.$$

is maximized. The following lemma shows that, (O, O') is a 2-approximation of the original separation problem.

Lemma 5.4 *For any* $A, B \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\mathsf{high}}\}$,

$$\sum_{i \in O \triangle O'} \frac{1}{m_i^\star} \geq \frac{1}{2} \sum_{i \in A \triangle B} \frac{1}{m_i^\star}.$$

Proof [Proof of Lemma 5.4] As O' is chosen so that

$$\sum_{i \in O \triangle O'} \frac{1}{m_i^\star}.$$

10. Given a point x, the separation oracle needs to decide whether x is in the feasible region. If not, the separation oracle should output a constraint that x violates.

is maximized, it follows that

$$\sum_{i\in O\triangle O'} \frac{1}{m_i^\star} \geq \sum_{i\in O\triangle A} \frac{1}{m_i^\star}$$

and

$$\sum_{i\in O\triangle O'} \frac{1}{m_i^\star} \geq \sum_{i\in O\triangle B} \frac{1}{m_i^\star}.$$

Thus,

$$2\sum_{i\in O\triangle O'} \frac{1}{m_i^\star} \geq \sum_{i\in O\triangle A} \frac{1}{m_i^\star} + \sum_{i\in O\triangle B} \frac{1}{m_i^\star} \geq \sum_{i\in A\triangle B} \frac{1}{m_i^\star}.$$

■

Now it remains to show how to find O' efficiently. In order to find O', we find $O_1 \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ such that

$$\sum_{i\in O\setminus O_1} \frac{1}{m_i^\star}$$

is maximized, and $O_2 \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ such that

$$\sum_{i\in O_2\setminus O} \frac{1}{m_i^\star}$$

is maximized. Again, the following lemma shows that, by using the method described above, we can get a 2-approximation.

Lemma 5.5 *For any* $O' \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$,

$$2\max\left\{\sum_{i\in O\setminus O_1} \frac{1}{m_i^\star}, \sum_{i\in O_2\setminus O} \frac{1}{m_i^\star}\right\} \geq \sum_{i\in O\triangle O'} \frac{1}{m_i^\star}.$$

Proof [Proof of Lemma 5.5]

$$2\max\left\{\sum_{i\in O\setminus O_1} \frac{1}{m_i^\star}, \sum_{i\in O_2\setminus O} \frac{1}{m_i^\star}\right\} \geq \sum_{i\in O\setminus O_1} \frac{1}{m_i^\star} + \sum_{i\in O_2\setminus O} \frac{1}{m_i^\star}.$$

According to our choice of O_1 and O_2, it follows that

$$\sum_{i\in O\setminus O_1} \frac{1}{m_i^\star} + \sum_{i\in O_2\setminus O} \frac{1}{m_i^\star} \geq \sum_{i\in O\setminus O'} \frac{1}{m_i^\star} + \sum_{i\in O'\setminus O} \frac{1}{m_i^\star} = \sum_{i\in O\triangle O'} \frac{1}{m_i^\star}.$$

■

The analysis above suggests, to decide whether there exists two subsets $A, B \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ such that

$$\sum_{i\in A\triangle B} \frac{1}{m_i^\star} \geq \frac{\varepsilon^2}{2\ln(2/\delta)}$$

approximately, it suffices to decide whether exists $O_1, O_2 \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ such that

$$\sum_{i \in O \setminus O_1} \frac{1}{m_i^\star} \geq \frac{\varepsilon^2}{2\ln(2/\delta)}$$

or

$$\sum_{i \in O_2 \setminus O} \frac{1}{m_i^\star} \geq \frac{\varepsilon^2}{2\ln(2/\delta)}.$$

The problem of finding O_1 and O_2, are actually bi-objective optimization problems. As mentioned in previous sections, the first constraint, i.e., $O_1, O_2 \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ can be relaxed to allow an additive approximate term of $\theta^{\text{high}} - \theta^{\text{low}}$, where $1/(\theta^{\text{high}} - \theta^{\text{low}})$ is bounded by the sample complexity of our algorithm. Thus, by using the approximate Pareto curve, it is straightforward to decide whether such O_2 exists or not. We set w_i to be $\frac{1}{m_i^\star}$, and further, for any $i \in O$, we set w_i to be zero. We can then decide whether O_2 exists or not by calling OPT and using w as the weight vector.

Deciding whether O_1 exists or not is more involved, but still in a similar manner. Again, our plan is to approximately decide the existence of such O_1. More specifically, our method will return \yes" when there exists O_1 such that

$$\sum_{i \in O \setminus O_1} \frac{1}{m_i^\star} \geq \frac{2\varepsilon^2}{2\ln(2/\delta)},$$

return \no" when for any $O_1 \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$,

$$\sum_{i \in O \setminus O_1} \frac{1}{m_i^\star} \leq \frac{\varepsilon^2}{2\ln(2/\delta)}$$

and return arbitrarily otherwise.

When

$$\sum_{i \in O} \frac{1}{m_i^\star} < \frac{2\varepsilon^2}{2\ln(2/\delta)}$$

we simply return \no", as there will be no O_1 such that

$$\sum_{i \in O \setminus O_1} \frac{1}{m_i^\star} \geq \frac{2\varepsilon^2}{2\ln(2/\delta)}.$$

Otherwise, we set w_i to be $\frac{1}{m_i^\star}$, and further, for any $i \notin O$, we set w_i to be zero. Then we use approximate Pareto curve to find $O_1 \in \{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ with approximately maximum $w(O_1)$.

Here, we use set the multiplicative approximation ratio to be

$$1 + \frac{\varepsilon^2}{2\ln(2/\delta)w(O)},$$

as such a multiplicative approximation ratio will induce an additive approximate term of

$$\frac{\varepsilon^2}{2\ln(2/\delta)w(O)} \cdot w(O_1^*) \leq \frac{\varepsilon^2}{2\ln(2/\delta)w(O)} \cdot w(O) = \frac{\varepsilon^2}{2\ln(2/\delta)},$$

where O_1^* denotes the subset in $\{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ with maximum

$$\sum_{i \in O \setminus O_1^*} \frac{1}{m_i^\star}.$$

Such an additive approximate term is enough to distinguish the two cases (return \yes" or \no") stated above. Meanwhile, the time complexity for calculating such an approximate Pareto curve is bounded by a function polynomial in the sample complexity of our algorithm.

Given the e cient implementation for SimultEst, Verify can be implemented in a similar manner. We also apply the Ellipsoid method and approximately solve the separation problem by using approximate Pareto curve. We do not repeat those details due to the similarity.

6. Optimal Algorithm for General Sampling Problem

In this section, we present a nearly optimal algorithm LPSample for the General-Samp problem. Given an instance $\mathcal{I} = (S, \mathcal{O})$ and a con dence level δ, LPSample either identi es an answer set in $\mathcal{O}$ as the answer, or reports an error. The algorithm is guaranteed to return the correct answer with probability $1 - \delta - \delta_0$, where $\delta_0 = 0.01$, while the probability of returning an incorrect answer is upper bounded by δ. Therefore, LPSample can be transformed to a δ-correct algorithm while retaining its sample complexity by applying the parallel simulation idea from Lemma 4.8.

6.1. Algorithm

Algorithm LPSample consists of two stages. In the rst stage, we sample each arm repeatedly in round-robin fashion, until the con dence region of μ intersects exactly one answer set $\hat{O} \in \mathcal{O}$. [11] We identify $\hat{O}$ as the candidate answer. We further sample each arm a few more times in order to obtain a su ciently tight con dence region for the second stage.

The second stage is devoted to verifying the candidate $\hat{O}$. We rst calculate the optimal sampling pro le by linear programming. Let $\text{Alt}(\hat{O})$ denote $\bigcup_{O \in \mathcal{O} \setminus \{\hat{O}\}} O$, the union of all answer sets other than $\hat{O}$. Each point $\nu \in \text{Alt}(\hat{O})$ de nes a constraint of the linear program, which states that su ciently many samples must be taken, in order to distinguish the actual mean pro le from ν. Finally, we verify the candidate answer by sampling the arms according to the sampling pro le.

Note that in the rst stage, LPSample samples the arms in an ine cient round-robin fashion, while the candidate answer $\hat{O}$ is verifed using the optimal sampling pro le in Stage 2. Thus, LPSample uses a less stringent con dence (i.e., δ_0) in Stage 1, and then adopts the required con dence level δ in the second stage.

11. Here $B(x, r)$ denotes the closed ℓ^2-ball $\{x' \in \mathbb{R}^n : \|x - x'\|_2 \leq r\}$.

Algorithm 7: LPSample($\mathcal{I}, \delta$)

Input: Instance $\mathcal{I} = (S, \mathcal{O})$ and confidence level δ.

Output: Either an answer set in $\mathcal{O}$ or an error.

$t \leftarrow 0, \delta_0 \leftarrow 0.01$;

repeat

  $t \leftarrow t + 1$;

  Sample each arm in S once;

  $\hat{\mu}^{(t)} \leftarrow$ empirical means of the arms among the first t samples;

  $r_t \leftarrow \sqrt{\left[2n + 3\ln\left(\delta_0/(4t^2)\right)^{-1}\right]/t}$;

until $B\left(\hat{\mu}^{(t)}, 3r_t\right)$ *intersects with exactly one of the answer sets, denoted by* $\hat{O}$;

$\alpha \leftarrow r_t/\sqrt{8n}$; $M \leftarrow \alpha^{-2}\left[2n + 3\ln(\delta_0/2)^{-1}\right]$;

$\hat{\mu} \leftarrow$ Sample($(M, M, \ldots, M)$);

if $B(\hat{\mu}, r_t)$ *intersects with* Alt($\hat{O}$) then

  return error;

Let x^* be the optimal solution to the following linear program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} x_i \\ \text{subject to} \quad & \sum_{i=1}^{n} (\nu_i - \hat{\mu}_i)^2 x_i \geq 1, \ \forall \nu \in \text{Alt}(\hat{O}), \\ & x_i \geq 0. \end{aligned} \tag{6}$$

$\beta \leftarrow 64$; $m \leftarrow \beta x_i^* \left(\ln \delta^{-1} + n\right)$;

$X \leftarrow$ Sample(m);

if $\sum_{i=1}^{n} m_i (X_i - \hat{\mu}_i)^2 \leq 36\left(\ln \delta^{-1} + n\right)$ then

  return $\hat{O}$;

else

  return error;

6.2. Correctness

Good Events. We start by de ning two \good events" conditioning on which the correctness and the sample complexity optimality of LPSample can be guaranteed. Recall that μ_i denote the mean of arm A_i. De ne $\mathcal{E}_0^{\text{good}}$ as the event that in Stage 1, $\left\|\hat{\mu}^{(k)} - \mu\right\|_2 \le r_k$ holds for all k, and $\|\hat{\mu} - \mu\|_2 \le \alpha$ also holds. Note that both $k\left\|\hat{\mu}^{(k)} - \mu\right\|_2^2$ and $M\|\hat{\mu} - \mu\|_2^2$ are χ^2 random variables with n degrees of freedom. The tail probability bound of the χ^2-distribution (Lemma 2.2) implies that

$$\Pr\left[\left\|\hat{\mu}^{(k)} - \mu\right\|_2 > r_k\right] = \Pr\left[k\left\|\hat{\mu}^{(k)} - \mu\right\|_2^2 > 2n + 3\ln\left(\frac{\delta_0}{4k^2}\right)^{-1}\right] \le \frac{\delta_0}{4k^2}.$$

Similarly,

$$\Pr\left[\|\hat{\mu} - \mu\|_2 > \alpha\right] = \Pr\left[M\|\hat{\mu} - \mu\|_2^2 > 2n + 3\ln\left(\frac{\delta_0}{2}\right)^{-1}\right] \le \frac{\delta_0}{2}.$$

By a union bound,

$$\Pr[\mathcal{E}_0^{\text{good}}] \ge 1 - \frac{\delta_0}{2} - \sum_{k=1}^{\infty}\frac{\delta_0}{4k^2} \ge 1 - \delta_0.$$

We de ne $\mathcal{E}^{\text{good}}$ as the event that in Stage 2, it holds that

$$\sum_{i=1}^{n} m_i(X_i - \mu_i)^2 \le 2n + 3\ln\delta^{-1}. \tag{7}$$

Note that $\sqrt{m_i}(X_i - \mu_i)$ follows the standard normal distribution. Thus, Lemma 2.2 implies that that $\Pr[\mathcal{E}^{\text{good}}] \ge 1 - \delta$.

LP Solution Bound. We have the following simple lemma, which upper bounds the optimal solution x^* of the linear program in Stage 2.

Lemma 6.1 $\sum_{i=1}^{n} x_i^* \le n r_t^{-2}$.

Proof Since LPSample completes Stage 1 without reporting an error, $B(\hat{\mu}, r_t)$ is disjoint from $\text{Alt}(\hat{O})$. In other words, for all $\nu \in \text{Alt}(\hat{O})$ we have $\|\nu - \hat{\mu}\|_2 > r_t$. It directly follows that

$$x_1 = x_2 = \cdots = x_n = r_t^{-2}$$

is a feasible solution of the linear program (6), which proves the lemma. ■

Lemma 6.2 (Soundness) *Conditioning on $\mathcal{E}^{\text{good}}$, LPSample never returns an incorrect answer.*

Proof Recall that at the end of algorithm LPSample, the following inequality is veri ed:

$$\sum_{i=1}^{n} m_i(X_i - \hat{\mu}_i)^2 \le 36\left(\ln\delta^{-1} + n\right). \tag{8}$$

Suppose the candidate answer $\hat{O}$ chosen in Stage 1 is correct, the lemma trivially holds, so assume that the candidate is incorrect (i.e., $\mu \in \mathrm{Alt}(\hat{O})$). We now show that conditioning on event $\mathcal{E}^{\mathrm{good}}$, inequality (8) is violated, and thus LPSample reports an error, rather than returning the incorrect answer $\hat{O}$.

Define $a_i = \sqrt{m_i}\,(X_i - \hat{\mu}_i)$. Note that inequality (8) is equivalent to $\|a\|_2 \le 6\sqrt{n + \ln \delta^{-1}}$. Let us write a into $a = b + c$, where $b_i = \sqrt{m_i}\,(X_i - \mu_i)$ and $c_i = \sqrt{m_i}\,(\mu_i - \hat{\mu}_i)$. We first note that conditioning on event $\mathcal{E}^{\mathrm{good}}$, inequality (7) guarantees that

$$\|b\|_2 = \sqrt{\sum_{i=1}^{n} m_i\,(X_i - \mu_i)^2} \le \sqrt{2n + 3\ln \delta^{-1}} < 2\sqrt{n + \ln \delta^{-1}}.$$

On the other hand, since $\mu \in \mathrm{Alt}(\hat{O})$, the constraint corresponding to point μ in linear program (6) implies that

$$\|c\|_2 = \sqrt{\sum_{i=1}^{n} m_i\,(\mu_i - \hat{\mu}_i)^2} = \sqrt{\beta\,(n + \ln \delta^{-1}) \sum_{i=1}^{n} x_i^*\,(\mu_i - \hat{\mu}_i)^2} \ge 8\sqrt{n + \ln \delta^{-1}}.$$

Therefore, we conclude that

$$\|a\|_2 = \|b + c\|_2 \ge \|c\|_2 - \|b\|_2 > 6\sqrt{n + \ln \delta^{-1}},$$

which completes the proof. ■

Lemma 6.3 (Completeness) *Conditioning on $\mathcal{E}_0^{\mathrm{good}} \cap \mathcal{E}^{\mathrm{good}}$, LPSample always returns the correct answer.*

Proof Recall that conditioning on event $\mathcal{E}_0^{\mathrm{good}}$, the actual mean profile μ is in $B(\hat{\mu}^{(t)}, r_t)$. According to LPSample, $\hat{O}$ is the only answer set that intersects $B(\hat{\mu}^{(t)}, r_t)$, and thus $\hat{O}$ is indeed the correct answer. It remains to show that LPSample terminates without reporting errors.

We first prove that at the end of Stage 1, $B(\hat{\mu}, r_t)$ and $\mathrm{Alt}(\hat{O})$ are disjoint. Let ν be an arbitrary point in $\mathrm{Alt}(\hat{O})$. Our choice of t ensures that $\left\|\hat{\mu}^{(t)} - \nu\right\|_2 \ge 3r_t$. Conditioning on event $\mathcal{E}_0^{\mathrm{good}}$, we also have $\left\|\hat{\mu}^{(t)} - \mu\right\|_2 \le r_t$ and $\|\hat{\mu} - \mu\|_2 \le \alpha < r_t$. It follows from the three inequalities above that

$$\|\hat{\mu} - \nu\|_2 \ge \left\|\hat{\mu}^{(t)} - \nu\right\|_2 - \left\|\hat{\mu}^{(t)} - \mu\right\|_2 - \|\hat{\mu} - \mu\|_2 > 3r_t - r_t - r_t = r_t,$$

which implies $\nu \notin B(\hat{\mu}, r_t)$. Therefore, $B(\hat{\mu}, r_t)$ and $\mathrm{Alt}(\hat{O})$ are disjoint, and LPSample finishes Stage 1 without reporting an error.

Next we show that LPSample does not report an error at the end of Stage 2 (i.e., inequality (8) holds). As in the proof of Lemma 6.2, define $a_i = \sqrt{m_i}\,(X_i - \hat{\mu}_i)$, $b_i = \sqrt{m_i}\,(X_i - \mu_i)$, and $c_i = \sqrt{m_i}\,(\mu_i - \hat{\mu}_i)$. Then inequality (8) is equivalent to showing $\|a\|_2 = \|b + c\|_2 \le$

$6\sqrt{n + \ln \delta^{-1}}$. Conditioning on $\mathcal{E}^{\text{good}}$, $\|b\|_2 < 2\sqrt{n + \ln \delta^{-1}}$ follows from inequality (7) as in Lemma 6.2. Next,

$$\begin{aligned}
\|c\|_2^2 &= \beta\left(n + \ln \delta^{-1}\right) \sum_{i=1}^{n} x_i^* \left(\mu_i - \hat{\mu}_i\right)^2 && \text{(by definition of } c \text{ and } m_i\text{)} \\
&\leq 64\left(n + \ln \delta^{-1}\right) \left(\sum_{i=1}^{n} x_i^*\right) \sum_{i=1}^{n} \left(\mu_i - \hat{\mu}_i\right)^2 && (\boldsymbol{u} \cdot \boldsymbol{v} \leq \|\boldsymbol{u}\|_1 \cdot \|\boldsymbol{v}\|_1) \\
&\leq 64\left(n + \ln \delta^{-1}\right) \cdot n r_t^{-2} \cdot \alpha^2 \\
&\leq 8\left(n + \ln \delta^{-1}\right).
\end{aligned}$$

Above, the third step applies Lemma 6.1 and the fact that $\|\hat{\mu} - \mu\|_2 \leq \alpha$. The last step plugs in the parameter $\alpha = r_t/\sqrt{8n}$. Therefore, we conclude that

$$\|a\|_2 \leq \|b\|_2 + \|c\|_2 < 2\sqrt{n + \ln \delta^{-1}} + \sqrt{8(n + \ln \delta^{-1})} < 6\sqrt{n + \ln \delta^{-1}},$$

and thus LPSample returns the correct answer without reporting an error. ■

6.3. Sample Complexity

We show that LPSample is nearly optimal: the sample complexity of the algorithm matches the instance lower bound (Low($\mathcal{I}$) $\ln \delta^{-1}$) as δ tends to zero. Specifically, we give an upper bound on the sample complexity of algorithm LPSample conditioning on the "good event" $\mathcal{E}_0^{\text{good}}$, in terms of Low($\mathcal{I}$) and

$$= \inf_{\nu \in \text{Alt}(O)} \|\mu - \nu\|_2 .$$

(Note that the assumption that O is disjoint from the closure of Alt(O) guarantees that > 0.) We first prove a simple lemma, which relates to Low($\mathcal{I}$).

Lemma 6.4 *Low*($\mathcal{I}$) $\geq$ $^{-2}$.

Proof By definition, Low($\mathcal{I}$) = $\sum_{i=1}^{n} \tau_i^*$, where $\{\tau_i^*\}$ is the optimal solution to (3). Note that

$$\text{Low}(\mathcal{I}) \|\nu_i - \mu_i\|_2^2 \geq \sum_{i=1}^{n} (\nu_i - \mu_i)^2 \tau_i^* \geq 1.$$

Thus

$$\text{Low}(\mathcal{I}) \geq \sup_{\nu \in \text{Alt}(O)} \|\nu - \mu\|_2^{-2} = \quad ^{-2}.$$

■

Lemma 6.5 *Conditioning on event* $\mathcal{E}_0^{\text{good}}$, *LPSample takes* $O($*Low*$(\mathcal{I})(\ln \delta^{-1} + n^3 + n \ln \quad ^{-1}))$ *samples.*

Proof Recall that in the first stage of LPSample, r_k is defined as

$$r_k = \sqrt{\frac{2n + 3\ln[\delta_0/(4k^2)]^{-1}}{k}},$$

and the number of samples taken in Stage 1 is $nt + nM$, where

$$M = \alpha^{-2}\left[2n + 3\ln(\delta_0/2)^{-1}\right] = O(n\alpha^{-2}) = O(n^2 r_t^{-2}),$$

and t is the smallest index such that $B(\hat{\mu}^{(t)}, 3r_t)$ intersects only one set in $\mathcal{O}$. In order to upper bound t, let t^* be the smallest integer such that $r_{t^*} < \quad /4$. A simple calculation gives $t^* = O\left(\quad^{-2}(n + \ln \quad^{-1})\right)$. Moreover, at round t^*, conditioning on event $\mathcal{E}_0^{\text{good}}$ implies that $\hat{\mu}^{(t^*)} \in B(\mu, r_{t^*})$. It follows that

$$B(\hat{\mu}^{(t^*)}, 3r_{t^*}) \subseteq B(\mu, 4r_{t^*}) \subset B(\mu, \quad).$$

By definition of $\quad$, $B(\hat{\mu}^{(t^*)}, 3r_{t^*})$ is disjoint from $\mathrm{Alt}(O)$, and thus the round-robin sampling in Stage 1 terminates before taking t^* samples from each arm (i.e., $t \le t^*$). Therefore, nt is upper bounded by

$$nt^* = O\left(\quad^{-2}(n^2 + n\ln \quad^{-1})\right).$$

Moreover, we have $nM = O(n^3 r_t^{-2}) = O(n^3 r_{t^*}^{-2}) = O(n^3 \quad^{-2})$. Also, we note that $\quad^{-2} = O(\mathrm{Low}(\mathcal{I}))$ by Lemma 6.4. Putting this all together, the number of samples in Stage 1 is

$$O\left(\mathrm{Low}(\mathcal{I})(n^3 + \ln \quad^{-1})\right).$$

Now for the second stage samples. Let τ^* be the optimal solution to the linear program defined in (3). By definition, $\mathrm{Low}(\mathcal{I}) = \sum_{i=1}^n \tau_i^*$. Then we construct a feasible solution to the linear program in Stage 2 from τ^*. Recall that conditioning on event $\mathcal{E}_0^{\text{good}}$, we have $|\hat{\mu}_i - \mu_i| \le \|\hat{\mu} - \mu\|_2 \le \alpha$ for all $i \in [n]$. It follows that for all $\nu \in \mathrm{Alt}(\hat{O})$,

$$(\nu_i - \hat{\mu}_i)^2 = [(\nu_i - \mu_i) + (\mu_i - \hat{\mu}_i)]^2 \ge (\nu_i - \mu_i)^2/2 - 2(\mu_i - \hat{\mu}_i)^2 \ge (\nu_i - \mu_i)^2/2 - 2\alpha^2.$$

Here the second step applies the inequality $(a + b)^2 \ge a^2/2 - 2b^2$. Therefore, for all $\nu \in \mathrm{Alt}(\hat{O})$,

$$\begin{aligned}\sum_{i=1}^n (\nu_i - \hat{\mu}_i)^2 \tau_i^* &\ge \sum_{i=1}^n \tau_i^* \left[(\nu_i - \mu_i)^2/2 - 2\alpha^2\right] \\ &\ge \frac{1}{2}\sum_{i=1}^n (\nu_i - \mu_i)^2 \tau_i^* - 2\alpha^2 \sum_{i=1}^n \tau_i^* \\ &\ge \frac{1}{2} - 2\alpha^2 \cdot n r_t^{-2} = \frac{1}{4}.\end{aligned} \tag{9}$$

The third step holds due to the feasibility of τ^* and the fact that $\sum_{i=1}^n \tau_i^* \le n r_t^{-2}$, which follows from an analogous argument to the proof of Lemma 6.1. The last step follows from our choice of parameter $\alpha = r_t/\sqrt{8n}$.

Inequality (9) implies that $x_i = 4\tau_i^*$ is a feasible solution of the linear program in Stage 2 of LPSample. It follows that the number of samples taken in Stage 2 is bounded by

$$\sum_{i=1}^n m_i = \beta\left(\ln \delta^{-1} + n\right)\sum_{i=1}^n x_i^* \le \beta\left(\ln \delta^{-1} + n\right)\sum_{i=1}^n 4\tau_i^* = O\left(\mathrm{Low}(\mathcal{I})\left(\ln \delta^{-1} + n\right)\right).$$

In conclusion, LPSample takes

$$O\left(\mathsf{Low}(\mathcal{I})(\ln \delta^{-1} + n^3 + n \ln \quad ^{-1})\right)$$

samples in Stage 1 and Stage 2 in total, conditioning on event $\mathcal{E}_0^{\text{good}}$. ■

Finally, we prove Theorem 1.13, which we restate for convenience in the following.

Theorem 1.13 (restated) *There is a δ-correct algorithm for* General-Samp *that takes*

$$O(\mathit{Low}(\mathcal{I})(\ln \delta^{-1} + n^3 + n \ln \quad ^{-1}))$$

samples on any instance $\mathcal{I} = (S, \mathcal{O})$ in expectation, where

$$= \inf_{\nu \in \mathsf{Alt}(O)} \|\mu - \nu\|_2$$

is defined as the minimum Euclidean distance between the mean profile μ and an alternative mean profile $\nu \in \mathsf{Alt}(O)$ with an answer other than O.

Proof By Lemmas 6.2, 6.3 and 6.5, LPSample is a (δ_0, δ, A, B)-correct algorithm for General-Samp (as per De nition 4.7), where $\mathcal{E}_0 = \mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$, $\mathcal{E}_1 = \mathcal{E}^{\text{good}}$, $\delta_0 = 0.01$, $A = \mathsf{Low}(\mathcal{I})$ and $B = \mathsf{Low}(\mathcal{I})(n^3 + n \ln \quad ^{-1})$. Lemma 4.8 implies that there is a δ-correct algorithm for General-Samp with expected sample complexity

$$O(A \ln \delta^{-1} + B) = O\left(\mathsf{Low}(\mathcal{I})(\ln \delta^{-1} + n^3 + n \ln \quad ^{-1})\right).$$

■

References

Jean-Yves Audibert and Sebastien Bubeck. Best arm identi cation in multi-armed bandits. In *COLT*, 2010.

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

Sebastien Bubeck, Tengyao Wang, and Nitin Viswanathan. Multiple identi cations in multi-armed bandits. *arXiv preprint arXiv:1205.3181*, 2012.

Wei Cao, Jian Li, Yufei Tao, and Zhize Li. On top-k selection in multi-armed bandits and hidden bipartite graphs. In *NIPS*, pages 1036{1044, 2015.

Alexandra Carpentier and Andrea Locatelli. Tight (lower) bounds for the xed budget best arm identi cation bandit problem. In *Proceedings of the 29th Conference on Learning Theory*, 2016.

Robert D Carr, Lisa Fleischer, Vitus J Leung, and Cynthia A Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *SODA*, pages 106{115, 2000.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Lijie Chen and Jian Li. On the optimal sample complexity for best arm identi cation. *arXiv preprint arXiv:1511.03774*, 2015.

Lijie Chen and Jian Li. Open problem: Best arm identi cation: Almost instance-wise optimality and the gap entropy conjecture. In *29th Annual Conference on Learning Theory*, pages 1643{1646, 2016.

Lijie Chen, Anupam Gupta, and Jian Li. Pure exploration of multi-armed bandit under matroid constraints. In *COLT*, 2016a.

Lijie Chen, Jian Li, and Mingda Qiao. Towards instance optimal bounds for best arm identi cation. *arXiv preprint arXiv:1608.06031*, 2016b.

Lijie Chen, Jian Li, and Mingda Qiao. Nearly Instance Optimal Sample Complexity Bounds for Top-k Arm Selection. *ArXiv e-prints*, February 2017.

Shouyuan Chen, Tian Lin, Irwin King, Michael R Lyu, and Wei Chen. Combinatorial pure exploration of multi-armed bandits. In *NIPS*, pages 379{387, 2014.

Herman Cherno . Sequential design of experiments. *The Annals of Mathematical Statistics*, 30(3):755{770, 1959.

VP Draglia, Alexander G Tartakovsky, and Venugopal V Veeravalli. Multihypothesis sequential probability ratio tests. i. asymptotic optimality. *IEEE Transactions on Information Theory*, 45(7):2448{2461, 1999.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Pac bounds for multi-armed bandit and markov decision processes. In *COLT*, pages 255{270. Springer, 2002.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *JMLR*, 7:1079{1105, 2006.

RH Farrell. Asymptotic behavior of expected sample size in certain one sided tests. *The Annals of Mathematical Statistics*, pages 36{72, 1964.

Victor Gabillon, Mohammad Ghavamzadeh, Alessandro Lazaric, and Sebastien Bubeck. Multi-bandit best arm identi cation. In *NIPS*, pages 2222{2230, 2011.

Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. Best arm identi cation: A uni ed approach to xed budget and xed con dence. In *NIPS*, pages 3212{3220, 2012.

Victor Gabillon, Alessandro Lazaric, Mohammad Ghavamzadeh, Ronald Ortner, and Peter Bartlett. Improved learning complexity in combinatorial pure exploration bandits. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 1004{1012, 2016.

Aurelien Garivier and Emilie Kaufmann. Optimal best arm identi cation with xed con - dence. In *29th Annual Conference on Learning Theory*, pages 998{1027, 2016.

Bhaskar Kumarr Ghosh. *Sequential tests of statistical hypotheses*. Addison-Wesley, 1970.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sebastien Bubeck. lil'ucb: An optimal exploration algorithm for multi-armed bandits. *COLT*, 2014.

Shivaram Kalyanakrishnan and Peter Stone. E cient selection of multiple bandit arms: Theory and practice. In *ICML*, pages 511{518, 2010.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. Pac subset selection in stochastic multi-armed bandits. In *ICML*, pages 655{662, 2012.

Zohar Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *ICML*, pages 1238{1246, 2013.

Zohar S Karnin. Veri cation based solution for structured mab problems. In *Advances in Neural Information Processing Systems*, pages 145{153, 2016.

Emilie Kaufmann and Shivaram Kalyanakrishnan. Information complexity in bandit subset selection. In *COLT*, pages 228{251, 2013.

Emilie Kaufmann, Olivier Cappe, and Aurelien Garivier. On the complexity of best arm identi cation in multi-armed bandit models. *arXiv preprint arXiv:1407.4443*, 2014.

Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, pages 1302{1338, 2000.

Andrea Locatelli, Maurilio Gutzeit, and Alexandra Carpentier. An optimal algorithm for the thresholding bandit problem. pages 258{265, 2016.

Shie Mannor and John N Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *JMLR*, 5:623{648, 2004.

Remi Munos et al. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends® in Machine Learning*, 7(1):1{129, 2014.

Mohammad Naghshvar, Tara Javidi, et al. Active sequential hypothesis testing. *The Annals of Statistics*, 41(6):2703{2738, 2013.

Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of computer and System Sciences*, 49(2):149{167, 1994.

Christos H Papadimitriou and Mihalis Yannakakis. On the approximability of trade-o s and optimal access of web sources. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 86{92. IEEE, 2000.

Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2002.

Abraham Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117{186, 1945.

Yuan Zhou, Xi Chen, and Jian Li. Optimal pac multiple arm identi cation with applications to crowdsourcing. In *ICML*, pages 217{225, 2014.

Organization of the Appendix

In Appendix B, we present the missing proofs in Section 5. In Appendices C and D, we prove our negative results on the sample complexity of Best-Set and General-Samp (Theorems 1.9 and 1.14).

Appendix A. Missing Proof in Section 4

In this section, we prove the \parallel simulation" lemma (Lemma 4.8) in Section 4, which we restate below for convenience.

Lemma 4.8 (restated) *If there is a (δ_0, δ, A, B) algorithm for a sampling problem for $\delta_0 = 0.01$ and any $\delta < 0.01$, there is also a δ-correct algorithm for any $\delta < 0.01$ that takes $O(A \ln \delta^{-1} + B)$ samples in expectation.*

Proof [Proof of Lemma 4.8] For each integer $k \geq 0$, let $\mathbb{A}_k$ be a $(\delta_0, \delta/2^{k+1}, A, B)$ algorithm for the problem. We construct an algorithm $\mathbb{A}$, which simulates the sequence $\{\mathbb{A}_k\}_{k\geq 0}$ of algorithms in parallel.

We number the time slots with positive integers $1, 2, \ldots$ At time slot t, for each integer $k \geq 0$ such that 2^k divides t, $\mathbb{A}$ either starts or resumes the execution of algorithm $\mathbb{A}_k$, until $\mathbb{A}_k$ requests a sample or terminates. In the former case, $\mathbb{A}$ draws a sample from the arm that $\mathbb{A}_k$ speci es and feeds it to $\mathbb{A}_k$. After that, the execution of $\mathbb{A}_k$ is suspended. As soon as some algorithm $\mathbb{A}_k$ terminates without an error (i.e., it indeed returns an answer), $\mathbb{A}$ outputs the answer that $\mathbb{A}_k$ returns.

To analyze this construction, we let $\mathcal{E}_{0,k}$ and $\mathcal{E}_{1,k}$ denote the events $\mathcal{E}_0$ and $\mathcal{E}_1$ in De nition 4.7 for algorithm $\mathbb{A}_k$. By de nition,

$$\Pr[\mathcal{E}_{0,k}] \geq 1 - \delta_0 - \delta/2^{k+1} \geq 0.98$$

and

$$\Pr[\mathcal{E}_{1,k}] \geq 1 - \delta/2^{k+1}.$$

We rst note that since $\mathbb{A}$ never returns an incorrect answer conditioning on $\bigcap_{k=0}^{\infty} \mathcal{E}_{1,k}$, by a union bound, the probability that $\mathbb{A}$ outputs an incorrect answer is upper bounded by

$$\sum_{k=0}^{\infty} \Pr\left[\overline{\mathcal{E}_{1,k}}\right] \leq \sum_{k=0}^{\infty} \delta/2^{k+1} = \delta,$$

and thus $\mathbb{A}$ is δ-correct. [12]

Then we analyze the sample complexity of $\mathbb{A}$. Let random variable T be the smallest index such that $\mathcal{E}_{0,T}$ happens. Since the execution of the algorithm sequence $\{\mathbb{A}_k\}$ is independent, we have

$$\Pr[T = k] \leq \prod_{j=0}^{k-1} (1 - \Pr[\mathcal{E}_{0,k}]) \leq 0.02^k.$$

Conditioning on $T = k$, $\mathbb{A}_k$ takes at most $\alpha(A \ln(2^{k+1}/\delta) + B)$ samples before it terminates for some universal constant α. Since $\mathbb{A}_k$ takes a sample every 2^k time slots, algorithm $\mathbb{A}$

12. We may easily verify that $\mathbb{A}$ terminates almost surely conditioning on $\bigcap_{k=0}^{\infty} \mathcal{E}_{1,k}$.

terminates within $\alpha 2^k(A\ln(2^{k+1}/\delta) + B)$ time steps. Then the total number of samples taken by $\mathbb{A}$ is at most

$$\sum_{j=0}^{\infty} \frac{\alpha 2^k(A\ln(2^{k+1}/\delta) + B)}{2^j} \leq \alpha 2^{k+1}(A\ln(2^{k+1}/\delta) + B).$$

Therefore, the expected number of samples taken by $\mathbb{A}$ is upper bounded by

$$\begin{aligned}
&\sum_{k=0}^{\infty} \Pr[T = k] \cdot \alpha 2^{k+1}(A\ln(2^{k+1}/\delta) + B) \\
&\leq \alpha \sum_{k=0}^{\infty} 0.02^k \cdot 2^{k+1}(A\ln 2^{k+1} + A\ln\delta^{-1} + B) \\
&\leq \alpha(A\ln\delta^{-1} + B)\sum_{k=0}^{\infty} 0.02^k \cdot 2^{k+1} + \alpha A \sum_{k=0}^{\infty} 0.02^k \cdot 2^{k+1}\ln 2^{k+1} \\
&= O\left(A\ln\delta^{-1} + B\right).
\end{aligned}$$

■

Appendix B. Missing Proofs in Section 5

In this section, we present the missing proofs of Lemmas 5.1 and 5.2 in Section 5.

B.1. Correctness

We restate Lemma 5.1 in the following.

Lemma 5.1 (restated) *For any $\delta \in (0, 0.01)$ and* Best-Set *instance $\mathcal{C}$, EfficientGapElim$(\mathcal{C}, \delta)$ returns the correct answer with probability $1 - \delta_0 - \delta$, and returns an incorrect answer w.p. at most δ.*

Proof Define $\{\mathcal{F}_r\}$ and $\{\widetilde{\mathcal{F}}_r\}$ as

$$\mathcal{F}_{r+1} := \{A \in \mathcal{F} : \hat{\mu}^{(r)}(A) \geq \theta_r\}$$

and

$$\widetilde{\mathcal{F}}_{r+1} := \{A \in \mathcal{F} : \hat{\mu}^{(r)}(A) \geq \theta_r - \epsilon_r/\lambda\}.$$

Intuitively, $\mathcal{F}_r$ is analogous to the set $\mathcal{F}_r$ used in NaiveGapElim, which represents the collection of remaining sets at the beginning of round r, while $\widetilde{\mathcal{F}}_{r+1}$ is a relaxed version of $\mathcal{F}_{r+1}$.

Then we note that at each round r, when SimultEst is called with parameters $\mu = \hat{\mu}^{(k-1)}$ and $\theta^{\text{high}} = \theta_{k-1} - \epsilon_{k-1}/\lambda$ for $k \in [r]$, the set $\{A' \in \mathcal{F} : \mu(A') \geq \theta^{\text{high}}\}$ involved in the mathematical program is exactly $\widetilde{\mathcal{F}}_k$. Similarly, when Verify is called at the last round with parameters $\theta_k^{\text{high}} = \theta_k - \epsilon_k/\lambda$, the set $\{A' \in \mathcal{F} : \hat{\mu}^{(k-1)}(A') \geq \theta_{k-1}^{\text{high}}\}$ is also identical to $\widetilde{\mathcal{F}}_k$.

Good events. As in the analysis of the NaiveGapElim algorithm, two good events $\mathcal{E}_0^{\text{good}}$ and $\mathcal{E}^{\text{good}}$ play important roles. Let $\mathcal{E}_{0,r}^{\text{good}}$ denote the event that either the algorithm terminates before or at round r, or it holds that

$$\left|(\hat{\mu}^{(r)}(A) - \hat{\mu}^{(r)}(B)) - (\mu(A) - \mu(B))\right| < \epsilon_k/\lambda$$

for all $k \in [r]$ and $A, B \in \widetilde{\mathcal{F}}_k$. Note that this definition is stronger than the one in the analysis of NaiveGapElim, where only the accuracy of the gaps between set pairs in $\widetilde{\mathcal{F}}_r$ is required. $\mathcal{E}_0^{\text{good}}$ is defined as the intersection of all $\mathcal{E}_{0,r}^{\text{good}}$'s. Moreover, we define $\mathcal{E}^{\text{good}}$ as the event that at line 7, it holds that

$$\left|(\hat{\mu}(\hat{O}) - \hat{\mu}(A)) - (\mu(\hat{O}) - \mu(A))\right| < \epsilon_k/\lambda$$

for all k and $A \in \widetilde{\mathcal{F}}_k$.

The following lemma, similar to Lemma 4.3, bounds the probability of the good events.

Lemma B.1 $\Pr\left[\mathcal{E}_0^{\text{good}}\right] \geq 1 - \delta_0$ *and* $\Pr\left[\mathcal{E}^{\text{good}}\right] \geq 1 - \delta$.

Proof Recall that $m^{(r)}$ is the sum of

$$\text{SimultEst}(\hat{\mu}^{(k-1)}, \theta_{k-1} - \epsilon_{k-1}/\lambda, \theta_{k-1} - 2\epsilon_{k-1}/\lambda, \epsilon_k/\lambda, \delta_r)$$

over all $k \in [r]$. This guarantees that $m^{(r)}$ is a valid solution to all programs. Specifically, for each $k \in [r]$ and $A, B \in \widetilde{\mathcal{F}}_k$, it holds that

$$\sum_{i \in A \triangle B} 1/m_i^{(r)} \leq \frac{(\epsilon_k/\lambda)^2}{2\ln(2/\delta_r)}.$$

By Lemma 2.1, it holds that

$$\Pr\left[\left|(\hat{\mu}^{(r)}(A) - \hat{\mu}^{(r)}(B)) - (\mu(A) - \mu(B))\right| < \epsilon_k/\lambda\right] \geq 1 - \delta_r.$$

A union bound over all possible choices of k, A, B yields that

$$\Pr\left[\mathcal{E}_{0,r}^{\text{good}}\right] \geq 1 - r|\mathcal{F}|^2\delta_r \geq 1 - \frac{\delta_0}{10r^2}.$$

It follows from another union bound over all r that $\Pr\left[\mathcal{E}_0^{\text{good}}\right] \geq 1 - \delta_0$, and a similar argument proves that $\Pr\left[\mathcal{E}^{\text{good}}\right] \geq 1 - \delta$. ■

Implications. We prove the analogues of Lemmas 4.4 and 4.5 for EfficientGapElim.

Lemma B.2 *Conditioning on* $\mathcal{E}_0^{\text{good}}$, $O \in \mathcal{F}_r \subseteq \widetilde{\mathcal{F}}_r$ *for all* r.

Proof [Proof of Lemma B.2] Suppose for a contradiction that $O \in \mathcal{F}_r \setminus \mathcal{F}_{r+1}$ for some r. Recall that OPT guarantees $\mathsf{opt}_r = \hat{\mu}^{(r)}(A)$ for some $A \in \mathcal{F}$ such that $\hat{\mu}^{(r-1)}(A) \geq \theta_{r-1} - \epsilon_{r-1}/\lambda$, i.e., $A \in \widetilde{\mathcal{F}}_r$. Since $A \in \widetilde{\mathcal{F}}_r$ and $O \in \mathcal{F}_r \subseteq \widetilde{\mathcal{F}}_r$, it holds conditioning on $\mathcal{E}_0^{\mathsf{good}}$ that

$$\hat{\mu}^{(r)}(O) - \hat{\mu}^{(r)}(A) > \mu(O) - \mu(A) - \epsilon_r/\lambda \geq -\epsilon_r/\lambda.$$

It follows that

$$\hat{\mu}^{(r)}(O) > \hat{\mu}^{(r)}(A) - \epsilon_r/\lambda = \mathsf{opt}_r - \epsilon_r/\lambda > \theta_r,$$

which leads to a contradiction to $O \notin \mathcal{F}_{r+1}$. ■

The following lemma is analogous to Lemma 4.5. In addition, we also characterize the relation between $G_{\geq r-1}$ and $\{A \in \mathcal{F} : \hat{\mu}^{(r)}(A) \geq \theta_r - 2\epsilon_r/\lambda\}$, which serves as a further relaxation of $\widetilde{\mathcal{F}}_r$.

Lemma B.3 *Conditioning on $\mathcal{E}_0^{\mathsf{good}}$, it holds that $G_{\geq r} \supseteq \widetilde{\mathcal{F}}_{r+1} \supseteq \mathcal{F}_{r+1} \supseteq G_{\geq r+1}$ and $G_{\geq r-1} \supseteq \{A \in \mathcal{F} : \hat{\mu}^{(r)}(A) \geq \theta_r - 2\epsilon_r/\lambda\}$.*

Proof [Proof of Lemma B.3] We prove by induction on r. The base case that $r = 0$ holds due to the observation that $\mathcal{F}_1 = \widetilde{\mathcal{F}}_1 = G_{\geq 0} = \mathcal{F}$. Now we prove the lemma for $r \geq 1$.

Part I. $A \notin G_{\geq r}$ implies $A \notin \widetilde{\mathcal{F}}_{r+1}$. Suppose that $A \in G_k$ for some $k \leq r-1$. Then by the inductive hypothesis, $A \in G_{\geq k} \subseteq \widetilde{\mathcal{F}}_k$. Also by Lemma B.2, $O \in \mathcal{F}_k \subseteq \widetilde{\mathcal{F}}_k$. Therefore, conditioning on event $\mathcal{E}_0^{\mathsf{good}}$, it holds that

$$\hat{\mu}^{(r)}(O) - \hat{\mu}^{(r)}(A) > \mu(O) - \mu(A) - \epsilon_k/\lambda > \epsilon_{k+1} - \epsilon_k/\lambda = (1/2 - 1/\lambda)\epsilon_k.$$

Recall that OPT guarantees that

$$\mathsf{opt}_r - \hat{\mu}^{(r)}(O) \geq \max_{B \in \mathcal{F}_r} \hat{\mu}^{(r)}(B) - \epsilon_r/\lambda - \hat{\mu}^{(r)}(O) \geq -\epsilon_r/\lambda.$$

The second step holds since, by Lemma B.2, $O \in \mathcal{F}_r$. Note that as $k \leq r-1$, $\epsilon_k \geq 2\epsilon_r$, and then the two inequalities above imply that

$$\mathsf{opt}_r - \hat{\mu}^{(r)}(A) > (1/2 - 1/\lambda)\epsilon_k - \epsilon_r/\lambda \geq (1 - 3/\lambda)\epsilon_r.$$

It follows that

$$\hat{\mu}^{(r)}(A) < \mathsf{opt}_r - (1 - 3/\lambda)\epsilon_r \leq \mathsf{opt}_r - (1/2 + 3/\lambda)\epsilon_r = \theta_r - \epsilon_r/\lambda,$$

and thus $A \notin \widetilde{\mathcal{F}}_{r+1}$. Here the second holds due to $\lambda = 20$.

Part II. $A \in G_{\geq r+1}$ implies $A \in \mathcal{F}_{r+1}$. For fixed $A \in G_{\geq r+1}$, the inductive hypothesis implies that $A \in G_{\geq r} \subseteq \widetilde{\mathcal{F}}_r$. Also by Lemma B.2, $O \in \widetilde{\mathcal{F}}_r$. Thus conditioning on event $\mathcal{E}_0^{\mathsf{good}}$,

$$\hat{\mu}^{(r)}(O) - \hat{\mu}^{(r)}(A) < \mu(O) - \mu(A) + \epsilon_r/\lambda \leq \epsilon_{r+1} + \epsilon_r/\lambda = (1/2 + 1/\lambda)\epsilon_r.$$

Note that OPT guarantees that $\text{opt}_r = \mu^{(r)}(B)$ for some $B \in \widetilde{\mathcal{F}}_r$. Thus, conditioning on $\mathcal{E}_0^{\text{good}}$,

$$\text{opt}_r - \mu^{(r)}(O) = \mu^{(r)}(B) - \mu^{(r)}(O) < \mu(B) - \mu(O) + \epsilon_r/\lambda \le \epsilon_r/\lambda.$$

Adding the two inequalities above yields

$$\mu^{(r)}(A) > \text{opt}_r - (1/2 + 2/\lambda)\epsilon_r = \theta_r,$$

and therefore $A \in \mathcal{F}_{r+1}$.

Part III. $A \notin G_{\ge r-1}$ implies $\mu^{(r)}(A) < \theta_r - 2\epsilon_r/\lambda$. Suppose $A \in G_k$ for $k \le r-2$. By the inductive hypothesis, $A \in G_{\ge k} \subseteq \widetilde{\mathcal{F}}_k$. Since $O \in \widetilde{\mathcal{F}}_k$ by Lemma B.2,

$$\mu^{(r)}(O) - \mu^{(r)}(A) > \mu(O) - \mu(A) - \epsilon_k/\lambda > \epsilon_{k+1} - \epsilon_k/\lambda = (1/2 - 1/\lambda)\epsilon_k.$$

The specification of OPT guarantees that

$$\text{opt}_r - \mu^{(r)}(O) \ge \max_{B \in \mathcal{F}_r} \mu^{(r)}(B) - \epsilon_r/\lambda - \mu^{(r)}(O) \ge -\epsilon_r/\lambda.$$

Recall that $k \le r-2$ and $\lambda = 20$. Therefore,

$$\begin{aligned}\mu^{(r)}(A) &< \text{opt}_r - (1/2 - 1/\lambda)\epsilon_k + \epsilon_r/\lambda \\ &= \text{opt}_r - (2 - 4/\lambda)\epsilon_r + \epsilon_r/\lambda \\ &< \text{opt}_r - (1/2 + 2/\lambda)\epsilon_r - 2\epsilon_r/\lambda = \theta_r - 2\epsilon_r/\lambda.\end{aligned}$$

■

Correctness conditioning on $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$. We show that EfficientGapElim always returns the correct answer O conditioning on both $\mathcal{E}_0^{\text{good}}$ and $\mathcal{E}^{\text{good}}$. Let r^* be a sufficiently large integer such that $G_{\ge r^*} = \{O\}$. By Lemma B.3, it holds that $\widetilde{\mathcal{F}}_{r^*+1} = \mathcal{F}_{r^*+1} = \{O\}$ conditioning on $\mathcal{E}_0^{\text{good}}$. Thus, the condition at line 4 is eventually satisfied, either before or at round $r^* + 1$.

It suffices to show that the condition of the if-statement at line 8 is also met, and thus the algorithm would return the correct answer, instead of reporting an error. Fix $k \in [r-1]$ and $A \in \mathcal{F}$ with $\mu^{(k)}(A) < \theta_k$. Since $A \notin \mathcal{F}_{k+1}$, by Lemma B.3, $A \notin G_{\ge k+1}$, and therefore $A \in G_t$ for some $t \le k$. By Lemma B.3, $A \in \widetilde{\mathcal{F}}_t$. Thus, conditioning on $\mathcal{E}^{\text{good}}$,

$$\mu(O) - \mu(A) > \mu(O) - \mu(A) - \epsilon_t/\lambda > \epsilon_{t+1} - \epsilon_t/\lambda = (1/2 - 1/\lambda)\epsilon_t \ge 2\epsilon_k/\lambda.$$

It follows that $\mu(O) - \mu(A) \ge 2\epsilon_k/\lambda$ for all $A \in \mathcal{F}$ with $\mu^{(k)}(A) < \theta_k$. According to the specification of Check, $\text{Check}(O, \mu^{(k)}, \mu, \theta_k, \epsilon_k/\lambda)$ always returns true, and thus the algorithm returns the optimal set O.

Soundness conditioning on $\mathcal{E}^{\text{good}}$. Now we show that the algorithm never returns an incorrect answer (i.e., a sub-optimal set) conditioning on $\mathcal{E}^{\text{good}}$. Suppose that at some round r, $\widetilde{\mathcal{F}}_r = \{\hat{O}\}$ for $\hat{O} \in \mathcal{F} \setminus \{O\}$, and thus the condition at line 4 is met. It suffices to show that the algorithm reports an error, rather than incorrectly returning $\hat{O}$ as the answer.

Since the correct answer O is not in $\widetilde{\mathcal{F}}_r$, there exists an integer k such that $O \in \widetilde{\mathcal{F}}_k \setminus \widetilde{\mathcal{F}}_{k+1}$. Conditioning on $\mathcal{E}^{\mathsf{good}}$, it holds that

$$\hat{\mu}(\hat{O}) - \hat{\mu}(O) < \mu(\hat{O}) - \mu(O) + \epsilon_k/\lambda \le \epsilon_k/\lambda.$$

Therefore, we have $\hat{\mu}^{(k)}(O) < \theta_k - \epsilon_k/\lambda$ and $\hat{\mu}(\hat{O}) - \hat{\mu}(O) \le \epsilon_k/\lambda$. Thus $\mathsf{Check}(\hat{O}, \hat{\mu}^{(k)}, \hat{\mu}, \theta_k, \epsilon_k/\lambda)$ is guaranteed to return false, and the algorithm does not return the incorrect answer $\hat{O}$.

This finishes the proof of Lemma 5.1. ■

B.2. Sample Complexity

We restate Lemma 5.2 for convenience.

Lemma 5.2 (restated) *For any $\delta \in (0, 0.01)$ and* Best-Set *instance $\mathcal{C}$, EfficientGapElim$(\mathcal{C}, \delta)$ takes*

$$O\left(\mathit{Low}(\mathcal{C}) \ln \delta^{-1} + \mathit{Low}(\mathcal{C}) \ln^2 \quad ^{-1} \left(\ln \ln \quad ^{-1} + \ln |\mathcal{F}|\right)\right)$$

samples conditioning on event $\mathcal{E}_0^{\mathsf{good}} \cap \mathcal{E}^{\mathsf{good}}$.

Proof [Proof of Lemma 5.2] For a Best-Set instance $\mathcal{C} = (S, \mathcal{F})$, let τ^* be the optimal solution to the program in (1):

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in S} \tau_i \\ \text{subject to} \quad & \sum_{i \in O \triangle A} 1/\tau_i \le [\mu(O) - \mu(A)]^2, \ \forall A \in \mathcal{F} \\ & \tau_i > 0, \ \forall i \in S. \end{aligned}$$

Recall that $\mathsf{Low}(\mathcal{C}) = \sum_{i \in S} \tau_i^*$.

We start by upper bounding the number of samples taken according to $m^{(r)}$ in each round r. Specifically, we construct a small feasible solution to the mathematical program defined in

$$\mathsf{SimultEst}(\hat{\mu}^{(k-1)}, \theta^{\mathsf{high}}, \theta^{\mathsf{low}}, \epsilon_k/\lambda, \delta_r),$$

where $\theta^{\mathsf{high}} = \theta_{k-1} - \epsilon_{k-1}/\lambda$ and $\theta^{\mathsf{low}} = \theta_{k-1} - 2\epsilon_{k-1}/\lambda$, thereby obtaining a bound on the optimal solution of the program.

Let $\alpha = 64\lambda^2 \ln(2/\delta_r)$ and $m_i = \alpha \tau_i^*$. Fix $A, B \in \{A' \in \mathcal{F} : \hat{\mu}^{(k-1)}(A') \ge \theta^{\mathsf{low}}\}$. By Lemma B.3, we have $A, B \in G_{\ge k-2}$, and thus both $\mu(O) - \mu(A)$ and $\mu(O) - \mu(B)$ are smaller than or equal to $\epsilon_{k-2} = 4\epsilon_k$. It follows that

$$\begin{aligned} \sum_{i \in A \triangle B} 1/m_i &\le \alpha^{-1} \left(\sum_{i \in O \triangle A} 1/\tau_i^* + \sum_{i \in O \triangle B} 1/\tau_i^* \right) \\ &\le \alpha^{-1} \left[[\mu(O) - \mu(A)]^2 + [\mu(O) - \mu(B)]^2 \right] \\ &\le 2\alpha^{-1} \cdot (4\epsilon_k)^2 = \frac{(\epsilon_k/\lambda)^2}{2 \ln(2/\delta_r)}. \end{aligned}$$

Here the second step holds since τ^* is a feasible solution to the program in (1). The last step applies $\alpha = 64\lambda^2 \ln(2/\delta_r)$. Therefore, this setting $\{m_i\}$ is a valid solution even for the *tightened* program de ned just above the description of SimultEst($\hat{\mu}^{(k-1)}, \theta^{\text{high}}, \theta^{\text{low}}, \epsilon_k/\lambda, \delta_r$) (Algorithm 4). Moreover, by our choice of $m_i = \alpha\tau_i^*$, the number of samples contributed by r and k is upper bounded by

$$\sum_{i\in S} m_i = \alpha \sum_{i\in S} \tau_i^* = O(\text{Low}(\mathcal{C}) \ln \delta_r^{-1}) = O\left(\text{Low}(\mathcal{C}) \left(\ln r + \ln |\mathcal{F}|\right)\right).$$

In sum, EfficientGapElim takes $O\left(\text{Low}(\mathcal{C}) \left(r \ln r + r \ln |\mathcal{F}|\right)\right)$ samples in round r.

This can now be used to bound the number of samples in all but the last round. Let $= \mu(O) - \max_{A\in\mathcal{F}\setminus\{O\}} \mu(A)$ and $r^* = \left\lfloor \log_2 \quad ^{-1} \right\rfloor + 1$. Observe that $G_{\geq r^*} = G_{\geq r^*+1} = \{O\}$. Thus by Lemma B.3, $\widetilde{\mathcal{F}}_{r^*+1} = \{O\}$ and the algorithm terminates before or at round $r^* + 1$. Summing over all r between 1 and r^* yields

$$\begin{aligned} O\left(\text{Low}(\mathcal{C}) \sum_{r=1}^{r^*} r \cdot (\ln r + \ln |\mathcal{F}|)\right) &= O\left(r^* \cdot \text{Low}(\mathcal{C}) \left(r^* \ln r^* + r^* \ln |\mathcal{F}|\right)\right) \\ &= O\left(\ln^2 \quad ^{-1} \cdot \text{Low}(\mathcal{C}) \left(\ln \ln \quad ^{-1} + \ln |\mathcal{F}|\right)\right). \end{aligned}$$

Then we bound the number of samples taken at the last round, denoted by round r. Let $\beta = 32\lambda^2 \ln(2r|\mathcal{F}|/\delta)$, and $m_i = \beta\tau_i^*$. We show that $\{m_i\}$ is a feasible solution to the program in

$$\text{Verify}(\{\hat{\mu}^{(k)}\}, \{\theta_k^{\text{high}}\}, \{\theta_k^{\text{low}}\}, \delta/(r|\mathcal{F}|)).$$

Here $\theta_k^{\text{high}} = \{\theta_k - \epsilon_k/\lambda\}$, and $\theta_k^{\text{low}} = \{\theta_k - 2\epsilon_k/\lambda\}$. Fix $k \in [r]$ and $A \in \{A' \in \mathcal{F} : \hat{\mu}^{(k-1)}(A) \geq \theta_{k-1}^{\text{low}}\}$. By Lemma B.3, we have $A \in G_{\geq k-2}$, which implies that $\mu(O) - \mu(A) \leq \epsilon_{k-2} = 4\epsilon_k$. Recall that by Lemma B.2, $\hat{O} = O$. Thus we have

$$\begin{aligned} \sum_{i\in\hat{O}\triangle A} 1/m_i &= \beta^{-1} \sum_{i\in O\triangle A} 1/\tau_i^* \\ &\leq \beta^{-1}[\mu(O) - \mu(A)]^2 \\ &\leq 32\beta^{-1}\epsilon_k^2 = \frac{(\epsilon_k/\lambda)^2}{2\ln(2r|\mathcal{F}|/\delta)}. \end{aligned}$$

Recall that $r \leq r^* + 1 = O(\ln \quad ^{-1})$. Therefore, the number of samples taken in the last round is upper bounded by

$$\begin{aligned} \sum_{i\in S} m_i = \beta \sum_{i\in S} \tau_i^* &= O\left(\text{Low}(\mathcal{C}) \left(\ln \delta^{-1} + \ln r + \ln |\mathcal{F}|\right)\right) \\ &= O\left(\text{Low}(\mathcal{C}) \left(\ln \delta^{-1} + \ln \ln \quad ^{-1} + \ln |\mathcal{F}|\right)\right). \end{aligned}$$

Therefore, conditioning on $\mathcal{E}_0^{\text{good}} \cap \mathcal{E}^{\text{good}}$, the number of samples taken by EfficientGapElim is

$$O\left(\text{Low}(\mathcal{C}) \ln \delta^{-1} + \text{Low}(\mathcal{C}) \ln^2 \quad ^{-1} \left(\ln \ln \quad ^{-1} + \ln |\mathcal{F}|\right)\right).$$

This completes the analysis of the sample complexity. ■

Appendix C. Worst-Case Lower Bound for Combinatorial Bandit

In this section we construct a family of Best-Set instance to show that

$$O(\mathsf{Low}(\mathcal{C}) \cdot (\ln|\mathcal{F}| + \ln\delta^{-1}))$$

samples are required for any δ-correct algorithm in the worst case. We need the following lemma for our theorem, which constructs a list of subsets resembling the Nisan-Wigderson design Nisan and Wigderson (1994).

Lemma C.1 *Given an integer n and there exists a list of $m = 2^{cn}$ subsets $S_1, S_2, \ldots, S_m$ of $[n]$ where c is a universal constant, such that $|S_i| = \ell =$ (n) for each S_i, and $|S_i \cap S_j| \le \ell/2$ for each $i \neq j$.*

Proof We prove the lemma via the probabilistic method. Let $\ell = n/10$, and $m = 2^{cn}$. We simply let $S_1, S_2, \ldots, S_m$ be a sequence of independent uniformly random subsets of $[n]$ with size ℓ. Clearly, we have

$$\Pr[|S_i \cap S_j| > \ell/2] \le 2^{-\ (n)}$$

for each $i \neq j$. Hence, we can set the constant c to be su ciently small so that

$$\Pr[\exists i \neq j, |S_i \cap S_j| > \ell/2] < m^2 \cdot 2^{-\ (n)} < 1,$$

which implies the existence of the desired list. ■

We now prove Theorem 1.9, which we restate here for convenience.

Theorem 1.9. (restated) *(i) For $\delta \in (0, 0.1)$, two positive integers n and $m \le 2^{cn}$ where c is a universal constant, and every δ-correct algorithm $\mathbb{A}$ for* Best-Set*, there exists an infinite sequence of n-arm instances $\mathcal{C}_1 = (S_1, \mathcal{F}_1), \mathcal{C}_2 = (S_2, \mathcal{F}_2), \ldots,$ such that $\mathbb{A}$ takes at least*

$$(\mathit{Low}(\mathcal{C}_k) \cdot (\ln|\mathcal{F}_k| + \ln\delta^{-1}))$$

samples in expectation on each $\mathcal{C}_k$, $|\mathcal{F}_k| = m$ for all k, and $\mathit{Low}(\mathcal{C}_k)$ goes to infinity.
(ii) Moreover, for each $\mathcal{C}_k$, there exists a δ-correct algorithm $\mathbb{A}_k$ for Best-Set *such that $\mathbb{A}_k$ takes*

$$O(\mathit{Low}(\mathcal{C}_k) \cdot \mathrm{poly}(\log n, \ln\delta^{-1}))$$

samples in expectation on it. (The constants in and O do not depend on n, m, δ and k.)

Our proof for the rst part is based on a simple but delicate reduction to the problem of distinguishing two instances with a much smaller con dence parameter $O(\delta/|\mathcal{F}|)$.

Proof [Proof of the rst part of Theorem 1.9] We x a real number $\in (0, 0.1)$, and let constant c and $\ell =$ (n) be as in Lemma C.1. For each subset $A \subseteq [n]$, we de ne $\mathcal{C}_A$ to be the n-arm instance whose i-th arm has mean when $i \in A$ and mean 0 otherwise. Let $S_1, S_2, \ldots, S_{2^{cn}}$ be a list whose existence is guaranteed by Lemma C.1, and set $\mathcal{F} = \{S_1, S_2, \ldots, S_m\}$.

Let $\mathbb{A}$ be a δ-correct algorithm for Best-Set. For a subset $A \in \mathcal{F}$, let $\mathcal{E}_A$ be the event that $\mathbb{A}$ outputs $\mathcal{C}_A$. Fixing a subset $A \in \mathcal{F}$, the de nition implies that

$$\sum_{B \in \mathcal{F}, B \neq A} \Pr_{\mathbb{A}, \mathcal{C}_A}[\mathcal{E}_B] \le \delta.$$

By a simple averaging argument, there exists another subset $B \in \mathcal{F}$ such that $\Pr_{\mathbb{A},\mathcal{C}_A}[\mathcal{E}_B] \leq 2\delta/|\mathcal{F}|$. Now, from the fact that $\mathbb{A}$ is δ-correct, we have $\Pr_{\mathbb{A},\mathcal{C}_B}[\mathcal{E}_B] \geq 0.9$. Combining the above two facts with Lemma 2.3, we can see that $\mathbb{A}$ must spend at least

$$d\left(\Pr_{\mathbb{A},\mathcal{C}_B}[\mathcal{E}_B], \Pr_{\mathbb{A},\mathcal{C}_A}[\mathcal{E}_B]\right) \cdot \quad ^{-2} = \quad ((\log|\mathcal{F}| + \log\delta^{-1}) \cdot \quad ^{-2})$$

samples on $\mathcal{C}_B$ in expectation. On the other hand, one can easily verify that setting $\tau_i = (1/\ell \cdot \quad ^{-2})$ satis es the constraints in the lower bound program (1) in Section 3, and hence $\mathrm{Low}(\mathcal{C}_B) \leq \quad (n/\ell \cdot \quad ^{-2}) = \quad (\quad ^{-2})$.

Therefore, to prove the rst part of this theorem, we set to be $1/n, 1/2n, 1/3n, \ldots$ and set $\mathcal{C}_k$ to be corresponding $\mathcal{C}_B$ constructed from the above procedure. (The property that $\leq 1/n$ will be used in the proof for the second part.) ■

For the second part of Theorem 1.9, we rst design an algorithm for an interesting special case of General-Samp.

Theorem C.2 *For a positive integer n, a positive real number $r \leq 1$ and a vector $u \in \mathbb{R}^n$, we define*

$$\mathcal{O} = \{\{u\}, \{v \in \mathbb{R}^n : \|u - v\|_2 \geq r\}\}.$$

There is a δ-correct algorithm for General-Samp *which takes*

$$O\left(n \log n \cdot r^{-2} \cdot (\log n + \ln\delta^{-1}) \cdot \ln\delta^{-1}\right)$$

samples in expectation on the instance $\mathcal{I} = (S, \mathcal{O})$, where S is a sequence of arms with mean profile u.

Before proving Theorem C.2, we show it implies the moreover part of Theorem 1.9.

Proof [Proof of the moreover part of Theorem 1.9] Let $\mathcal{C}_k = (S_k, \mathcal{F}_k)$ be a constructed instance in the proof of the rst part of Theorem 1.9, and , B, $\mathcal{F}$, ℓ, m be the corresponding parameters during its construction. From our choice of , we have $\leq 1/n$. And in the whole proof we assume n is su ciently large for simplicity.

Let $\mathbb{A}_{\mathsf{ball}}$ be the algorithm guaranteed by Theorem C.2. Our algorithm works as follows:

- Given an instance $\mathcal{C} = (S, \mathcal{F})$, run an arbitrary δ-correct algorithm for Best-Set when $\mathcal{F} \neq \mathcal{F}_k$.
- Run $\mathbb{A}_{\mathsf{ball}}$ with $r = c_1 \cdot \quad \cdot \sqrt{n}$, mean pro le u set as the mean pro le of instance $\mathcal{C}_k$ and con dence level set as $\delta/2$, where c_1 is a constant to be speci ed later. (Note that $r \leq 1$ as $\leq 1/n$.)
  - { Recall that $\mathcal{O} = \{A_1, A_2\}$, where $A_1 = \{u\}$ and $A_2 = \{v \in \mathbb{R}^n : \|u - v\|_2 \geq r\}$.
  - { (Case I) If $\mathbb{A}_{\mathsf{ball}}$ returns A_1, outputs set B.
  - { (Case II) Otherwise, run an arbitrary $\delta/2$-correct algorithm for Best-Set, and outputs its output.

First, we condition on the event $\mathcal{E}_{\mathsf{good}}$ that both $\mathbb{A}_{\mathsf{ball}}$ and the simulated algorithm in Case II operate correctly, which happens with probability at least $1-\delta$. Then we prove its correctness. Since we condition on $\mathcal{E}_{\mathsf{good}}$, whenever it enters Case II, it must output the correct answer. So it would only make mistakes in Case I.

Now we suppose that the algorithm enters Case I. Let u be the mean pro le of the instance $\mathcal{C}_k$, and v be the mean pro le of the given instance $\mathcal{C}$. Conditioning on $\mathcal{E}_{\mathsf{good}}$, from Theorem C.2, we must have

$$\|u-v\|_2 < c_1 \cdot \quad \cdot \sqrt{n}, \tag{10}$$

since otherwise $v \in A_2$ and $\mathbb{A}_{\mathsf{ball}}$ would output A_2 instead. We are going to show in this case, the correct answer must be B. That is tantamount to prove that for $A \in \mathcal{F}$ with $A \neq B$, we have

$$\sum_{i\in B} v_i - \sum_{i\in A} v_i = \sum_{i\in B\setminus A} v_i - \sum_{i\in A\setminus B} v_i > 0.$$

Note that from the construction of $\mathcal{C}_k$, we have $u_i =$ when $i \in B$, and $u_i = 0$ otherwise. Therefore,

$$\sum_{i\in B} u_i - \sum_{i\in A} u_i = \sum_{i\in B\setminus A} u_i - \sum_{i\in A\setminus B} u_i \geq (|B| - |A\cap B|) \cdot \quad = \quad \cdot \ell/2,$$

and

$$\left|\left(\sum_{i\in B} u_i - \sum_{i\in A} u_i\right) - \left(\sum_{i\in B} v_i - \sum_{i\in A} v_i\right)\right| = \left|\sum_{i\in B\setminus A}(u_i - v_i) - \sum_{i\in A\setminus B}(u_i - v_i)\right|$$
$$\leq \|u-v\|_1 \leq \|u-v\|_2 \cdot \sqrt{n} < c_1 \cdot \quad \cdot n.$$

Since $\ell =$ (n), we set c_1 to be a su ciently small constant so that $c_1 \cdot \quad \cdot n < \quad \cdot \ell/2$. This implies that

$$\sum_{i\in B\setminus A} v_i - \sum_{i\in A\setminus B} v_i > 0$$

and hence

$$\sum_{i\in B} v_i > \sum_{i\in A} v_i.$$

Therefore, B strictly dominates any other set $A \in \mathcal{F}$ in the given instance $\mathcal{C}$, which means it is the correct answer.This concludes the proof for its correctness.

For the sample complexity on the instance $\mathcal{C}_k$, note that conditioning on $\mathcal{E}_{\mathsf{good}}$, it must enter Case I, which means it takes

$$\begin{aligned} &O(n \cdot r^{-2} \cdot \mathsf{poly}(\log n, \ln \delta^{-1})) \\ &= O(\quad ^{-2} \cdot \mathsf{poly}(\log n, \ln \delta^{-1})) \\ &= O(\mathsf{Low}(\mathcal{C}_k) \cdot \mathsf{poly}(\log n, \ln \delta^{-1})) \end{aligned}$$

samples on that instance with probability $1-\delta$.

Strictly speaking, the above sample complexity does not hold in expectation, as the algorithm may takes an arbitrary number of samples if $\mathcal{E}_{\text{good}}$ does not happen. So we complete the final step by a simple application of the parallel simulation trick (see Lemma 4.8), to transform the above algorithm into an algorithm with an

$$O(\text{Low}(\mathcal{C}_k) \cdot \text{poly}(\log n, \ln \delta^{-1}))$$

expected sample complexity on $\mathcal{C}_k$. ■

Finally, we devote the rest of this section to prove Theorem C.2.

Proof [Proof of Theorem C.2] Without loss of generality, we can assume that u is the all zero vector $\mathbf{z}$. Let $A_1 = \{u\}$ and $A_2 = \{v \in \mathbb{R}^n : \|u - v\|_2 \geq r\}$, then $\mathcal{O} = \{A_1, A_2\}$. For simplicity We assume n is sufficiently large in the whole proof. Our algorithm works as follows.

- Given an instance $\mathcal{I}_0 = (S, \mathcal{O}_0)$, when $\mathcal{O}_0 \neq \mathcal{O}$, run another δ-correct algorithm for General-Samp instead (for example the algorithm in Section 6).
- For each integer k from 1 to $\lceil \log_2 n \rceil + 2$.
  - { Pick $N_k = c_1 \cdot n \log n \cdot 2^{-k} \cdot \ln \delta^{-1}$ arms at uniformly random, let the set of taken arms be S_k.
  - { For each arm $a \in S_k$, take $c_2 \cdot r^{-2} \cdot 2^k \cdot (\log n + \ln \delta^{-1})$ samples from it and let $\hat{\mu}_a^k$ be its empirical mean. If there is an arm $a \in S_k$ such that $|\hat{\mu}_a^k| > r \cdot 2^{-k/2-1}$, output A_2 and terminates the algorithm.
- If the algorithm does not halt in the above step, then output A_1.

To show the algorithm works, we start by setting c_2 to be a sufficiently large constant so that for each k, and each arm $a \in S_k$, we have

$$\Pr\left[|\hat{\mu}_a^k - \mu_a| \geq r \cdot 2^{-k/2-1}\right] < \delta/2 \cdot n^{-2}.$$

By a simple union bound over all k, with probability at least $1 - \delta/2$,

$$\left|\hat{\mu}_a^k - \mu_a\right| < r \cdot 2^{-k/2-1}$$

for all k and $a \in S_k$. We denote the above as event $\mathcal{E}_{\text{good}}$. A simple calculation shows that the above algorithm takes

$$O\left(n \log n \cdot r^{-2} \cdot (\log n + \ln \delta^{-1}) \cdot \ln \delta^{-1}\right)$$

samples.

Next we prove its correctness. Let the mean profile of the given instance $\mathcal{I}_0$ be x. We first show that when x equals $u = \mathbf{z}$ (i.e. $x \in A_1$), the algorithm outputs A_1 with probability at least $1 - \delta$. Conditioning on event $\mathcal{E}_{\text{good}}$, for each k and $a \in S_k$, we have $|\hat{\mu}_a^k| < r \cdot 2^{-k/2-1}$,

therefore the algorithm outputs the correct answer A_1. Since $\Pr[\mathcal{E}_{\text{good}}] \geq 1-\delta$, we finish the case when $x = z$.

For the second case when $x \in A_2$, we have $\|x\|_2 \geq r$, which means

$$\sum_{i=1}^{n} x_i^2/r^2 \geq 1.$$

Now, for each positive integer k, we define $X_k = \{i : x_i^2/r^2 \in (2^{-k}, 2^{-k+1}]\}$, and let

$$W(X_k) := \sum_{i \in X_k} x_i^2/r^2.$$

We can see

$$\sum_{k=\lceil \log_2 n \rceil + 3}^{\infty} W(X_k) \leq n \cdot 2^{-\lceil \log_2 n \rceil - 2} \leq 1/4.$$

and hence

$$\sum_{k=1}^{\lceil \log_2 n \rceil + 2} W(X_k) \geq 3/4.$$

Let $k^\star$ be the k with maximum $W(X_k)$, then we have

$$W(X_{k^\star}) \geq \frac{1}{2 \log_2 n}.$$

When $k = k^\star$ in the above algorithm, we have that $|X_k| \geq \frac{1}{2 \log_2 n} \cdot 2^{k-1}$. Therefore, we can set c_1 to be sufficiently large so that we have

$$\Pr[|S_k \cap X_k| > 0] > 1 - \delta/2.$$

Let the above be event $\mathcal{E}_{\text{non-empty}}$. We claim that conditioning on both $\mathcal{E}_{\text{good}}$ and $\mathcal{E}_{\text{non-empty}}$, our algorithm correctly outputs A_2. Let $a \in S_k \cap X_k$, we have that $|\mu_a| > r \cdot 2^{-k/2}$. Moreover, $|\hat{\mu}_a^k| > r \cdot 2^{-k/2-1}$ from the definition of $\mathcal{E}_{\text{good}}$. Hence our algorithm outputs A_2. Since $\Pr[\mathcal{E}_{\text{good}} \cap \mathcal{E}_{\text{non-empty}}] \geq 1-\delta$, this finishes the case when $x \in A_2$, and hence the proof. ■

Appendix D. Another Worst-Case Lower Bound for the General Case

Recall that Best-Set is clearly a special case of General-Samp, so the lower bound in Section C also applies to the latter problem. Here we present another lower bound for the General-Samp problem, which illustrates the \non-uniform" nature of the instance-wise lower bound $\text{Low}(\mathcal{I})$. In the following we will construct a family of instances which are similar to an OR function and prove an

$$O(\text{Low}(\mathcal{I}) \cdot (n + \ln \delta^{-1}))$$

worst-case lower bound for all δ-correct algorithm $\mathbb{A}$ for the general sampling problem.

Theorem 1.14. (restated) *For $\delta \in (0, 0.1)$, a positive integer n and every δ-correct algorithm $\mathbb{A}$ for the general sampling problem, there exists an infinite sequence of n-arm instances $\mathcal{I}_1 = (S_1, \mathcal{O}_1), \mathcal{I}_2 = (S_2, \mathcal{O}_2), \ldots$, such that $\mathbb{A}$ takes at least*

$$(\mathit{Low}(\mathcal{I}_k) \cdot (\ln \delta^{-1} + n))$$

samples in expectation on each $\mathcal{I}_k$, $|\mathcal{O}_k| = O(1)$ for all k, and $\mathit{Low}(\mathcal{I}_k)$ goes to infinity. Moreover, for each $\mathcal{I}_k$, there exists a δ-correct algorithm $\mathbb{A}_k$ for General-Samp *such that $\mathbb{A}_k$ takes*

$$O(\mathit{Low}(\mathcal{I}_k) \cdot \ln \delta^{-1})$$

samples in expectation on it. (The constants in and O does not depend on n, m, δ and k.)

Proof We x a real number $\in (0, 1]$. Let z be the all zero vector with length n, and e_i be the length-n vector whose i-th element is and all other elements are zero. Consider the following collection of answer sets $\mathcal{O} = \{A_1, A_2\}$, where $A_1 = \{e_1, e_2, \ldots, e_n\}$ and $A_2 = \{z\}$. That is, we must distinguish between the cases when all arms have mean zero, and when exactly one arm has mean . In the rest of the proof, we will always assume the collection of answers of the instances are $\mathcal{O}$. Therefore, to specify an instance, we only need to specify a mean pro le.

For each $i \in [n]$, let $\mathcal{I}^{(i)}$ be the instance with mean pro le e_i, and $\mathcal{I}^{(0)}$ be the instance with mean pro le z. Let $\mathbb{A}$ be a δ-correct algorithm for the general sampling problem. First, it is not hard to see that $\mathrm{Low}(\mathcal{I}^{(i)}) = \ ^{-2}$ for each $i \in [n]$. We are going to show that $\mathbb{A}$ must draw at least $(n \cdot \ ^{-2})$ samples in expectation on at least one $\mathcal{I}^{(i)}$, where $i \in [n]$. When n is a constant, the above holds trivially, so we assume from now on that $n \geq 100$.

Consider the following new algorithm $\mathbb{A}_{\mathsf{new}}$, which simply simulates $\mathbb{A}$ as long as it draws at most $c_1 \cdot n \ ^{-2}$ samples, where c_1 is a small constant to be speci ed later. $\mathbb{A}_{\mathsf{new}}$ outputs $\mathbb{A}$'s output if $\mathbb{A}$ halts before the speci ed amount of steps, and outputs $\perp$ otherwise. Now, consider running $\mathbb{A}_{\mathsf{new}}$ on instance $\mathcal{I}^{(0)}$. Let $p_\perp$ be the probability that $\mathbb{A}_{\mathsf{new}}$ outputs $\perp$, and τ_i be the number of samples taken on the i-th arm. We claim that $p_\perp > 0.5$.

Suppose for contradiction that $p_\perp \leq 0.5$. So with probability at least 0.5, the simulated version of $\mathbb{A}$ within $\mathbb{A}_{\mathsf{new}}$ outputs something before it halts. Let p_i be the probability that $\mathbb{A}_{\mathsf{new}}$ outputs i. Since $\sum_{i=1}^{n} p_i \leq 1$, there are at least $n/2$ values of i satisfying $p_i \leq 2/n$. Let i° be such an i with the minimum $\mathbb{E}_{\mathbb{A}_{\mathsf{new}}, \mathcal{I}^{(0)}}[\tau_i]$. We have

$$\mathbb{E}_{\mathbb{A}_{\mathsf{new}}, \mathcal{I}^{(0)}}[\tau_{i^\circ}] \leq 2 \cdot c_1 \ ^{-2}$$

since

$$\sum_{i=1}^{n} \mathbb{E}_{\mathbb{A}_{\mathsf{new}}, \mathcal{I}^{(0)}}[\tau_i] \leq c_1 \cdot n \ ^{-2}.$$

Let $\mathcal{E}_{\mathsf{err}}$ be the event that $\mathbb{A}_{\mathsf{new}}$ outputs something di erent from $\perp$ and i°. Observe that $\Pr_{\mathbb{A}_{\mathsf{new}}, \mathcal{I}^{(0)}}[\mathcal{E}_{\mathsf{err}}] \geq 0.5 - 2/n \geq 0.48$.

Now we run $\mathbb{A}_{\mathsf{new}}$ on $\mathcal{I}^{(i^\circ)}$. Note that $\mathcal{I}^{(i^\circ)}$ and $\mathcal{I}^{(0)}$ di er only on arm i°, so by Lemma 2.3, we have

$$d\left(\Pr_{\mathbb{A}_{\mathsf{new}}, \mathcal{I}^{(0)}}[\mathcal{E}_{\mathsf{err}}], \Pr_{\mathbb{A}_{\mathsf{new}}, \mathcal{I}^{(i^\circ)}}[\mathcal{E}_{\mathsf{err}}]\right) \leq \mathbb{E}_{\mathbb{A}_{\mathsf{new}}, \mathcal{I}^{(0)}}[\tau_{i^\circ}] \ ^{2} = 2c_1.$$

For a su ciently small c_1, we can see the above leads to $\Pr_{\mathbb{A}_{\mathsf{new}},\mathcal{I}^{(i^\circ)}}[\mathcal{E}_{\mathsf{err}}] > 0.2$, which implies that running the original algorithm $\mathbb{A}$ yields an incorrect answer with probability at least 0.2 on instance $\mathcal{I}^{(i^\circ)}$, contradiction to the fact that $\mathbb{A}$ is δ-correct.

Therefore, we conclude that $p_\perp \geq 0.5$, which means the simulated $\mathbb{A}$ runs for a full $c_1 \cdot n \quad ^{-2}$ period with probability at least 0.5. Let $\mathcal{E}_\perp$ be the event that $\mathbb{A}_{\mathsf{new}}$ outputs $\perp$, and $i^\star$ be the i with minimum $\mathbb{E}_{\mathbb{A}_{\mathsf{new}},\mathcal{I}^{(0)}}[\tau_i]$. Clearly, we have $\mathbb{E}_{\mathbb{A}_{\mathsf{new}},\mathcal{I}^{(0)}}[\tau_i] \leq c_1 \cdot \quad ^{-2}$. Again as above, we run $\mathbb{A}_{\mathsf{new}}$ on instance $\mathcal{I}^{(i^\star)}$ and use Lemma 2.3 to get

$$d\left(\Pr_{\mathbb{A}_{\mathsf{new}},\mathcal{I}^{(0)}}[\mathcal{E}_\perp], \Pr_{\mathbb{A}_{\mathsf{new}},\mathcal{I}^{(i^\star)}}[\mathcal{E}_\perp]\right) \leq \mathbb{E}_{\mathbb{A}_{\mathsf{new}},\mathcal{I}^{(0)}}[\tau_{i^\star}] \quad ^2 = c_1.$$

To prove the rst part of our theorem, the above discussion gives us the (Low($\mathcal{I}_k$) $\cdot$ n) term. The (Low($\mathcal{I}_k$) $\cdot \ln \delta^{-1}$) part follows from Theorem 3.1, and we can set = $1/1, 1/2, \ldots, 1/k$ and let $\mathcal{I}_k$ be the corresponding $\mathcal{I}_{i^\star}$.

For the second part of the theorem, let $\mathcal{I}_k$ be a constructed instance, and , $i^\star$ be the parameters as in its construction process. Our algorithm $\mathbb{A}_k$ simply takes $O(\quad ^{-2} \ln \delta^{-1})$ samples from arm $i^\star$ so that

$$\Pr[|\mu_{i^\star} - \hat{\mu}_{i^\star}| < \quad /2] \geq 1 - \delta/2,$$

where $\hat{\mu}_{i^\star}$ is the empirical mean of arm $i^\star$.

If $\hat{\mu}_{i^\star} > \quad /2$ then it outputs A_1 and halts. Else, it runs another $\delta/2$-correct algorithm for General-Samp (for example, the algorithm in Section 6). Clearly, this is a δ-correct algorithm. And with probability at least $1 - \delta$, it takes $O(\quad ^{-2} \ln \delta^{-1})$ samples in total when running on instance $\mathcal{I}_k$. Finally we can turn the sample complexity into a bound in expectation via the parallel simulation trick (Lemma 4.8), which concludes the proof. ■