Proceedings of Machine Learning Research vol 65:1–37, 2017

Learning with Limited Rounds of Adaptivity: Coin Tossing, Multi-Armed Bandits, and Ranking from Pairwise Comparisons

Arpit Agarwal AARPIT@SEAS.UPENN.EDU
Shivani Agarwal ASHIVANI@SEAS.UPENN.EDU
Sepehr Assadi SASSADI@CIS.UPENN.EDU
Sanjeev Khanna SANJEEV@CIS.UPENN.EDU
Department of Computer and Information Science, University of Pennsylvania

Abstract

In many learning settings, active/adaptive querying is possible, but the number of rounds of adaptivity is limited. We study the relationship between query complexity and adaptivity in identifying the k most biased coins among a set of n coins with unknown biases. This problem is a common abstraction of many well-studied problems, including the problem of identifying the k best arms in a stochastic multi-armed bandit, and the problem of top-k ranking from pairwise comparisons.

An r-round adaptive algorithm for the k most biased coins problem specifies in each round the set of coin tosses to be performed based on the observed outcomes in earlier rounds, and outputs the set of k most biased coins at the end of r rounds. When $r = 1$, the algorithm is known as *non-adaptive*; when r is unbounded, the algorithm is known as *fully adaptive*. While the power of adaptivity in reducing query complexity is well known, full adaptivity requires repeated interaction with the coin tossing (feedback generation) mechanism, and is highly sequential, since the set of coins to be tossed in each round can only be determined after we have observed the outcomes of the coin tosses from the previous round. In contrast, algorithms with only few rounds of adaptivity require fewer rounds of interaction with the feedback generation mechanism, and offer the benefits of parallelism in algorithmic decision-making. Motivated by these considerations, we consider the question of how much adaptivity is needed to realize the optimal worst case query complexity for identifying the k most biased coins. Given any positive integer r, we derive essentially matching upper and lower bounds on the query complexity of r-round algorithms. We then show that $\Theta(\log^* n)$ rounds are both necessary and sufficient for achieving the optimal worst case query complexity for identifying the k most biased coins. In particular, our algorithm achieves the optimal query complexity in at most $\log^* n$ rounds, which implies that on any realistic input, 5 parallel rounds of exploration suffice to achieve the optimal worst-case sample complexity. The best known algorithm prior to our work required $\Theta(\log n)$ rounds to achieve the optimal worst case query complexity even for the special case of $k = 1$.

Keywords: Most biased coins, Best arms identification, Limited adaptivity, Multi-armed bandits, Ranking from pairwise comparisons, Top-k ranking, Active learning, Adaptivity

1. Introduction

In the classical *probably approximately correct* (PAC) model, the learner is a passive observer who is given a collection of randomly sampled observations from which to learn. In recent years, there has been growing interest in *active learning* models, where the learner can actively request labels or feedback at specific data points; the hope is that, by adaptively guiding the data collection process,

© 2017 A. Agarwal, S. Agarwal, S. Assadi & S. Khanna.

learning can be accomplished with fewer observations than in the passive case. Most learning algorithms operate in one of these settings: learning is either fully passive, or fully active.

In an increasing number of applications, while active querying is possible, the number of *rounds* of interaction with the feedback generation mechanism is limited. For example, in crowdsourcing, one can actively request feedback by sending queries to the crowd, but there is typically a waiting time before queries are answered; if the overall task is to be completed within a certain time frame, this effectively limits the number of rounds of interaction. Similarly, in marketing applications, one can actively request feedback by sending surveys to customers, but there is typically a waiting time before survey responses are received; again, if the marketing campaign is to be completed within a certain time frame, this effectively limits the number of rounds of interaction.

In this paper, we study active/adaptive learning with *limited rounds of adaptivity*, where the learner can actively request feedback at specific data points, but can do so in only a small number of rounds. Specifically, the learner is free to query any number of data points in each round; however, all data points to be queried in a given round must be submitted *simultaneously*, based only on feedback received in previous rounds. In this setting, we are interested not only in bounding the overall query complexity of the learner, but rather in understanding the tradeoff between the number of rounds and the overall query complexity: how many queries are needed given a fixed number of rounds, and conversely, given a target number of queries, how many rounds are necessary?

We study this question in the context of an abstract coin tossing problem, and discuss how the results give us novel insights into the round vs. query complexity tradeoff for two problems that have received increasing interest in the learning theory community in recent years: multi-armed bandits, and ranking from pairwise comparisons[1].

The abstract coin problem we study can be described as follows: say we are given n coins with unknown biases, each of which can be 'queried' by tossing the coin and observing the outcome of the toss. The goal is to find the k coins with highest biases. This problem is a special case of the problem of finding the k best arms in a stochastic multi-armed bandit (MAB), and has received considerable attention in recent years (Even-Dar et al., 2006; Kalyanakrishnan and Stone, 2010; Audibert and Bubeck, 2010; Kalyanakrishnan et al., 2012; Gabillon et al., 2012; Jamieson et al., 2013; Bubeck et al., 2013; Karnin et al., 2013; Chen and Li, 2015; Kaufmann et al., 2016; Jun et al., 2016; Chen et al., 2017). In particular, it is known that $O\big(\frac{n \log k}{\Delta_k^2}\big)$ coin tosses suffice to find the k most biased coins with arbitrarily high constant probability, where Δ_k is the gap between the k-th and $(k+1)$-th largest biases (Kalyanakrishnan and Stone, 2010; Even-Dar et al., 2006). It is also known that this bound is optimal in terms of the worst-case query complexity (Kalyanakrishnan et al., 2012; Mannor and Tsitsiklis, 2004). However, the previous best algorithms for this problem all required $\Omega(\log n)$ rounds of adaptivity to achieve the optimal worst-case query complexity. But are $\Omega(\log n)$ rounds necessary for achieving this optimal query complexity? (see Table 1; see also Section 2 for the exact definition of parameters involved).

We present an algorithm, AGRESSIVE-ELIMINATION, that significantly improves upon the round complexity of state-of-the-art algorithms, yet still achieves the optimal worst-case query complexity: given the gap parameter Δ_k, our algorithm returns the k most biased coins using $O\big(\frac{n \log k}{\Delta_k^2}\big)$ coin tosses with arbitrarily large constant probability in only $\log^*(n)$ rounds of adaptivity. The algorithm proceeds in rounds and in each round performs: *(i)* an "estimation" phase to approximate

1. In the MAB and ranking literature, the query complexity of an algorithm is often referred to as simply its *sample complexity*. In this paper we use the two terms interchangeably.

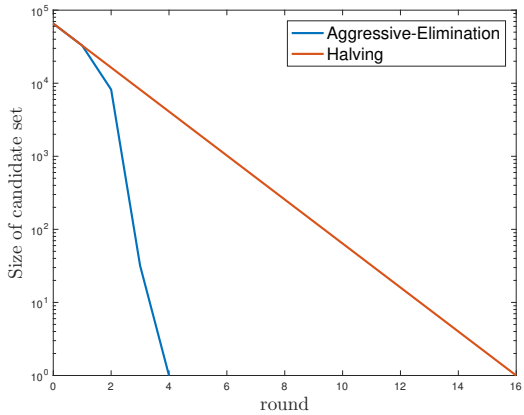

Figure 1: An example illustrating that our algorithm eliminates items more "aggresively" as compared to the HALVING algorithm of Kalyanakrishnan and Stone (2010); Even-Dar et al. (2006). Here, $n = 2^{16}$ and $k = 1$.

the bias of each coin, and *(ii)* an "elimination" phase to reduce the number of possible candidates and finds the top k most biased coins among the remaining candidates in the subsequent rounds. The elimination phase gets more "aggressive" over the rounds: in each round, the number of remaining coins reduces to an exponentially smaller fraction (across different rounds) of the current coins. This allows the algorithm to find the top k most biased coins in only $\log^* n$ rounds of adaptivity (as opposed to $\log n$ if the fraction was constant throughout). Figure 1 gives an example of the rate at which items are eliminated per round for AGRESSIVE-ELIMINATION algorithm, and the $\log n$-round HALVING algorithm (Kalyanakrishnan and Stone, 2010; Even-Dar et al., 2006). The main insight behind our algorithm is that by removing more and more coins in the elimination phase we can allocate more and more budget (i.e., samples for each remaining coin) to the estimation phase which in turn results in even more decrease in the number of candidate coins for the next round.

We further prove, perhaps surprisingly, that the $\log^*(n)$ bound achieved by our algorithm is essentially the "correct" number of rounds of adaptivity required for obtaining the optimal worst-case query complexity, even when k is only a constant. More formally, we prove that any algorithm that only uses $O\big(\frac{n}{\Delta_k^2}\big)$ coin tosses and recovers the set of top k most biased coins with some constant probability requires $\Omega(\log^*(n))$ rounds of adaptivity. Our lower bound proof is based on analyzing a family of "hard" instances for the problem which consists of k heavy coins and $n-k$ light coins. Using information- theoretic machinery, we show that any algorithm that uses small number of coin tosses in the first round can only "trap" the set of heavy coins in a large pool of candidates. We then inductively show that this forces the algorithm to still solve a "hard" problem on a large domain in the subsequent rounds which we show is not possible due the limited budget of the algorithm.

Finally, we address the question of round vs. query complexity tradeoff for this problem in a more fine-grained level: For any fixed number of rounds r, we present an algorithm for the above coin problem that uses $O\big(\frac{n}{\Delta_k^2}(\text{ilog}^{(r)}(n) + \log k)\big)$ coin tosses and prove a matching lower bound whenever k is a constant. Here, $\text{ilog}^{(r)}(\cdot)$ denotes the iterated logarithm of order r. Our results pro-

Table 1: Summary of some results for k best arms identification in stochastic multi-armed bandits.

	Algorithm	# Rounds of Adaptivity	Sample/Query Complexity
$k = 1$	Even-Dar et al. (2002)	$\Theta(\log(n))$	$O(\frac{n \log(1/\delta)}{\Delta_1^2})$
	Audibert and Bubeck (2010)	$\Theta(n)$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log^2(\frac{n}{\delta})\right)$
	Chen and Li (2015)	$\Omega(\log(n))$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\frac{\log(\min\{n, \Delta_i^{-1}\})}{\delta})\right)$
All $k \in [n]$	Kalyanakrishnan and Stone (2010)	$\Theta(\log(n))$	$O(\frac{n \log(k/\delta)}{\Delta_k^2})$
	Bubeck et al. (2013)	$\Theta(n)$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log^2(\frac{n}{\delta})\right)$
	This paper	$\log^*(n)$	$O(\frac{n \log(k/\delta)}{\Delta_k^2})$

Table 2: Summary of some results on top-k ranking from pairwise comparisons.

	Pairwise Comparison Model	# Rounds of Adaptivity	Sample/Query Complexity
Chen and Suh (2015)	Bradley-Terry-Luce	Non-adaptive	$O\left(\frac{n \log(n/\delta)}{(w_{[k]} - w_{[k+1]})^2}\right)$
Shah and Wainwright (2015)	General	Non-adaptive	$O(\frac{n \log(n/\delta)}{\Delta_k^2})$
Braverman et al. (2016)	Noisy Permutation	4	$O\left(\frac{n \log(n/\delta)}{(1-2p)^2}\right)$
Busa-Fekete et al. (2013), Heckel et al. (2016)	General	$\Omega\left(\Delta_k^{-2} \cdot \log(n)\right)$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\frac{n}{\delta \Delta_i})\right)$
This paper	General	$\log^*(n)$	$O(\frac{n \log(k/\delta)}{\Delta_k^2})$

vide a near-complete understanding of the power of each additional round of adaptivity in reducing the query complexity of the algorithms for this problem.

Our results for the above coin problem are also applicable to the problem of top-k ranking from pairwise comparisons, another problem that has received considerable interest in recent years (Feige et al., 1994; Busa-Fekete et al., 2013; Chen and Suh, 2015; Shah and Wainwright, 2015; Jang et al., 2016; Heckel et al., 2016; Davidson et al., 2014; Braverman et al., 2016). Most top-k ranking approaches we are aware of assume either a non-adaptive setting or a fully adaptive setting; the main exceptions to this are Feige et al. (1994); Davidson et al. (2014); Braverman et al. (2016), who consider the top-k ranking problem under limited rounds of adaptivity, but under the restricted *noisy permutation* model of pairwise comparisons (defined in Section 3). In our work, we make no assumptions on the underlying pairwise comparison model. Again, our results for the abstract coin problem above give us a novel algorithm for top-k ranking from pairwise comparisons that requires only $\log^*(n)$ rounds, with matching lower bounds; to our knowledge, this is the first study of this problem under general pairwise comparison models in the limited-adaptivity setting. See Table 2 for a summary (see also Section 3 for the exact definition of parameters involved).

Our work shows that for a well-studied class of learning problems, the power of fully adaptive exploration in minimizing worst-case query complexity is realizable by just a few rounds of adaptive exploration. In fact, for any realistic input size for the problems considered here, our work shows that at most 5 adaptive rounds are needed to realize optimal worst-case query complexity. We hope

that our techniques can be used for other classes of learning problems to gain an insight into how the query complexity changes as one interpolates between the fully passive and fully active settings.

1.1. Related Work

The general question of computation with limited rounds of adaptivity has been studied for certain problems such as sorting and selection in the theoretical computer science (TCS) literature under the term *parallel algorithms* (Valiant, 1975; Bollobás and Thomason, 1983; Ajtai et al., 1986; Pippenger, 1987; Alon and Azar, 1988; Cole, 1988; Bollobás and Brightwell, 1990; Feige et al., 1994; Davidson et al., 2014; Braverman et al., 2016). However, with the exception of Feige et al. (1994); Davidson et al. (2014); Braverman et al. (2016), these studies all operate in a deterministic setting, where any sample yields a deterministic outcome; this is unlike the setting we consider in our problems, where there is an underlying probabilistic model and queries yield noisy outcomes.

We note that the coin problem studied by Karp and Kleinberg (2007) is different from ours: there, given a ranked list of coins with unknown biases and a target bias $p \in (0, 1)$, the goal is to find the coins that have bias greater than p. In our case we do not know a ranking on the coins. Another line of work on biased coin identification is that of Chandrasekaran and Karp (2014); Malloy et al. (2012); Jamieson et al. (2016): there, given an infinite population of coins, each of which is of one of two types, 'heavy' or 'light', the goal is to identify a coin of the heavy type. In our case we have a finite population of coins, each of which can be of a different type. Moreover, all these previous papers work in the fully adaptive setting, while our focus is on the limited-adaptivity setting.

The problem of best arm identification in MABs has mostly been considered in a fully adaptive setting, where the learner can observe the outcome of any arm pull before selecting the next arm to be pulled (Even-Dar et al., 2006; Audibert and Bubeck, 2010; Kalyanakrishnan et al., 2012; Gabillon et al., 2012; Jamieson et al., 2013; Bubeck et al., 2013; Karnin et al., 2013; Hillel et al., 2013; Perchet et al., 2015; Chen and Li, 2015; Kaufmann et al., 2016; Jun et al., 2016; Chen et al., 2017). A recent work by Jun et al. (2016) is most closely related to our work. It considers algorithms that pull multiple arms in each round and there is a bound on the number of arms that the algorithm is allowed to pull in each round. However, the number of rounds required by their algorithm in the worst-case is $\Omega(\log(n))$ irrespective of the bound on the number of pulls in each round.

The problem of top-k ranking from (noisy) pairwise comparisons has mostly been considered in either the non-adaptive setting or the fully adaptive setting (Busa-Fekete et al., 2013; Chen and Suh, 2015; Shah and Wainwright, 2015; Jang et al., 2016; Heckel et al., 2016). Feige et al. (1994), and more recently Davidson et al. (2014); Braverman et al. (2016), considered a setting with limited rounds of adaptivity, but under a restricted pairwise comparison model that we refer to as the *noisy permutation* model (see Section 3 for details). In contrast, in this work, we make no assumptions on the underlying pairwise comparison model.

A more comprehensive summary of previous work appears in Tables 3–4 in Appendix D, and Table 5 in Appendix E (see also Appendix D.1).

1.2. Notation

For any integer $a \geq 1$, $[a] := \{1, \ldots, a\}$. For a (multi-)set of numbers $\{a_1, \ldots, a_n\}$, we define $a_{[i]}$ as the i-th largest value in this set (ties are broken arbitrarily). For any integer $r \geq 0$, $\text{ilog}^{(r)}(a)$ denotes the iterated logarithms of order r, i.e. $\text{ilog}^{(r)}(a) = \max\left\{\log\left(\text{ilog}^{(r-1)}(a)\right), 1\right\}$ and

$\text{ilog}^{(0)}(a) = a$. Matrices and vectors are denoted in boldface, e.g., $\boldsymbol{A}$ and $\boldsymbol{b}$, and random variables in serif, e.g., X.

For a random variable X, $\text{supp}(\mathsf{X})$ denotes the support of X and $\text{dist}(\mathsf{X})$ denotes its distribution. We use $\mathcal{U}$ to denote the uniform distribution. For any $p \in [0,1]$, $\mathcal{B}(p)$ denotes the Bernoulli distribution with mean p. We denote the Shannon entropy of a random variable A by $\mathbb{H}(\mathsf{A})$ and the mutual information of two random variables A and B by $\mathbb{I}(\mathsf{A}\,;\mathsf{B})$. Finally, for any two probability distribution μ, ν over the same support, $\|\mu - \nu\|_{tvd}$ denotes the total variation distance between μ and ν. A summary of useful information theory facts used in this paper is provided in Appendix A.

2. Finding the k Most Biased Coins / k Best Arms

Here, we present our main results on finding the k most biased coins using coin tosses with a limited number of rounds of adaptivity. We give an algorithm for this problem in Section 2.1, and a matching lower bound showing the algorithm achieves an optimal worst-case tradeoff between round and query complexity in Section 2.2. The coin problem is equivalent to the problem of the k best arms identification problem in MABs with Bernoulli reward distributions. Our results also extend to the more general case of MABs with sub-Guassian reward distributions; see Appendix D.

The specific problem we consider can be stated formally as follows: given n coins with unknown biases $p_1, \ldots, p_n$, and an integer $k \in [n]$, the goal is to identify (via tosses of the n coins) the set of k most biased coins. An important parameter in determining the query complexity of this problem is the *gap parameter* $\Delta_k := p_{[k]} - p_{[k+1]}$, i.e. the gap between the k-th and $(k+1)$-th highest biases (recall that $p_{[i]}$ denotes the bias of the i-th most biased coin). We also define $\Delta_i = \max\{\left|p_{[i]} - p_{[k+1]}\right|, \left|p_{[i]} - p_{[k]}\right|\}$. We will assume throughout that the set of k most biased coins is unique, i.e. that $\Delta_k > 0$; we will also assume our algorithm is given a lower bound Δ on the gap parameter ($\Delta_k \geq \Delta > 0$).[2]

We are interested here in algorithms that require limited *rounds* of adaptivity. In each round, an algorithm can decide to query various coins by tossing them (with no limit on the number of coins that can be tossed in a round or on the number of times any given coin can be tossed in a round); however, all tosses to be conducted in a given round must be chosen *simultaneously*, based only on the outcomes observed in previous rounds. We say an algorithm is an *r-round algorithm* if it uses at most r rounds of adaptivity; the total number of coin tosses it uses is termed its *query complexity*. For any $\delta \in [0,1)$, we say an algorithm is a *δ-error algorithm* for the above problem if it correctly returns the set of k most biased coins with probability at least $1 - \delta$.

2.1. A Limited-Adaptivity Algorithm for Finding the k Most Biased Coins

Our main algorithmic result is the following:

Theorem 1 *There exists an algorithm that given an integer $k \in [n]$, a set of n coins with gap parameter $\Delta_k \in (0,1)$, target number of rounds $r \geq 1$, and confidence parameter $\delta \in [0,1)$, finds the set of k most biased coins w.p. $\geq 1 - \delta$ using $O\left(\frac{n}{\Delta_k^2} \cdot \left(\text{ilog}^{(r)}(n) + \log\left(k/\delta\right)\right)\right)$ coin tosses and r rounds of adaptivity.*

2. We point out that the assumption that $\Delta_k > 0$ is *only* for simplicity of exposition; by picking Δ_k to be the gap between the bias of the k-th most biased coin and the next largest *distinct* bias value, our algorithm works as it is. The assumption about knowledge of Δ is also common in the MAB and ranking literature; see, e.g., (Even-Dar et al., 2006; Kalyanakrishnan and Stone, 2010; Chen and Suh, 2015; Shah and Wainwright, 2015).

We also point out that by setting $r = \log^*(n)$ in Theorem 1, we can achieve the *optimal worst-case query complexity* (Kalyanakrishnan et al., 2012; Mannor and Tsitsiklis, 2004) in a significantly smaller number of rounds of adaptivity than previous work.

Corollary 2 *There exists an algorithm that given an integer $k \in [n]$, a set of n coins with gap parameter $\Delta_k \in (0,1)$, and confidence parameter $\delta \in [0,1)$, finds the set of k most biased coins w.p. $\geq 1-\delta$ using $O\left(\frac{n}{\Delta_k^2} \cdot \log(k/\delta)\right)$ coin tosses and only $\log^*(n)$ rounds of adaptivity.*

2.1.1. ALGORITHM

We design a recursive algorithm, which we term as AGRESSIVE-ELIMINATION, for proving Theorem 1. The pseudo-code is given in Algorithm 1. It takes as input a set $S \subseteq [n]$ of $m \geq k$ *candidate* coins for the top k coins and a parameter r denoting the number of rounds of adaptivity the algorithm can use. In addition, the algorithm is given the confidence parameter $\delta \in (0,1)$ and a lower bound on the gap parameter $\Delta \leq \Delta_k$. Given this input, Algorithm 1 essentially does the following:

1. **Estimation phase:** Toss each coin $O\left(\frac{1}{\Delta^2} \cdot \left(\text{ilog}^{(r)}(m) + \log(k/\delta)\right)\right)$ many times and estimate the bias of each coin.
2. **Elimination phase:** Let S' be the set of $O(\frac{m}{\text{ilog}^{(r-1)}(m)})$ coins with the largest estimated biases. Recursively solve the problem for the set S' in the remaining $r-1$ rounds.

We point out that the estimation phase of the algorithm is allowed to be erroneous, i.e. there might be large deviations between the estimated biases and the true biases for a relatively large fraction of coins. The elimination phase is then designed to be robust to such errors by selecting a suitably large subset for the next round. As rounds progress, the set of candidates for k most biased coins shrinks more and more such that in the last round, the algorithm can estimate the bias of each candidate with high confidence and return the k most biased coins. We should also point that in any round, if the input set S becomes too small, i.e. is of size $O(k)$, then Algorithm 1 bypasses the subsequent rounds and simply runs the 1-round algorithm on this set to recover the answer.

2.1.2. ANALYSIS

Given a target number of rounds r as input, Algorithm 1 clearly uses at most r rounds of adaptivity. In Lemma 3 below we establish correctness of the algorithm, i.e. we show that it correctly returns the k most biased coins with probability at least $1-\delta$. In Appendix B, we bound its query complexity (number of coin tosses). Theorem 1 then follows immediately from these two results.

For simplicity of exposition, in the remainder of this section we assume w.l.o.g. that the coins are indexed such that $p_i \geq p_{i+1}\ \forall i \in [n-1]$, so that the set of k most biased coins is simply $[k]$ (this is used for analysis purposes only; the algorithm does not know this indexing).

Lemma 3 *Suppose S is any subset of the coins $[n]$ with gap parameter $\Delta \leq \Delta_k$, such that $|S| = m$ and $[k] \subseteq S$. For any number of rounds $1 \leq r \leq \log^*(m) - 3$ and any confidence parameter $\delta \in (0,1)$,* Algorithm 1 *correctly returns the set of k most biased coins w.p. $\geq 1-\delta$.*

Algorithm 1 AGRESSIVE-ELIMINATION$(S_r, k, r, \delta, \Delta)$

Input: set $S_r \subseteq [n]$ of coins, number of desired top items k, number of rounds r, confidence parameter $\delta \in (0,1)$, and lower bound on gap parameter $\Delta \leq \Delta_k$
Let $m = m_r = |S_r|$ and $t_r := \frac{2}{\Delta^2} \cdot \left(\text{ilog}^{(r)}(m) + \log(8k/\delta)\right)$.
Toss each coin $i \in S_r$ for t_r times.
For each $i \in S_r$, define $\widehat{p}_i$ as the fraction of times coin i turns up heads.
Sort the coins in S_r in a decreasing order of $\widehat{p}$-values.
if $r = 1$ **then**
  Return: the set of k most biased coins (according to $\widehat{p}$-values).
else
  Let $m_{r-1} := k + \frac{m}{\text{ilog}^{(r-1)}(m)}$ and S_{r-1} be the set of m_{r-1} most biased coins according to $\widehat{p}$.
end if
if $m_{r-1} \leq 2k$ **then**
  Return: AGRESSIVE-ELIMINATION$(S_{r-1}, k, 1, \delta/2, \Delta)$.
else
  Return: AGRESSIVE-ELIMINATION$(S_{r-1}, k, r-1, \delta/2, \Delta)$.
end if

In the remainder of this section, we fix $\varepsilon := \Delta/2$. Before proving Lemma 3, we need the following simple claim. The proof is a simple application of Hoeffding's inequality (see Appendix B).

Claim 1 *For any round $r \geq 1$, and any coin $i \in S_r$,* $\Pr\left(|\widehat{p}_i - p_i| \geq \varepsilon\right) \leq \frac{\delta}{4k \cdot \text{ilog}^{(r-1)}(m)}$.

Proof [of Lemma 3.] The proof is by induction on the number of rounds r. In the following, we use $\mathcal{A}_r$ to denote Algorithm 1 with input number of rounds r.
Base case: For $r = 1$, Claim 1 ensures that for any $i \in S_1$, $\Pr\left(|\widehat{p}_i - p_i| \geq \varepsilon\right) \leq \frac{\delta}{4k \cdot \text{ilog}^{(0)}(m_1)} \leq \frac{\delta}{m_1}$ as $\text{ilog}^{(r-1)}(m_1) = \text{ilog}^{(0)}(m_1) = m_1$ by definition. By taking a union bound over all m_1 coins, we obtain that w.p. $\geq 1 - \delta$, simultaneously for all coins $i \in S_1$, $|\widehat{p}_i - p_i| < \varepsilon$. On the other hand, we know for all $i \in [k]$ and $j \in S_1 \setminus [k]$, $p_i - p_j \geq \Delta = 2\varepsilon$, and hence the returned set of k most biased coins according to $\widehat{p}$-values is the correct answer. This proves the base case.
Induction step: Suppose the lemma is true for all number of rounds smaller than some $r \leq \log^*(m) - 3$; we prove that $\mathcal{A}_r$ also returns the set of k most biased coins w.p. $\geq 1 - \delta$.

Let $I = \{i \in [k] : \widehat{p}_i < p_i - \varepsilon\}$ (underestimated coins in top k) and $J = \{j \in S_r \setminus [k] : \widehat{p}_j > p_j + \epsilon\}$ (overestimated coins in $S_r \setminus [k]$). We know that for all $i \in [k]$ and $j \in S_r \setminus [k]$, $p_i - p_j \geq 2\varepsilon$. As the algorithm identifies a set of $m_{r-1} = k + \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}$ coins with the highest estimated biases (according to $\widehat{p}$) to recurse upon, we have,

$$\Pr\left(\mathcal{A}_r \text{ errs}\right) \leq \Pr\left(|I| > 0\right) + \Pr\left(|J| > \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}\right) + \Pr\left(\mathcal{A}_{r-1} \text{ errs} \mid \mathcal{E}\right) \tag{1}$$

where $\mathcal{E}$ denotes the event that $|I| = 0$ and $|J| \leq \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}$.

In the following, we bound probability of each event above. We first have,

$$\Pr\left(|I| > 0\right) \leq \sum_{i \in [k]} \Pr\left(\widehat{p}_i < p_i - \varepsilon\right) \leq_{\text{Claim 1}} k \cdot \frac{\delta}{4k \cdot \text{ilog}^{(r-1)}(m_r)} \leq \frac{\delta}{4} \tag{2}$$

where the last inequality is true because $\text{ilog}^{(r-1)}(m_r) \geq 1$.

We next bound the second term. For all $j \in S_r \setminus [k]$, we define an indicator random variable Y_j which is 1 iff $\widehat{p}_j > p_j + \varepsilon$. We further define $\mathsf{Y} := \sum_j \mathsf{Y}_j$. We have,

$$\mathbb{E}[\mathsf{Y}] = \sum_j \mathbb{E}[\mathsf{Y}_j] = \sum_j \Pr(\widehat{p}_j > p_j + \varepsilon) \leq_{\text{Claim 1}} \sum_j \frac{\delta}{4k \cdot \text{ilog}^{(r-1)}(m_r)} \leq \frac{\delta \cdot m_r}{4 \cdot \text{ilog}^{(r-1)}(m_r)}$$

Notice that $\mathsf{Y} = |J|$; hence,

$$\Pr\left(|J| > \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}\right) \leq \Pr\left(\mathsf{Y} > \frac{4}{\delta} \cdot \mathbb{E}[\mathsf{Y}]\right) \leq \frac{\delta}{4} \tag{3}$$

where the second inequality is by Markov bound.

Finally, we calculate the probability of error of $\mathcal{A}_{r-1}$ conditioned on that none of the two events above happens (i.e., the event $\mathcal{E}$). In this case, we have $[k] \subseteq S_{r-1}$ and that $\Delta \leq \Delta_k$. As $r \leq \log^*(m_r) - 3$ (by the lemma statement), we have $r - 1 \leq (\log^*(m_r) - 1) - 3 = \log^*(\log m_r) - 3 \leq \log^*(m_{r-1}) - 3$. Therefore, the input to $\mathcal{A}_{r-1}$ satisfies the assumptions in the lemma statement as well and since the confidence parameter for $\mathcal{A}_{r-1}$ is $\delta/2$, by induction, we obtain that $\Pr(\mathcal{A}_{r-1} \text{ errs} \mid \mathcal{E}) \leq \delta/2$. By plugging in this bound, together with Eq (2) and Eq (3) to Eq (1), we obtain that $\mathcal{A}_r$ is also a δ-error algorithm. This proves the induction step. ■

2.2. A Lower Bound for the k Most Biased Coins Problem

We further establish a tight lower bound on the tradeoff between the round complexity and query complexity of any algorithm for the k most biased coins problem:

Theorem 4 *For any parameter $\Delta \in (0, \frac{1}{2})$ and any integers $n, k \geq 1$, there exists a distribution $\mathcal{D}$ on instances of the k most biased coins problem with n coins and gap parameter $\Delta_k = \Delta$ such that for any integer $r \geq 1$, any r-round algorithm that finds the k most biased coins in instances sampled from $\mathcal{D}$ w.p. at least $3/4$ has query complexity $\Omega\left(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(\frac{n}{k})\right)$.*

An important corollary of Theorem 4 is that achieving the optimal worst-case query complexity requires (slightly) *super-constant* round complexity:

Corollary 5 *For any parameter $\Delta \in (0, \frac{1}{2})$ and any integers $n, k \geq 1$, there exists a distribution on instances of the k most biased coins problem with n coins and gap parameter $\Delta_k = \Delta$ such that any $(\frac{1}{4})$-error algorithm for finding the top k most biased coins that uses $O(n/\Delta^2)$ coin tosses on these instances requires round complexity at least $(\log^*(n) - \log^*(\Theta(\log^* n)))$.*

We remark that by Corollary 2, the bound on the round complexity of algorithms in Corollary 5 is tight up to an extremely small *additive* factor of $\log^*(\Theta(\log^* n))$ when k is a constant.

2.2.1. PROOF SKETCH FOR $k = 1$

Here we give a detailed sketch of the proof of Theorem 4 for the case $k = 1$, i.e. the case of finding the most biased coin. A complete proof and generalization to all values of k is in Appendix C.

Fix any arbitrary value $\Delta \in (0, \frac{1}{2})$ (possibly a function of n) and a constant $p < 1 - \Delta$. Our hard input distribution is defined based on the parameters Δ and p:

Distribution $\mathcal{D}_n^{\Delta,p}$ (A hard input distribution on n coins with gap parameter $\Delta_1 = \Delta$).

- Sample an index $i^\star \in [n]$ uniformly at random.
- Let $p_i = \begin{cases} p + \Delta & \text{if } i = i^\star \\ p & \text{otherwise} \end{cases} \quad \forall i \in [n]$.
- Return the coins $[n]$ with biases $\{p_i\}_{i=1}^n$.

It is immediate to see that in any instance of $\mathcal{D}_n^{\Delta,p}$, $\Delta_1 = \Delta$. Moreover, one can see that finding the most biased coin in this family of instances is equivalent to determining the value of $i^\star$. We use this to prove a lower bound on the query complexity of any algorithm on these instances.

Define the recursive function $e(r) = e(r-1) + o(1/r^2)$ with $e(1) = 0$. Our main result in this section is Lemma 6 below, for which we give a detailed proof sketch. Theorem 4 (for the case $k = 1$) then follows from Lemma 6 (see Appendix C for details).

Lemma 6 *Fix any integers $n, r \geq 1$. Suppose $\mathcal{A}_r$ is an r-round algorithm that given an instance sampled from $\mathcal{D}_n^{\Delta,p}$, correctly outputs the most biased coin w.p. $\geq \frac{2}{3} + e(r)$. Then $\mathcal{A}_r$ must have query complexity $\Omega(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(n))$.*

In the following, fix $n, r \geq 1$ and algorithm $\mathcal{A}_r$ as in Lemma 6. Note that by an averaging argument, we can assume w.l.o.g that $\mathcal{A}_r$ is deterministic (see Appendix C for a formal proof).

Before proving Lemma 6, let us first set up some notation. Let S_1 denote the multi-set of coins tossed in the first round by $\mathcal{A}_r$. Let s_1 denote the size of S_1 counting the multiplicities. We define the *outcome profile* of S_1 as the s_1-dimensional tuple $T = ((i_1, \theta_1), (i_2, \theta_2), \ldots, (i_{s_1}, \theta_{s_1}))$, whereby for any $j \in [s_1]$, i_j and θ_j, denote, respectively, the index of the j-th coin in S_1 and the outcome of its toss, i.e., heads or tails. We use I to denote the random variable for the index $i^\star$ in $\mathcal{D}_n^{\Delta,p}$, and T to denote the random variable for the vector T. We further use Θ_j, for any $j \in [s_1]$, to denote the random variable for θ_j defined above. In order to prove Lemma 6, we will need the following key lemma that bounds the "information" revealed about the index $i^\star$ in *the first round* based on the number of coin tosses conducted by $\mathcal{A}_r$ in this round:

Lemma 7 $\mathbb{I}(\mathsf{I}\,;\mathsf{T}) = O(s_1 \cdot \Delta^2/n)$.

The proof of Lemma 7 involves: *(i)* separating the information revealed by each coin toss $(i_j, \theta_j) \in T$ using the chain rule of mutual information and the fact that for any $j \in [s_1]$, Θ_j and $\Theta \setminus \Theta_j$ are *independent conditioned on* I (see Appendix C, Eq (13)); and *(ii)* using the KL-divergence between distribution of $\text{dist}(\Theta_j)$ and $\text{dist}(\Theta_j \mid \mathsf{I})$ to bound the information revealed by each coin toss by $O(\Delta^2/n)$. A formal proof is given in Appendix C.

Proof [(Sketch) for Lemma 6.] The proof is by induction on the number of rounds. The base case of this induction, which asserts that the query complexity of $\mathcal{A}_1$ is $\Omega(\frac{n}{\Delta^2} \cdot \log n)$, follows from

Lemma 7 and Fano's inequality (see Fact 2) and is provided in Appendix C. Now assume inductively that Lemma 6 is true for all integers smaller than r and we want to prove this for r-round algorithms. Our proof of the induction step is by contradiction. We show that if there exists an algorithm $\mathcal{A}_r$ with smaller query complexity than the bound stated in Lemma 6 for $\mathcal{D}_n^{\Delta,p}$, then there also exists an $(r-1)$-round algorithm $\mathcal{A}_{r-1}$ with smaller query complexity than the corresponding bound for the distribution $\mathcal{D}_m^{\Delta,p}$ for some appropriately chosen $m \leq n$.

Intuitively, Lemma 7 implies that if the number of coin tosses in the first round is small, then the outcome profile T, on average, does not reveal much information about the identity of the most biased coin, i.e. I. More formally, if we assume (by way of contradiction) that the query complexity of $\mathcal{A}_r$ is $o(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(n))$, then we have,

$$\mathbb{H}(\mathsf{I} \mid \mathsf{T}) = \mathbb{H}(\mathsf{I}) - \mathbb{I}(\mathsf{I}\,;\mathsf{T}) =_{\text{Fact 1-(a)}} \log n - \mathbb{I}(\mathsf{I}\,;\mathsf{T}) =_{\text{Lemma 7}} \log n - o(\text{ilog}^{(r)}(n)/r^4) \quad (4)$$

where in the last part we used the fact that s_1 is at most the query complexity of $\mathcal{A}_r$. Now consider any fixed possible outcome profile T for $\mathcal{A}_r$ in round one, i.e., any possible value for T. We say that T is *uninformative* iff $\mathbb{H}(\mathsf{I} \mid \mathsf{T} = T) = \log n - o(\text{ilog}^{(r)}(n)/r^2)$. Roughly speaking, whenever the outcome profile in the first round is uninformative, the algorithm is quite "uncertain" about the identity of the most biased coin, and hence needs to find it among a large pool of candidate coins in the next $(r-1)$ rounds. This we argue is not possible as by induction hypothesis as the available budget is not large enough to solve the problem in $(r-1)$ rounds on such a large domain.

We can show that our assumption on the query complexity of $\mathcal{A}_r$ implies that there exists an uninformative outcome profile T_{ui} in the first round which still results in a correct output by $\mathcal{A}_r$ in the subsequent rounds, i.e., $\Pr(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}) \leq \delta + o(1/r^2)$ (see Claim 7 in Appendix C).

Fix the uninformative profile T_{ui} and define $\psi := \text{dist}(\mathsf{I} \mid \mathsf{T} = T_{\mathsf{ui}})$. Using the fact that the conditional entropy $\mathbb{H}(\mathsf{I} \mid \mathsf{T} = T_{\mathsf{ui}})$ is quite close to $\log n$ (i.e., to the entropy of the uniform distribution), the distribution ψ can be expressed as a convex combination of distributions $\psi_0, \psi_1, \ldots, \psi_k$, i.e., $\psi = \sum_i q_i \cdot \psi_i$ (for $\sum_i q_i = 1$) such that $q_0 = o(1/r^2)$ and for all $i \geq 1$,

$$|\text{supp}(\psi_i)| \geq 2^{\left(\log n - o(\text{ilog}^{(r)}(n))\right)} \geq \frac{n}{\left(\text{ilog}^{(r-1)}(n)\right)^{o(1)}} \quad (5)$$

$$\|\psi_i - \mathcal{U}_i\|_{tvd} = o(1/r^2) \quad (6)$$

where $\mathcal{U}_i$ is the uniform distribution on $\text{supp}(\psi_i)$ (see Lemma 8 in Appendix A and discussion before Eq (17) in Appendix C). With this notation,

$$\delta + o(1/r^2) \geq_{\text{Claim 7}} \Pr(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}) = \sum_i q_i \cdot \Pr(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}, \mathsf{I} \sim \psi_i)$$

As $q_0 = o(1/r^2)$, by an averaging argument, we have that there exists a distribution ψ_i for some $i \geq 1$ such that $\Pr(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}, \mathsf{I} \sim \psi_i) \leq \delta + o(1/r^2)$. W.l.o.g. let this distribution be ψ_1 and define $m := |\text{supp}(\psi_1)|$. We will need the following claim:

Claim 2 *There exists a deterministic $(r-1)$-round $\left(\delta + o(1/r^2)\right)$-error algorithm for the best coin problem on $\mathcal{D}_m^{\Delta,p}$ with query complexity at most equal to the query complexity of $\mathcal{A}_r$ on $\mathcal{D}_n^{\Delta,p}$.*

Proof Let $\mathcal{A}_{r,T_{\mathsf{ui}}}$ be an $(r-1)$-round algorithm obtained by running $\mathcal{A}_r$ from the second round onwards assuming that the outcome profile in the first round was T_{ui}. We use $\mathcal{A}_{r,T_{\mathsf{ui}}}$ to design a randomized algorithm $\mathcal{A}'$ for $\mathcal{D}_m^{\Delta,p}$.

Given any instance sampled from $\mathcal{D}_m^{\Delta,p}$, $\mathcal{A}'$ maps $[m]$ to $\text{supp}(\psi_1)$ (using any arbitrary bijection). Next, it runs $\mathcal{A}_{r,T_{\text{ui}}}$ as follows: if $\mathcal{A}_{r,T_{\text{ui}}}$ chooses to toss a coin in $\text{supp}(\psi_1)$, $\mathcal{A}'$ also chooses the corresponding coin in $[m]$; otherwise, if $\mathcal{A}_{r,T_{\text{ui}}}$ chooses to toss a coin in $[n] \setminus \text{supp}(\psi_1)$, $\mathcal{A}'$ simply tosses a coin from the distribution $\mathcal{B}(p)$ and returns the result to $\mathcal{A}_{r,T_{\text{ui}}}$. Finally, if $\mathcal{A}_{r,T_{\text{ui}}}$ outputs a coin from $\text{supp}(\psi_1)$, $\mathcal{A}'$ returns the corresponding coin in $[m]$ and otherwise $\mathcal{A}'$ simply return an arbitrary coin in $[m]$ as the answer.

It is trivially true that the query complexity of $\mathcal{A}'$ is at most that of $\mathcal{A}$. Hence, in the following we prove the correctness of $\mathcal{A}'$. Let $\mathcal{D}'$ be the distribution of underlying instances on $[n]$ created by $\mathcal{A}'$. Let $\mathcal{U}_1$ be the uniform distribution on $\text{supp}(\psi_1)$. It is straightforward to verify that $\mathcal{D}' = \mathcal{D}_n^{\Delta,p} \mid \mathsf{I} \sim \mathcal{U}_1$, and that $\mathcal{D}'$ is a deterministic function of I. As such,

$$\begin{aligned}\Pr_{\mathcal{D}_m^{\Delta,p}}\left(\mathcal{A}' \text{ errs}\right) &= \Pr_{\mathcal{D}_n^{\Delta,p}}\left(\mathcal{A}_{r,T_{\text{ui}}} \text{ errs} \mid \mathsf{I} \sim \mathcal{U}_1\right) \leq_{\text{Fact 5}} \Pr_{\mathcal{D}_n^{\Delta,p}}\left(\mathcal{A}_{r,T_{\text{ui}}} \text{ errs} \mid \mathsf{I} \sim \psi_1\right) + \|\psi_1 - \mathcal{U}_1\|_{tvd} \\ &=_{\text{Eq (18)}} \Pr_{\mathcal{D}_n^{\Delta,p}}\left(\mathcal{A}_r \text{ errs} \mid \mathsf{I} \sim \psi_1, \mathsf{T} = T_{\text{ui}}\right) + o(1/r^2) \leq \delta + o(1/r^2)\end{aligned}$$

To finalize the proof, note that by an averaging argument, there exists a fixing of the randomness in $\mathcal{A}'$ that results in the same error guarantee. But by fixing the randomness in $\mathcal{A}'$ we obtain a deterministic $(r-1)$-round algorithm $\mathcal{A}''$ that errs on $\mathcal{D}_m^{\Delta,p}$ w.p. at most $\delta + o(1/r^2)$. ■

We can now conclude the proof of Lemma 6. By Claim 2, there exists an $(r-1)$-round algorithm $\mathcal{A}_{r-1}$ that errs w.p. at most $\delta + o(1/r^2) = 1/3 - e(r) + o(1/r^2) = 1/3 - e(r-1)$ on instances of $\mathcal{D}_m^{\Delta,p}$ with (at most) the same query complexity as $\mathcal{A}_r$. But by induction hypothesis (as $\mathcal{A}_{r-1}$ is an $(r-1)$-round algorithm), we know that the query complexity of $\mathcal{A}_{r-1}$ is

$$\begin{aligned}\Omega\left(\frac{m}{\Delta^2 \cdot (r-1)^4} \cdot \text{ilog}^{(r-1)}(m)\right) &=_{\text{Eq (5)}} \Omega\left(\frac{1}{\Delta^2 \cdot (r-1)^4} \cdot \frac{n}{\left(\text{ilog}^{(r-1)}(n)\right)^{o(1)}} \cdot \text{ilog}^{(r-1)}(n)\right) \\ &= \Omega\left(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(n)\right) \\ &\qquad\qquad \text{(as } \text{ilog}^{(r)}(n) = \log(\text{ilog}^{(r-1)}(n))\text{)}\end{aligned}$$

which is in contradiction with the bound of $o\left(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(n)\right)$ on the query complexity of $\mathcal{A}_r$ and hence $\mathcal{A}_{r-1}$. This completes the proof (sketch) of Lemma 6. ■

3. Top-k Ranking from Pairwise Comparisons

The problem of ranking from pairwise comparisons arises in many applications including sports rankings, recommender systems, crowdsourcing and others, and has received increasing attention in recent years (Gleich and Lim, 2011; Jamieson and Nowak, 2011; Negahban et al., 2012; Busa-Fekete et al., 2013; Rajkumar and Agarwal, 2014; Chen and Suh, 2015; Shah and Wainwright, 2015; Jang et al., 2016; Heckel et al., 2016; Braverman et al., 2016). Here there are n items, and an unknown preference matrix $\mathbf{P} \in [0,1]^{n \times n}$ satisfying $P_{ij} + P_{ji} = 1$ for all $i, j \in [n]$, such that whenever items i and j are compared, item i beats item j with probability P_{ij} and j beats i with

probability $P_{ji} = 1 - P_{ij}$. Previous studies have often made strong assumptions on the preference matrix $\mathbf{P}$; here we consider a very general setting where we make no assumptions on $\mathbf{P}$.

We are interested in the problem of identifying the top-k items according to the *Borda score*, which for item i is defined as the probability that i beats another item j drawn uniformly at random:

$$\tau_i = \frac{1}{n-1} \sum_{j \neq i} P_{ij} \, .$$

Ranking according to Borda scores is very natural and encompasses several special cases. For example, Chen and Suh (2015) and Jang et al. (2016) assume $\mathbf{P}$ follows a *Bradley-Terry-Luce* (BTL) model, under which there is a 'score' vector $\mathbf{w} \in \mathbb{R}^n_{++}$ such that $P_{ij} = \frac{w_i}{w_i + w_j} \; \forall i, j$, and seek to identify the top-k items according to the scores w_i; it can be verified that for such $\mathbf{P}$, ranking by Borda scores is equivalent to ranking by the scores w_i. Feige et al. (1994); Braverman et al. (2016) assume $\mathbf{P}$ follows a *noisy permutation* model[3], under which there is a permutation $\sigma \in S_n$ and noise parameter $p \in [0, \frac{1}{2})$ such that $P_{ij} = 1 - p$ if $\sigma(i) < \sigma(j)$ and $P_{ij} = p$ otherwise, and seek to identify the top-k items according to σ; again, it can be verified that for such $\mathbf{P}$, ranking by Borda scores is equivalent to ranking according to σ. Here we make no such assumptions on $\mathbf{P}$. The general problem of top-k ranking from pairwise comparisons under Borda scores has been considered recently by Busa-Fekete et al. (2013), Shah and Wainwright (2015) and Heckel et al. (2016); however, these studies are either in the non-adaptive setting (where pairwise comparisons are observed for randomly drawn item pairs) or in the fully adaptive setting (where one can actively query pairs to be compared with no limit on the number of rounds of adaptivity). Here we consider the limited-adaptivity setting, and show that our results for the coin problem studied in Section 2 also yield an optimal algorithm and corresponding lower bound for top-k ranking in this setting.

In order to apply the algorithm of Section 2 to the top-k ranking problem, observe that we can view each item i as a coin with bias p_i equal to its Borda score τ_i. In order to toss coin i, we simply select another item $j \in [n] \setminus \{i\}$ uniformly at random, and compare i and j; clearly, this results in a win for item i (heads outcome) with probability τ_i. Thus, the AGRESSIVE-ELIMINATION algorithm from Section 2 applies directly, with $O(\frac{n}{\Delta_k^2} \log k)$ pairwise comparisons and $\log^*(n)$ rounds of adaptivity. Thus we require fewer comparisons than in the passive setting, and fewer rounds of adaptivity than the previous active algorithms of Busa-Fekete et al. (2013) and Heckel et al. (2016) (see Table 2).

For the lower bound, we need to be more careful, since not all collections of n coins with biases $p_1, \ldots, p_n$ can be realized as the Borda scores of a preference matrix $\mathbf{P}$ (to see this, consider $n = 2$ coins with biases $p_1 = 0.9$ and $p_2 = 0.8$; these clearly cannot be realized as Borda scores of a preference matrix $\mathbf{P}$, due to the constraint that $P_{12} = 1 - P_{21}$). However, we show that a simple reduction allows us to convert the hard instances for the coin problem used in the lower bound of Theorem 4 (and Corollary 5) into hard instances for the pairwise comparison problem, thereby leading to a similar tight lower bound for the top-k ranking problem; we give details in Appendix E.

3. The results of Feige et al. (1994); Braverman et al. (2016) can be further extended to a slightly more general model where $\mathbf{P}$ is such that there is a permutation $\sigma \in S_n$ and noise parameter $p \in [0, \frac{1}{2})$ such that $P_{ij} \geq 1 - p$ if $\sigma(i) < \sigma(j)$ and $P_{ij} \leq p$ otherwise.

4. Conclusion

We considered the question of learning with limited rounds of adaptivity in the context of several learning problems: the k most biased coins problem, the closely related k best arms identification problem in stochastic multi-armed bandits (MABs), and top-k ranking from pairwise comparisons. We developed an algorithm which applies to all these problems, and that achieves the optimal worst-case query complexity for these problems in just $\log^*(n)$ rounds of adaptivity, in contrast with previous results which require $\Omega(\log n)$ rounds. We also gave a matching lower bound showing that any algorithm achieving this query complexity must use $\Omega(\log^*(n))$ rounds of adaptivity.

In recent years, there also has been much interest in the MAB literature (and increasingly, in the ranking literature) in adaptive algorithms whose query complexity depends not only on the gap Δ_k between the k-th and $(k+1)$-th best items, but also on the gaps of other items (see Tables 1–2). The optimal query complexity as a function of these parameters, referred to as *instance-wise optimality*, is not yet fully understood despite significant progress in recent years; see, e.g., (Chen and Li, 2015; Chen et al., 2017) and references therein. The round complexity of the state-of-the-art algorithms (Karnin et al., 2013; Jamieson et al., 2013; Chen and Li, 2015) for this setting has at least a logarithmic dependence on n, as they call the $\log(n)$-round HALVING algorithm of Even-Dar et al. (2006) as a subroutine. It is possible to reduce the round complexity of these algorithms to have a $\log^*$ dependence on n by using an (ϵ, δ)-PAC version[4] of our algorithm as a subroutine instead of HALVING. This $\log^*(n)$ dependence is also necessary as our lower bound of $\Omega(\log^*(n))$ rounds also applies to the case of instance-wise algorithms. But the round complexity of these algorithms also depends on the gaps Δ_i's, and it is not clear whether the dependence on these Δ_i's is necessary. Closing this gap remains an interesting open question; its resolution would further enhance our understanding of the role of the degree of adaptivity in designing learning algorithms.

Acknowledgments

Sepehr Assadi and Sanjeev Khanna are supported in part by National Science Foundation grants CCF-1552909, CCF-1617851, and IIS-1447470.

4. Here, the goal is to return a set of k coins whose biases are at least $p_{[k]} - \epsilon$ with probability $\geq 1 - \delta$, for some parameters ϵ, δ. Our algorithm can be easily extended to this (ϵ, δ)-PAC setting.

References

Miklos Ajtai, János Komlos, William L Steiger, and Endre Szemerédi. Deterministic selection in $o(\log \log n)$ parallel time. In *STOC*, 1986.

Noga Alon and Yossi Azar. Sorting, approximate sorting, and searching in rounds. *SIAM J. Discrete Math.*, 1(3):269–280, 1988.

Sepehr Assadi, Sanjeev Khanna, and Yang Li. On estimating maximum matching size in graph streams. In *SODA*, 2017.

Jean-Yves Audibert and Sébastien Bubeck. Best Arm Identification in Multi-Armed Bandits. In *COLT*, 2010.

Béla Bollobás and Graham Brightwell. Parallel selection with high probability. *SIAM Journal on Discrete Mathematics*, 3(1):21–31, 1990.

Béla Bollobás and Andrew Thomason. Parallel sorting. *Discrete Applied Mathematics*, 6(1):1–11, 1983.

Mark Braverman, Jieming Mao, and S. Matthew Weinberg. Parallel Algorithms for Select and Partition with Noisy Comparisons. In *STOC*, 2016.

Sébastien Bubeck, Tengyao Wang, and Nitin Viswanathan. Multiple identifications in multi-armed bandits. In *ICML*, 2013.

Róbert Busa-Fekete, B. Szorenyi, Weiwei Cheng, Paul Weng, and E. Hullermeier. Top-k selection based on adaptive sampling of noisy preferences. In *ICML*, 2013.

Karthekeyan Chandrasekaran and Richard Karp. Finding a most biased coin with fewest flips. In *Journal of Machine Learning Research*, volume 35, pages 394–407, 2014.

Lijie Chen and Jian Li. On the Optimal Sample Complexity for Best Arm Identification. *arXiv preprint arXiv:1511.03774*, 2015. URL http://arxiv.org/abs/1511.03774.

Lijie Chen, Jian Li, and Mingda Qiao. Nearly Instance Optimal Sample Complexity Bounds for Top-k Arm Selection. *arXiv preprint arXiv:1702.03605*, 2017. URL https://arxiv.org/abs/1702.03605.

Yuxin Chen and Changho Suh. Spectral MLE: Top-k rank aggregation from pairwise comparisons. In *ICML*, 2015.

Herman Chernoff. *Sequential analysis and optimal design*. SIAM, 1972.

Richard Cole. Parallel merge sort. *SIAM J. Comput.*, 17(4):770–785, 1988.

Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006. ISBN 978-0-471-24195-9.

Susan Davidson, Sanjeev Khanna, Tova Milo, and Sudeepa Roy. Top-k and clustering with noisy comparisons. *ACM Transactions on Database Systems (TODS)*, 39(4):35, 2014.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. PAC Bounds for Multi-Armed Bandit and Markov Decision Processes. In *COLT*, 2002.

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

Uriel Feige, Prabhakar Raghavan, David Peleg, and Eli Upfal. Computing with Noisy Information. *SIAM Journal on Computing*, 23(5):1001–1018, 1994. doi: 10.1137/S0097539791195877.

Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. In *NIPS*, 2012.

Alison L Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International statistical review*, 70(3):419–435, 2002.

David F Gleich and Lek-heng Lim. Rank aggregation via nuclear norm minimization. In *KDD*, pages 60–68, 2011.

Reinhard Heckel, Nihar B. Shah, Kannan Ramchandran, and Martin J Wainwright. Active Ranking from Pairwise Comparisons and when Parametric Assumptions Dont Help. *arXiv preprint arXiv:1606.08842*, 2016. URL http://arxiv.org/abs/1606.08842.

Eshcar Hillel, Zohar Shay Karnin, Tomer Koren, Ronny Lempel, and Oren Somekh. Distributed exploration in multi-armed bandits. In *NIPS*, 2013.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sebastien Bubeck. On Finding the Largest Mean Among Many. *arXiv preprint arXiv:1306.3917v1*, 2013. URL http://arxiv.org/abs/1306.3917.

Kevin Jamieson, Daniel Haas, and Ben Recht. The Power of Adaptivity in Identifying Statistical Alternatives. In *NIPS*, 2016.

Kevin G Jamieson and Robert D. Nowak. Active Ranking using Pairwise Comparisons. In *NIPS*, 2011.

Minje Jang, Sunghyun Kim, Changho Suh, and Sewoong Oh. Top-k Ranking from Pairwise Comparisons: When Spectral Ranking is Optimal. *arXiv preprint arXiv:1603.04153*, 2016. URL http://arxiv.org/abs/1603.04153.

Kwang-Sung Jun, Kevin Jamieson, Robert Nowak, and Xiaojin Zhu. Top Arm Identification in Multi-Armed Bandits with Batch Arm Pulls. In *AISTATS*, 2016.

Shivaram Kalyanakrishnan and Peter Stone. Efficient Selection of Multiple Bandit Arms: Theory and Practice. In *ICML*, 2010.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. PAC Subset Selection in Stochastic Multi-armed Bandits. In *ICML*, 2012.

Zohar Karnin, Tomer Koren, and Oren Somekh. Almost Optimal Exploration in Multi-Armed Bandits. In *ICML*, 2013.

Richard M. Karp and Robert Kleinberg. Noisy binary search and its applications. In *SODA*, 2007.

Emilie Kaufmann, Olivier Cappé, and Aurlien Garivier. On the Complexity of Best-Arm Identification in Multi-Armed Bandit Models. *Journal of Machine Learning Research*, 17:1–42, 2016.

Matthew L Malloy, Gongguo Tang, and Robert D. Nowak. Quickest search for a rare distribution. In *Information Sciences and Systems (CISS)*. IEEE, 2012.

Shie Mannor and John N Tsitsiklis. The Sample Complexity of Exploration in the Multi-Armed Bandit Problem. *Journal of Machine Learning Research*, 5:623–648, 2004.

Sahand Negahban, Sewoong Oh, and Devavrat Shah. Iterative ranking from pair-wise comparisons. In *NIPS*, 2012.

Vianney Perchet, Philippe Rigollet, Sylvain Chassang, and Erik Snowberg. Batched bandit problems. In *COLT*, 2015.

Nicholas Pippenger. Sorting and selecting in rounds. *SIAM J. Comput.*, 16(6):1032–1038, 1987.

Arun Rajkumar and Shivani Agarwal. A statistical convergence perspective of algorithms for rank aggregation from pairwise data. In *ICML*, 2014.

Nihar B. Shah and Martin J. Wainwright. Simple, Robust and Optimal Ranking from Pairwise Comparisons. *arXiv preprint arXiv:1512.08949*, 2015. URL http://arxiv.org/abs/1512.08949.

Leslie G. Valiant. Parallelism in comparison problems. *SIAM Journal on Computing*, 4(3):348–355, 1975.

Appendix A. Tools From Information Theory

Our lower bound proof relies on basic concepts from information theory. For a broader introduction to the field, and proofs of the claims below, we refer the reader to the excellent text by Cover and Thomas (2006).

We denote the *Shannon Entropy* of a random variable A by $\mathbb{H}(\mathsf{A})$ and the *mutual information* of two random variables A and B by $\mathbb{I}(\mathsf{A}\,;\mathsf{B}) = \mathbb{H}(\mathsf{A}) - \mathbb{H}(\mathsf{A} \mid \mathsf{B}) = \mathbb{H}(\mathsf{B}) - \mathbb{H}(\mathsf{B} \mid \mathsf{A})$. Additionally, $H_2(\cdot)$ denotes the binary entropy function: for any real number $\delta \in (0,1)$, $H_2(\delta) := \mathbb{H}(\mathsf{A})$ where $\mathsf{A} \sim \mathcal{B}(\delta)$. We use the following basic facts about entropy and mutual information in this paper. The proofs can be found in Cover and Thomas (2006), Chapter 2.

Fact 1 *Let* A*,* B*, and* C *be three (possibly correlated) random variables.*

(a) $0 \le \mathbb{H}(\mathsf{A}) \le \log|\mathsf{A}|$*, and* $\mathbb{H}(\mathsf{A}) = \log|\mathsf{A}|$ *iff* A *is uniformly distributed over its support.*

(b) $\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) \ge 0$*. The equality holds iff* A *and* B *are* independent *conditioned on* C*.*

(c) Conditioning can only drop the entropy: $\mathbb{H}(\mathsf{A} \mid \mathsf{B},\mathsf{C}) \le \mathbb{H}(\mathsf{A} \mid \mathsf{B})$*. The equality holds iff* $\mathsf{A} \perp \mathsf{C} \mid \mathsf{B}$*.*

(d) Chain rule of mutual information: $\mathbb{I}(\mathsf{A},\mathsf{B}\,;\mathsf{C}) = \mathbb{I}(\mathsf{A}\,;\mathsf{C}) + \mathbb{I}(\mathsf{B}\,;\mathsf{C} \mid \mathsf{A})$*.*

The following Fano's inequality states that if a random variable A can be used to estimate the value of another random variable B, then A should "consume" most of B's entropy.

Fact 2 (Fano's inequality) *Let* A,B *be random variables and* f *be a function that given* A *predicts a value for* B*. If* $\Pr\left(f(\mathsf{A}) \neq \mathsf{B}\right) \le \delta$*, then* $\mathbb{H}(\mathsf{B} \mid \mathsf{A}) \le H_2(\delta) + \delta \cdot \log|\mathsf{B}|$*.*

For two distributions μ and ν over the same probability space, the *Kullback-Leibler divergence* between μ and ν is defined as $\mathbb{D}(\mu \,||\, \nu) := \mathbb{E}_{a\sim\mu}\left[\log \frac{\Pr_\mu(a)}{\Pr_\nu(a)}\right]$. For our proofs, we need the following relation between mutual information and KL-divergence.

Fact 3 *For random variables* $\mathsf{A},\mathsf{B},\mathsf{C}$*,*

$$\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) = \mathop{\mathbb{E}}_{(b,c)\sim\mathrm{dist}(\mathsf{B},\mathsf{C})}\Big[\mathbb{D}(\mathrm{dist}(\mathsf{A} \mid \mathsf{C}=c) \,||\, \mathrm{dist}(\mathsf{A} \mid \mathsf{B}=b, \mathsf{C}=c))\Big].$$

The following fact can be proven by bounding the KL-divergence by χ^2-distance (see, e.g., Gibbs and Su (2002), Theorem 5).

Fact 4 *For any two parameters* $0 < p, q < 1$*,*

$$\mathbb{D}(\mathcal{B}(p) \,||\, \mathcal{B}(q)) \le \frac{(p-q)^2}{q \cdot (1-q)}$$

We denote the *total variation distance* between two distributions μ and ν over the same probability space Ω by $\|\mu - \nu\|_{tvd} = \frac{1}{2} \cdot \sum_{x\in\Omega} |\Pr_\mu(x) - \Pr_\nu(x)|$. We have,

Fact 5 *Suppose* μ *and* ν *are two distributions for an event* $\mathcal{E}$*, then,* $\Pr_\mu(\mathcal{E}) \le \Pr_\nu(\mathcal{E}) + \|\mu - \nu\|_{tvd}$*.*

Finally, we use the following auxiliary lemma that allows us to decompose the distribution of any random variable with high entropy to a convex combination of a small number of near uniform distributions plus a low probability "noise term". A similar lemma was proven in (the full version of) Assadi et al. (2017) (see Lemma 2.4). We provide a self-contained proof of this lemma here for completeness.

Lemma 8 *Let* $\mathsf{A} \sim \mathcal{D}$ *be a random variable on* $[n]$ *with* $\mathbb{H}(\mathsf{A}) \geq \log n - \gamma$ *for some* $\gamma \geq 1$*. For any* $\varepsilon > \exp(-\gamma)$*, there exists* $\ell + 1$ *distributions* $\psi_0, \psi_1, \ldots, \psi_\ell$ *on* $[n]$ *along with* $\ell + 1$ *probabilities* $p_0, p_1, \ldots, p_\ell$ *(*$\sum_i p_i = 1$*) for some* $\ell = O(\gamma/\varepsilon^3)$ *such that* $\mathcal{D} = \sum_{i=1}^{\ell} p_i \cdot \psi_i$*,* $p_0 = O(\varepsilon)$*, and for any* $i \geq 1$*,*

1. $\log |\text{supp}(\psi_i)| \geq \log n - \gamma/\varepsilon$.
2. $\|\psi_i - \mathcal{U}_i\|_{tvd} = O(\varepsilon)$ *where* $\mathcal{U}_i$ *denotes the uniform distribution on* $\text{supp}(\psi_i)$.

Proof We partition $\text{supp}(\mathcal{D})$ into ℓ' sets $S_0, S_1, \ldots, S_{\ell'}$ for $\ell' = O(\gamma/\varepsilon^3)$ that S_0 contains every element $a \in \text{supp}(\mathcal{D})$ such that either $\Pr(\mathsf{A} = a) \leq \varepsilon/n$ or $\Pr(\mathsf{A} = a) \geq 2^{2\gamma/\varepsilon}/n$, and for each $i \geq 1$, S_i contains every element $a \in \text{supp}(\mathcal{D})$ where $(1+\varepsilon)^{-(i+1)} \leq \Pr(\mathsf{A} = a) < (1+\varepsilon)^{-i}$. For any $i \geq 1$, we say that a set S_i is *large* if $|S_i| \geq 2^{\left(\log n - \frac{\gamma}{\varepsilon}\right)}$ and is *small* otherwise. Let $\mathcal{L}$ (resp. $\mathcal{S}$) denote the set of all elements that belong to a large set (resp. a small set). Moreover, let ℓ be the number of large sets and without loss of generality, assume $S_1, \ldots, S_\ell$ are these large sets.

We define the $\ell + 1$ distributions in the lemma statement as follows. Let ψ_0 be the distribution $\mathcal{D}$ conditioned on A being in $S_0 \cup \mathcal{S}$, and let $p_0 := \Pr_{\mathcal{D}}(\mathsf{A} \in S_0 \cup \mathcal{S})$. For each $i \geq 1$, let ψ_i be the distribution $\mathcal{D}$ conditioned on A being in S_i, i.e., the i-th large set and let $p_i := \Pr_{\mathcal{D}}(\mathsf{A} \in S_i)$.

By construction, we have $\mathcal{D} = \sum_i p_i \cdot \psi_i$. Moreover, for each $i \geq 1$, since the support of ψ_i is a large set, we have $\log |\text{supp}(\psi_i)| \geq \left(\log n - \frac{\gamma}{\varepsilon}\right)$. Finally, since each $a \in \text{supp}(\psi_i)$ has $\Pr_{\mathcal{D}}(\mathsf{A} = a) \in [(1+\varepsilon)^{-(i+1)}, (1+\varepsilon)^{-i})$, it is straightforward to verify that $\|\psi_i - \mathcal{U}_i\|_{tvd} = O(\varepsilon)$. In the following, we prove that $p_0 = O(\varepsilon)$ which finalizes the proof.

We first bound the probability that A belongs to the set S_0.

Claim 3 $\Pr(\mathsf{A} \in S_0) \leq 2\varepsilon$.

Proof Let L be the set of elements a with $\Pr(\mathsf{A} = a) \leq \varepsilon/n$ and H be the set of elements a with $\Pr(\mathsf{A} = a) \geq 2^{2\gamma/\varepsilon}/n$; hence $\Pr(\mathsf{A} \in S_0) = \Pr(\mathsf{A} \in L) + \Pr(\mathsf{A} \in H)$. It is clear that $\Pr(\mathsf{A} \in L) \leq \varepsilon$. We further prove that $\Pr(\mathsf{A} \in H) \leq \varepsilon$ also.

Assume by contradiction that $\Pr(\mathsf{A} \in H) > \varepsilon$. We have $|H| \leq n/2^{2\gamma/\varepsilon}$ as otherwise the probability of being in H would be more than one. Let $\mathsf{B} \in \{0, 1\}$ be a random variable denoting whether or not $\mathsf{A} \in H$. We have,

$$\mathbb{H}(\mathsf{A} \mid \mathsf{B}) \geq_{\text{Fact 1-(b)}} \mathbb{H}(\mathsf{A}) - \mathbb{H}(\mathsf{B}) \geq_{\text{Fact 1-(a)}} \mathbb{H}(\mathsf{A}) - 1 \geq \log n - \gamma - 1 \tag{7}$$

On the other hand,

$$\begin{aligned}
\mathbb{H}(\mathsf{A} \mid \mathsf{B}) &= \Pr(\mathsf{A} \in H) \cdot \mathbb{H}(\mathsf{A} \mid \mathsf{B} = 1) + \Pr(\mathsf{A} \notin H) \cdot \mathbb{H}(\mathsf{A} \mid \mathsf{B} = 0) \\
&<_{\text{Fact 1-(a)}} \Pr(\mathsf{A} \in H) \cdot \log |H| + \Pr(\mathsf{A} \notin H) \cdot \log n \\
&\leq \varepsilon \cdot (\log n - 2\gamma/\varepsilon) + \varepsilon \cdot \log n \\
&= \log n - 2\gamma \leq \log n - \gamma - 1 && (\gamma \geq 1)
\end{aligned}$$

contradicting Eq (7). ■

Next, we bound the probability that A belongs to a small set.

Claim 4 $\Pr(\mathsf{A} \in \mathcal{S}) \leq 2\varepsilon$.

Proof Suppose by contradiction that $\Pr(\mathsf{A} \in \mathcal{S}) > 2\varepsilon$. Let $\mathsf{B} \in \{0,1\}$ be a random variable that denotes whether A chosen from $\mathcal{D}$ belongs to $\mathcal{L}$ or $\mathcal{S}$. We have,

$$\mathbb{H}(\mathsf{A} \mid \mathsf{B}) \geq_{\text{Fact 1-(b)}} \mathbb{H}(\mathsf{A}) - \mathbb{H}(\mathsf{B}) \geq_{\text{Fact 1-(a)}} \mathbb{H}(\mathsf{A}) - 1 \geq \log n - \gamma - 1 \tag{8}$$

The total number of elements belonging to small sets is at most,

$$|\mathcal{S}| \geq O(\gamma/\varepsilon^3) \cdot 2^{\left(\log n - \frac{\gamma}{\varepsilon}\right)} = 2^{\left(\log n - \frac{\gamma}{\varepsilon} + \log O(\gamma/\varepsilon^3)\right)} \tag{9}$$

Hence,

$$\begin{aligned}
\mathbb{H}(\mathsf{A} \mid \mathsf{B}) &= \Pr(\mathsf{A} \in \mathcal{S}) \cdot \mathbb{H}(\mathsf{A} \mid \mathsf{B} = 1) + \Pr(\mathsf{A} \notin \mathcal{S}) \cdot \mathbb{H}(\mathsf{A} \mid \mathsf{B} = 0) \\
&<_{\text{Fact 1-(a)}} 2\varepsilon \cdot \log|\mathcal{S}| + (1 - 2\varepsilon) \cdot \log n \\
&\leq_{\text{Eq (9)}} 2\varepsilon \cdot \left(\log n - \frac{\gamma}{\varepsilon} + \log O(\gamma/\varepsilon^3)\right) + (1 - 2\varepsilon) \cdot \log n \\
&= \log n - 2\gamma + \log O(\gamma/\varepsilon^3) < \log n - \gamma \quad (\text{as } \log O(\gamma/\varepsilon^3) < \gamma \text{ since } \varepsilon > \exp(-\gamma))
\end{aligned}$$

contradicting Eq (8). ■

To conclude,

$$p_0 = \Pr(\mathsf{A} \in S_0) + \Pr(\mathsf{A} \in \mathcal{S}) \leq 4\varepsilon$$

finalizing the proof. ■

Appendix B. Details of the Algorithm for Finding the Most Biased Coins

We present the proof of Theorem 1 in details in this section. Throughout this section, for any algorithm $\mathcal{A}$, $\mathsf{cost}(\mathcal{A})$ denotes the query complexity of $\mathcal{A}$ and $\mathsf{deg}(\mathcal{A})$ denotes the degree of adaptivity it uses, i.e., its round complexity. We start by providing a high level overview of the proof.

Overview: To illustrate the main ideas behind our algorithm, we focus on the case that $k = 1$. Consider the following type of input for best k coins problem: there exists a single *heavy* coin and $n-1$ *light* coins with the gap of Δ between the bias of the heavy coin and any light coin. It follows from a simple application of the Hoeffding's bound that for any $\delta \in (0,1)$, $O(\log(1/\delta)/\Delta^2)$ coin tosses are sufficient to distinguish whether a single coin is heavy or not with probability $1 - \delta$. We can now use this simple observation to design an r-round algorithm for each number of rounds r.

The case of $r = 1$ is quite simple: simply set $\delta = \Theta(\frac{1}{n})$ and a union bound ensures that with some constant probability, *every* coin is distinguished correctly, which allows us to output the heavy coin correctly. Now consider the case when $r = 2$. Here, the limited budget for 2-round algorithms

in Theorem 1 does not allow us to distinguish every coin correctly in the first round of coin tossing. Instead, we make the following simple yet crucial observation: it is enough for us to only classify the heavy coin and a large fraction of light coins correctly in the first round. Indeed by setting the parameter $\delta = \Theta(\frac{1}{\log n})$ (i.e., performing $O(n \log\log n/\Delta^2)$ coin tosses in the first round), we can reduce the set of possible choices for the heavy coin to roughly $n/\log n$ coins. But then our budget allows us to run the previous 1-round algorithm in the second round on this *smaller* set of coins to find the heavy coin. This results in the total number of coins tosses being $O(n \log\log n/\Delta^2)$ (in the first round) plus $O\left((n/\log n) \cdot \log(n/\log n)/\Delta^2\right) = O(n/\Delta^2)$ (in the second run), which matches the bounds for the $r = 2$ case in Theorem 1.

This discussion leads us to the following generic r-round algorithm: perform a number of coin tosses in the first round to recover a sufficiently smaller set that almost surely contains the heavy coin; recursively solve the problem on the remaining coins using the $(r-1)$-round version of the algorithm in the subsequent rounds. Here, "sufficiently smaller set" should be chosen such that the query complexity of an $(r-1)$-round algorithm on this set is within the budget of the r-round algorithm (over the original set of coins). Exploiting this approach to its fullest allows us to design our r-round algorithm for any number of rounds r and prove Theorem 1.

We now proceed to formalize the above discussion. Throughout this section, for simplicity of exposition, we will assume that the coins are indexed such that $p_i \geq p_{i+1}, \forall i \in [n-1]$, in which case the set of k most biased coins would be $[k]$. Note that the algorithm does not know this indexing. We provide our algorithm in Section B.1 and its analysis in Section B.2.

B.1. Algorithm

We design a recursive algorithm for proving Theorem 1. The pseudo-code is given in Algorithm 1. There are two main parameters in the input to Algorithm 1: a set $S \subseteq [n]$ of m *candidate* coins for the top k coins and a parameter r denoting the number of rounds of adaptivity the algorithm can use. In addition, the algorithm is given the confidence parameter $\delta \in (0, 1)$ and a lower bound on the gap parameter $\Delta \leq \Delta_k$. Given this input, Algorithm 1 works as follows:

1. Estimation phase: The algorithm first tosses each coin

$$t = O\left(\frac{1}{\Delta^2} \cdot \left(\text{ilog}^{(r)}(m) + \log(k/\delta)\right)\right)$$

many times and estimate the bias of each coin.

2. Elimination phase: Next, the algorithm identifies a set S' of $O(\frac{m}{\text{ilog}^{(r-1)}(m)})$ coins with the largest estimated bias and recursively solve the problem for the set S' in $r-1$ rounds.

We point out that the estimation phase of algorithm is allowed to be erroneous, i.e., there might be large deviations between the estimated biases and the true biases for a relatively large fraction of coins. The elimination phase is then designed to be robust to such error by selecting a suitably large subset for the next round. As rounds progress, the set of candidates for k most biased coins shrinks more and more such that in the last round, the algorithm can estimate the bias of each candidate with high confidence and return the k most biased coins. We should also point that in any round, if

the input set S is too small, i.e., of size $O(k)$, then Algorithm 1 bypasses the subsequent rounds and simply run the 1-round algorithm on this set to recover the top k coins.

B.2. Analysis

We establish the correctness of Algorithm 1 by proving Lemma 3 and then providing a bound on the number of coin tosses it makes in Lemma 9. We recall Lemma 3 (first appeared in Section 2.1.2).

Lemma 3 *Suppose S is any subset of coins $[n]$ with size m and gap parameter $\Delta \leq \Delta_k$ such that $[k] \subseteq S$. For any number of rounds $1 \leq r \leq \log^*(m) - 3$ and any confidence parameter $\delta \in (0,1)$, Algorithm 1 returns the set of k most biased coins w.p. at least $1-\delta$.*

Before proving Lemma 3, we need the following simple claim. In the remainder of this section, we fix $\varepsilon := \Delta/2$.

Claim 1 *For any round $r \geq 1$, and any coin $i \in S_r$,*

$$\Pr\left(|\widehat{p}_i - p_i| \geq \varepsilon\right) \leq \frac{\delta}{4k \cdot \mathrm{ilog}^{(r-1)}(m)}.$$

Proof By Hoeffding's inequality, we have,

$$\begin{aligned}\Pr\left(|\widehat{p}_i - p_i| \geq \varepsilon\right) &\leq 2\exp\left(-2\epsilon^2 \cdot t_r\right)\\ &\leq 2\exp\left(-\left(\mathrm{ilog}^{(r)}(m) + \log(8k/\delta)\right)\right) \leq \frac{\delta}{4k \cdot \mathrm{ilog}^{(r-1)}(m)}\end{aligned}$$

as $\mathrm{ilog}^{(r)}(m) = \log \mathrm{ilog}^{(r-1)}(m)$. ■

In the following, for any integer $r \geq 1$, we use $\mathcal{A}_r$ to denote Algorithm 1 with r number of rounds. We now prove Lemma 3.

Proof [of Lemma 3.]

The proof is by induction on the number of rounds r.

Base case: The base case follows immediately from Claim 1. Indeed for $r = 1$, Claim 1 ensures that for any $i \in S_1$,

$$\Pr\left(|\widehat{p}_i - p_i| \geq \varepsilon\right) \leq \frac{\delta}{4k \cdot \mathrm{ilog}^{(0)}(m_1)} \leq \frac{\delta}{m_1}$$

as $\mathrm{ilog}^{(r-1)}(m_1) = m_1$ by definition. By taking a union bound over all m_1 coins, we obtain that w.p. $1-\delta$, simultaneously for all coins $i \in S_1$, $|\widehat{p}_i - p_i| < \varepsilon$. This implies that w.p. $1-\delta$,

$$\begin{aligned}&\forall i \in [k] && \widehat{p}_i > p_i - \varepsilon = p_i - \Delta/2 \geq p_k - \Delta/2\\ &\forall j \in S_1 \setminus [k] && \widehat{p}_j < p_j + \varepsilon \leq p_j + \Delta/2 \leq p_{k+1} + \Delta/2\end{aligned}$$

As $\Delta \leq p_k - p_{k+1}$, we obtain that the returned set of k most biased coins according to $\widehat{p}$-values is the correct answer, finalizing the proof of the base case.

Induction step: Suppose the lemma is true for all number of rounds smaller than $r \leq \log^*(m) - 3$ and we prove it for the case of r rounds, i.e., for $\mathcal{A}_r$. In particular, we need to show that $\mathcal{A}_r$ returns the set of k most biased coins with probability at least $1 - \delta$.

Let $I = \{i \in [k] : \widehat{p}_i < p_i - \varepsilon\}$ and $J = \{j \in S_r \setminus [k] : \widehat{p}_j > p_j + \epsilon\}$. We know that for all $i \in [k]$ and $j \in S_r \setminus [k]$, $p_i - p_j \geq 2\varepsilon$. As the algorithm identifies a set of $m_{r-1} = k + \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}$ coins with the highest estimated biases (according to $\widehat{p}$) to recurse upon, we have,

$$\Pr(\mathcal{A}_r \text{ errs}) \leq \Pr(|I| > 0) + \Pr\left(|J| > \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}\right) + \Pr(\mathcal{A}_{r-1} \text{ errs} \mid \mathcal{E}) \tag{10}$$

where $\mathcal{E}$ denotes the event that $|I| = 0$ and $|J| \leq \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}$, i.e., the complement of the first two events above.

In the following, we bound probability of each event above. We first have,

$$\Pr(|I| > 0) \leq \sum_{i \in [k]} \Pr(\widehat{p}_i < p_i - \varepsilon) \leq_{\text{Claim 1}} k \cdot \frac{\delta}{4k \cdot \text{ilog}^{(r-1)}(m_r)} \leq \frac{\delta}{4} \tag{11}$$

where the last inequality is true because $\text{ilog}^{(r-1)}(m_r) \geq 1$.

We next bound the probability that $|J| > \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}$. For all $j \in S_r \setminus [k]$, we define an indicator random variable Y_j which is 1 iff $\widehat{p}_j > p_j + \varepsilon$. We further define $\mathsf{Y} := \sum_j \mathsf{Y}_j$. We have,

$$\mathbb{E}[\mathsf{Y}] = \sum_j \mathbb{E}[\mathsf{Y}_j] = \sum_j \Pr(\widehat{p}_j > p_j + \varepsilon) \leq_{\text{Claim 1}} \sum_j \frac{\delta}{4k \cdot \text{ilog}^{(r-1)}(m_r)} \leq \frac{\delta \cdot m_r}{4 \cdot \text{ilog}^{(r-1)}(m_r)}$$

Notice that $\mathsf{Y} = |J|$; hence,

$$\Pr\left(|J| > \frac{m_r}{\text{ilog}^{(r-1)}(m_r)}\right) \leq \Pr\left(\mathsf{Y} > \frac{4}{\delta} \cdot \mathbb{E}[\mathsf{Y}]\right) \leq \frac{\delta}{4} \tag{12}$$

where the last inequality is by Markov bound.

Finally, we calculate the probability of error of $\mathcal{A}_{r-1}$ conditioned on that none of the two events above happens (i.e., the event $\mathcal{E}$). In that case, we have $[k] \subseteq S_{r-1}$ and that $\Delta \leq \Delta_k$. As $r \leq \log^*(m_r) - 3$ (by the lemma statement), we have $r - 1 \leq (\log^*(m_r) - 1) - 3 \leq \log^*(\log m_r) - 3 \leq \log^*(m_{r-1}) - 3$. Therefore, the input to $\mathcal{A}_{r-1}$ satisfies the assumptions in the lemma statement as well and since the confidence parameter for $\mathcal{A}_{r-1}$ is $\delta/2$, we obtain that $\Pr(\mathcal{A}_{r-1} \text{ errs} \mid \mathcal{E}) \leq \delta/2$. By plugging in this bound, together with Eq (11) and Eq (12) to Eq (10), we obtain that $\mathcal{A}_r$ is also a δ-error algorithm, finalizing the proof of induction step. ■

Next, we prove an upper bound on the query complexity of $\mathcal{A}_r$ for any $r \geq 1$.

Lemma 9 *Suppose the input to Algorithm 1 satisfies the assumptions in Lemma 3; then Algorithm 1 makes at most* $\frac{10m}{\Delta^2} \cdot \left(\text{ilog}^{(r)}(m) + \log(8k/\delta)\right)$ *many coin tosses.*

Proof The proof is again by induction on the number of rounds r. The base case of $r = 1$ is trivially true. Now suppose the bounds are true for all integers smaller than $r \leq \log^*(m) - 3$ and we prove the lemma for the case of r rounds, i.e., for $\mathcal{A}_r$. Note that the total number of coin tosses in $\mathcal{A}_r$ is the sum of coins tosses in step 3 (which is $m \cdot t_r$) and the coins tosses in the recursive call which we bound bellow. For the recursive call there are two cases to consider depending on which of step 12 (Case 1) or step 14 (Case 2) in Algorithm 1 is being executed.

Case 1: In this case $\mathcal{A}_1$ is called with the confidence parameter $\delta/2$ on at most $2k$ coins. We do not use the induction hypothesis here and instead argue directly that,

$$\begin{aligned}
\mathsf{cost}(\mathcal{A}_r) &= m \cdot t_r + \mathsf{cost}(\mathcal{A}_1) \\
&\leq m \cdot t_r + \frac{4k}{\Delta^2} \cdot (\log(2k) + \log(16k/\delta)) \\
&\leq m \cdot t_r + \frac{8k}{\Delta^2} \cdot \log(8k/\delta) \\
&\leq m \cdot t_r + \frac{8m}{\Delta^2} \cdot \log(8k/\delta) && \text{(as } k \leq m\text{)} \\
&= \frac{2m}{\Delta^2} \cdot \left(\mathrm{ilog}^{(r)}(m) + \log(8k/\delta)\right) + \frac{8m}{\Delta^2} \cdot \log(8k/\delta) \\
& && \text{(by plugging in the value of } t_r\text{)} \\
&< \frac{10m}{\Delta^2} \cdot \left(\mathrm{ilog}^{(r)}(m) + \log(8k/\delta)\right)
\end{aligned}$$

which proves the induction step in this case.

Case 2: In this case, $\mathcal{A}_{r-1}$ is called with the confidence parameter $\delta/2$ on at most $\frac{2m}{\mathrm{ilog}^{(r-1)}(m)}$ coins. Hence, by induction, the total number of coin tosses made in recursive calls is

$$\begin{aligned}
\mathsf{cost}(\mathcal{A}_r) &= m \cdot t_r + \mathsf{cost}(\mathcal{A}_{r-1}) \\
&\leq m \cdot t_r + \frac{20m}{\Delta^2 \cdot \mathrm{ilog}^{(r-1)}(m)} \cdot \left(\mathrm{ilog}^{(r-1)}(2m) + \log(16k/\delta)\right) \\
&\leq m \cdot t_r + \frac{20m}{\Delta^2 \cdot \mathrm{ilog}^{(r-1)}(m)} \cdot \left(\mathrm{ilog}^{(r-1)}(m) + 1 + \log(8k/\delta) + 1\right) \\
&< m \cdot t_r + \frac{20m}{\Delta^2} + \frac{22m \cdot \log(8k/\delta)}{\Delta^2 \cdot \mathrm{ilog}^{(r-1)}(m)} \\
&< \frac{2m}{\Delta^2} \cdot \left(\mathrm{ilog}^{(r)}(m) + \log(8k/\delta)\right) + \frac{8m \cdot \mathrm{ilog}^{(r)}(m)}{\Delta^2} + \frac{8m \cdot \log(8k/\delta)}{\Delta^2}
\end{aligned}$$

where in the last inequality we used the bound on t_r plus the fact that $\mathrm{ilog}^{(r)}(m) \geq 16$ as $r \leq \log^*(m) - 3$. This concludes the proof of Lemma 9. ■

Theorem 1 now follows immediately from Lemma 3 and Lemma 9.

Appendix C. Details of the Lower Bound for the Most Biased Coins Problem

In this section, we provide the complete proof of Theorem 4. We first prove Theorem 4 for the case of $k = 1$, i.e., the case of finding the most biased coin and then show that a simple reduction extends this result to all possible values of k. Throughout this section, for any algorithm $\mathcal{A}$, $\mathsf{cost}(\mathcal{A})$ denotes the query complexity of $\mathcal{A}$ and $\mathsf{deg}(\mathcal{A})$ denotes the degree of adaptivity it uses, i.e., its round complexity. We start by providing a high level overview of our proof.

Overview. Consider the following input for the best coins problem introduced earlier in Section B: we have a collection of $n-1$ *light* coins and single *heavy* coin with the difference of Δ between the bias of the heavy coin and any light coin. A textbook result is that to classify a single coin as heavy or light correctly with sufficiently large constant probability $\Omega(1/\Delta^2)$ coins tosses are needed (see, e.g., Chernoff (1972)). Using this, it is possible to argue that $\Omega(n/\Delta^2)$ coin tosses are needed in these instances to recover the heavy coin. However notice that Theorem 4 is proving a stronger result on the query complexity of r-round algorithms for any $r \leq \log^* n - \log^* \Theta(\log^* n)$. To achieve this stronger bound, we take on a different approach as described below. For the purpose of the following discussion, it would be convenient to see Δ as some constant and hence suppress the bounds on Δ in asymptotic notation. We emphasize that this assumption is only for the purpose of following discussion and is not needed in our proof.

Our starting point is the following key claim that we prove: intuitively speaking, if an algorithm only tosses $n \cdot s$ coins in the first round, then it can only reduce the set of candidate coins (for being heavy) to $n/2^{\Theta(s)}$ possible coins. More formally, this means that conditioned on the outcome of the first $n \cdot s$ coin tosses, the heavy coin is still distributed (almost) uniformly at random over a set of $n/2^{\Theta(s)}$ possible coins.

Having this result, it is then easy to argue that one needs to set $s = \Omega(\log n)$ to recover the heavy coin in one round, resulting in an $\Omega(n \log n)$ lower bound on the query complexity of 1-round algorithms. There is also a more important takeaway from this discussion: any r-round algorithm that does not spend relatively large number of coin tosses in its first round is forced to find the heavy coin from a large pool of candidates (with essentially no further information) in the next $(r-1)$ rounds. Consequently, we can prove Theorem 4 inductively, by showing that if the number of coin tosses of an r-round algorithm is $o(n \cdot \text{ilog}^{(r-1)}(n))$, then in particular by setting $s = o(\text{ilog}^{(r)}(n))$ in the above argument, we end up with $\approx n/\sqrt{\text{ilog}^{(r-1)}(n)}$ possible choices for the heavy coin after the first round that needs to be further pruned in the subsequent $(r-1)$ rounds. But by induction, we need ,

$$\Omega(\left(n/\sqrt{\text{ilog}^{(r-1)}(n)}\right) \cdot \text{ilog}^{(r-1)}(n) = \Omega(n \cdot \sqrt{\text{ilog}^{(r-1)}(n)}) = \omega(n \cdot \text{ilog}^{(r)}(n))$$

many coin tosses to solve the problem in $(r-1)$ rounds (over $n/\sqrt{\text{ilog}^{(r-1)}(n)}$ coins), a contradiction with the bounds on the query complexity of the r-round algorithm.

To make the latter intuition precise we use a "round elimination" argument. We show that given any "good" r-round algorithm (i.e., a one with better query complexity than the bound in Theorem 4), there should exists a set of observed coins tosses outcome in the first round, such that conditioned on these coin tosses being the outcome of the first round two events simultaneously happens: (i) the algorithm still outputs a correct answer with essentially the same probability even after this conditioning, and (ii), the distribution of the heavy coin is close to uniform (in total variation distance) on a "large" subset of coins. Having this, we create a "good" $(r-1)$-round algorithm (defined as before) which "embed" its set of coin in the support of the distribution for heavy coin in above discussion and simulates the "missing input" (on the larger domain) for the r-round algorithm using independent randomness, which contradicts the induction step.

We now formalize the proof outlined above. Fix any arbitrary value $\Delta \in (0, 1/2)$ (possibly a function of n) and a constant $p < 1 - \Delta$. Our hard input distribution is defined based on the parameters Δ and p.

Distribution $\mathcal{D}_n^{\Delta,p}$. A hard input distribution on n coins for finding the most biased coin with the gap parameter $\Delta_1 = \Delta$.

- Sample an index $i^\star \in [n]$ uniformly at random.
- Let $p_{i^\star} = p + \Delta$ and $p_i = p$ for any $i \neq i^\star$ in $[n]$.
- Return the coins $[n]$ with biases $\{p_i\}_{i=1}^n$.

It is immediate to see that in any instance sampled from $\mathcal{D}_n^{\Delta,p}$, $\Delta_1 = \Delta$. Moreover, one can see that finding the most biased coin in this family of instances is equivalent to determining the value of $i^\star$. We use this fact to prove a lower bound on the query complexity of any algorithm for these instances.

Define the recursive function $e(r) = e(r-1) + o(1/r^2)$ with $e(1) = 0$. The following lemma is the main result of this section (which first appeared in Section 2.2).

Lemma 6 *Fix any integers $n, r \geq 1$. Suppose $\mathcal{A}_r$ is an r-round algorithm that given an instance sampled from $\mathcal{D}_n^{\Delta,p}$ outputs the most biased coin correctly w.p. at least $2/3 + e(r)$; then,*

$$\mathsf{cost}(\mathcal{A}_r) = \Omega(\frac{n}{\Delta^2 \cdot r^4} \cdot \mathrm{ilog}^{(r)}(n)).$$

Fix $n, r \geq 1$ and algorithm $\mathcal{A}_r$ as in Lemma 6. Note that by an averaging argument, we can assume w.l.o.g that $\mathcal{A}_r$ is deterministic. Indeed, for any randomized algorithm that errs w.p. at most δ on the distribution $\mathcal{D}_n^{\Delta,p}$, there exists a choice of random bits (used by the algorithm) that conditioned on, the error probability of algorithm is still at most δ where the probability is taken over the randomness of the distribution and observed outcomes. Hence, by conditioning on these random bits we obtain a deterministic algorithm with the same performance guarantee. Consequently, in the following, we assume that the algorithm $\mathcal{A}_r$ is deterministic.

To continue, we need some notation. Recall that S_1 denotes the multi-set of coins tossed in the first round by $\mathcal{A}_r$. Let s_1 denote the size of S_1 counting the multiplicities. We define the *outcome profile* of S_1 as the s_1-dimensional tuple $T = ((i_1, \theta_1), (i_2, \theta_2), \ldots, (i_{s_1}, \theta_{s_1}))$, whereby for any $j \in [s_1]$, i_j and θ_j, denote, respectively, the index of the j-th coin in S_1 and its value, i.e., heads or tails. We use I to denote the random variable for the index $i^\star$ in $\mathcal{D}_n^{\Delta,p}$, and T to denote the random variable for the vector T. We further use Θ_j, for any $j \in [s_1]$, to denote the random variable for parameter θ_j defined above. We let $\Theta := (\Theta_1, \ldots, \Theta_{s_1})$. Finally, for any k-dimensional tuple $X = (X_1, \ldots, X_k)$ and index $i \in [k]$, we define $X^{<i} = (X_1, \ldots, X_{i-1})$ and $X^{-i} := (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_k)$.

Notice that random variables $\Theta_1, \ldots, \Theta_{s_1}$ are in general correlated in the distribution $\mathcal{D}_n^{\Delta,p}$. However, we argue that for any $j \in [s_1]$, Θ_j and Θ^{-j} are *independent conditioned on* I. Indeed, conditioning on I fixes the distribution of the coins: Bernoulli $\mathcal{B}(p+\Delta)$ for the coin $i^\star$ and Bernoulli $\mathcal{B}(p)$ for the remaining coins. Therefore, since for all $j \in [s_1]$, Θ_j is sampled from the distribution of the coin i_j, we have,

$$\Theta_j \perp \Theta^{-j} \mid \mathsf{I} \tag{13}$$

The following lemma bounds the "information" revealed about the index $i^\star$ (i.e., the most biased coin) in *the first round* based on the number of coin tosses done by $\mathcal{A}_r$ in this round.

Lemma 7 $\mathbb{I}(\mathsf{I}\,;\mathsf{T}) = O(s_1 \cdot \Delta^2/n)$.

Proof Recall that $\mathcal{A}_r$ is a deterministic algorithm. This means that the multi-set S_1 of the coins being tossed in the first round by $\mathcal{A}_r$ is fixed apriori. As such, the random variable T is only a function of the vector $\Theta = (\Theta_1, \ldots, \Theta_{s_1})$. We have,

$$\begin{aligned}\mathbb{I}(\mathsf{I}\,;\mathsf{T}) = \mathbb{I}(\mathsf{I}\,;\Theta) &=_{\text{Fact 1-(d)}} \sum_{j=1}^{s_1} \mathbb{I}(\mathsf{I}\,;\Theta_j \mid \Theta^{<j}) = \sum_{j=1}^{s_1} \mathbb{H}(\Theta_j \mid \Theta^{<j}) - \mathbb{H}(\Theta_j \mid \Theta^{<j}, \mathsf{I}) \\ &\leq \sum_{j=1}^{s_1} \mathbb{H}(\Theta_j) - \mathbb{H}(\Theta_j \mid \mathsf{I}) = \sum_{j=1}^{s_1} \mathbb{I}(\mathsf{I}\,;\Theta_j) \end{aligned} \tag{14}$$

where the inequality is true since: (i) $\mathbb{H}(\Theta_j \mid \Theta^{<j}) \leq \mathbb{H}(\Theta_j)$ as conditioning can only drop the entropy (Fact 1-(c)), and (ii) $\mathbb{H}(\Theta_j \mid \Theta^{<j}, \mathsf{I}) = \mathbb{H}(\Theta_j \mid \mathsf{I})$ since by Eq (13), $\Theta_j \perp \Theta^{<j} \mid \mathsf{I}$ and hence further conditioning on $\Theta^{<j}$ does not change the entropy of Θ_j conditioned on I (again by Fact 1-(c)). We now bound each term $\mathbb{I}(\mathsf{I}\,;\Theta_j)$ in the above sum. In order to do this, we write,

$$\mathbb{I}(\mathsf{I}\,;\Theta_j) = \mathop{\mathbb{E}}_{i^\star \sim \mathcal{U}([n])} [\mathbb{D}(\text{dist}(\Theta_j) \,||\, \text{dist}(\Theta_j \mid \mathsf{I} = i^\star))] \tag{15}$$

as $i^\star$ is chosen uniformly at random from $[n]$ in $\mathcal{D}_n^{\Delta,p}$.

The following claim bounds each term above individually.

Claim 5 *For any $j \in [s_1]$,*

$$\mathbb{D}(\text{dist}(\Theta_j) \,||\, \text{dist}(\Theta_j \mid \mathsf{I} = i^\star)) = \begin{cases} O(\Delta^2) & \text{if } i_j = i^\star \\ O(\Delta^2/n^2) & \text{otherwise} \end{cases}$$

Proof Fix any $j \in [s_1]$. By definition of $\mathcal{D}_n^{\Delta,p}$, we have,

$$\text{dist}(\Theta_j) = \mathcal{B}(p + \Delta/n), \ \text{dist}(\Theta_j \mid \mathsf{I} = i_j) = \mathcal{B}(p + \Delta), \ \text{and dist}(\Theta_j \mid \mathsf{I} \neq i_j) = \mathcal{B}(p).$$

Suppose first $i_j = i^\star$. In this case,

$$\begin{aligned}\mathbb{D}(\text{dist}(\Theta_j) \,||\, \text{dist}(\Theta_j \mid \mathsf{I} = i^\star)) &= \mathbb{D}(\mathcal{B}(p + \Delta/n) \,||\, \mathcal{B}(p + \Delta)) \\ &\leq_{\text{Fact 4}} \frac{(p + \Delta/n - (p + \Delta))^2}{(p + \Delta) \cdot (1 - (p + \Delta))} = O(\Delta^2)\end{aligned}$$

since $p + \Delta$ and $(1 - (p + \Delta))$ are both constants bounded away from zero.

Similarly, if $i_j \neq i^\star$,

$$\begin{aligned}\mathbb{D}(\text{dist}(\Theta_j) \,||\, \text{dist}(\Theta_j \mid \mathsf{I} = i^\star)) &= \mathbb{D}(\mathcal{B}(p + \Delta/n) \,||\, \mathcal{B}(p)) \\ &\leq_{\text{Fact 4}} \frac{(p + \Delta/n - p)^2}{p \cdot (1 - p)} = O(\Delta^2/n^2)\end{aligned}$$

■

By plugging in the bounds established in Claim 5 to Eq (15), we have,

$$\begin{aligned}\mathbb{I}(\mathsf{I}\,;\Theta_j) &= \frac{1}{n}\cdot\mathbb{D}(\text{dist}(\Theta_j)\,||\,\text{dist}(\Theta_j\mid \mathsf{I}=i_j)) + \frac{n-1}{n}\cdot\mathbb{D}(\text{dist}(\Theta_j)\,||\,\text{dist}(\Theta_j\mid \mathsf{I}\neq i_j))\\ &=_{\text{Claim 5}} O(\Delta^2/n) + O(\Delta^2/n^2) = O(\Delta^2/n)\end{aligned}$$

Finally, we can plug this in Eq (14) and obtain that $\mathbb{I}(\mathsf{I}\,;\mathsf{T}) \leq \sum_{j=1}^{s_1} O(\Delta^2/n) = O(s_1\cdot\Delta^2/n)$, proving the lemma. ■

We are now ready to prove Lemma 6. The proof is by induction (on the number of rounds r). In the following claim, we prove the base case of this induction.

Claim 6 *With the assumption and notation of Lemma 6,* $\mathsf{cost}(\mathcal{A}_1) = \Omega(\frac{n}{\Delta^2}\cdot\log n)$.

Proof We use Fano's inequality (Fact 2) to prove this claim. Let $\delta = 1/3 - e(1) = 1/3$. Recall that algorithm $\mathcal{A}_1$ tosses the coins S_1 and given the value of these coin tosses in the outcome profile T determines the most biased coin w.p. at least $1-\delta = 2/3$. As argued earlier, determining the most biased coin in distribution $\mathcal{D}_n^{\Delta,p}$ is equivalent to determining the value of index $i^\star$. As such, T is an δ-error estimator for I. Hence,

$$\delta\cdot|\mathsf{I}| + H_2(\delta) \geq_{\text{Fact 2}} \mathbb{H}(\mathsf{I}\mid\mathsf{T}) = \mathbb{H}(\mathsf{I}) - \mathbb{I}(\mathsf{I}\,;\mathsf{T})$$

Now notice that $|\mathsf{I}| = \mathbb{H}(\mathsf{I}) = \log n$ by Fact 1-(a) as I is uniform over $[n]$. Moreover, by Lemma 7, $\mathbb{I}(\mathsf{I}\,;\mathsf{T}) = O(s_1\cdot\Delta^2/n)$. By reordering the terms above, we obtain that $s_1 = \Omega(\frac{n}{\Delta^2}\cdot\log n)$. Noting that $\mathsf{cost}(\mathcal{A}_1) = s_1$ finalizes the proof. ■

Now assume inductively that Lemma 6 is true for all integers smaller than r and we want to prove this for r-round algorithms. The proof of induction step is by contradiction. We show that if there exists an algorithm $\mathcal{A}_r$ with smaller query complexity than the bounds stated in Lemma 6 for $\mathcal{D}_n^{\Delta,p}$, then there also exists an $(r-1)$-round algorithm $\mathcal{A}_{r-1}$ with smaller query complexity on distribution $\mathcal{D}_m^{\Delta,p}$ for some appropriately chosen $m \leq n$, which contradicts the induction hypothesis.

Lemma 7 essentially implies that if the number of coin tosses in the first round is small, then the outcome profile T, on average, does not reveal much information about the identity of the most biased coin, i.e., I. More formally, if we assume (by way of contradiction) that $\mathsf{cost}(\mathcal{A}_r) = o(\frac{n}{\Delta^2\cdot r^4}\cdot\text{ilog}^{(r)}(n))$, we have,

$$\mathbb{H}(\mathsf{I}\mid\mathsf{T}) = \mathbb{H}(\mathsf{I}) - \mathbb{I}(\mathsf{I}\,;\mathsf{T}) =_{\text{Fact 1-(a)}} \log n - \mathbb{I}(\mathsf{I}\,;\mathsf{T}) =_{\text{Lemma 7}} \log n - o(\text{ilog}^{(r)}(n)/r^4) \qquad (16)$$

where in the last part we used the fact that $s_1 \leq \mathsf{cost}(\mathcal{A}_r)$. Now consider any fixed possible outcome profile T for $\mathcal{A}_r$ in round one, i.e., any possible value for T. We say that T is *uninformative* iff $\mathbb{H}(\mathsf{I}\mid\mathsf{T}=T) = \log n - o(\text{ilog}^{(r)}(n)/r^2)$. Roughly speaking, whenever the outcome profile in the first round is uninformative, the algorithm is quite "uncertain" about the identity of the most biased coin, and hence needs to find it among a large pool of candidate coins in the next $(r-1)$ rounds. This we argue is not possible as by induction hypothesis the available budget is not large enough to solve the problem in $(r-1)$ rounds on such a large domain (by induction hypothesis).

We start by showing that our assumption on query complexity of $\mathcal{A}_r$ implies that there exists an uninformative outcome profile in the first round which still results in a good output by $\mathcal{A}_r$ in the subsequent rounds.

Claim 7 *There exists an uninformative outcome profile T_{ui} of coin tosses in the first round of $\mathcal{A}_r$ such that,*

$$\Pr\left(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}\right) \leq \delta + o(1/r^2).$$

Proof Let $C := \log n - \mathbb{H}(\mathsf{I} \mid \mathsf{T})$. By Eq (16), $C = o(\text{ilog}^{(r)}(n)/r^4)$. For any $\varepsilon > 0$, we have,

$$\begin{aligned}\Pr_T\left(\log n - \mathbb{H}(\mathsf{I} \mid \mathsf{T} = T) \geq \frac{r^2}{\varepsilon} \cdot C\right) &\leq \frac{\varepsilon \cdot \mathbb{E}_T\left[\log n - \mathbb{H}(\mathsf{I} \mid \mathsf{T} = T)\right]}{r^2 \cdot C} \\ &= \frac{\varepsilon \cdot (\log n - \mathbb{H}(\mathsf{I} \mid \mathsf{T}))}{r^2 \cdot C} = \frac{\varepsilon}{r^2} \\ &\qquad \text{(by the choice of } C = \log n - \mathbb{H}(\mathsf{I} \mid \mathsf{T}))\end{aligned}$$

This means that w.p. at least $1 - \varepsilon/r^2$, $\mathbb{H}(\mathsf{I} \mid \mathsf{T} = T) \geq \log n - \frac{1}{\varepsilon} \cdot o(\text{ilog}^{(r)}(n)/r^2)$. By taking ε small enough, we have that the probability that T is uninformative is $1 - o(1/r^2)$.

Using this, we can write,

$$\Pr\left(\mathcal{A}_r \text{ errs} \mid T \text{ is uninformative}\right) \leq \Pr\left(\mathcal{A}_r \text{ errs}\right) + \Pr\left(T \text{ is not uninformative}\right) \leq \delta + o(1/r^2)$$

The assertion of the claim now follows by an averaging argument. ■

Let T_{ui} be the uninformative outcome profile in Claim 7 and define $\psi := \text{dist}(\mathsf{I} \mid \mathsf{T} = T_{\mathsf{ui}})$. As $\mathbb{H}(\mathsf{I} \mid \mathsf{T} = T_{\mathsf{ui}}) = \log n - o(\text{ilog}^{(r)}(n)/r^2)$, we can apply Lemma 8 on random variable $\mathsf{I} \mid \mathsf{T} = T_{\mathsf{ui}}$ with parameters $\gamma = o(\text{ilog}^{(r)}(n)/r^2)$ and $\varepsilon = o(1/r^2)$ to express its distribution ψ as a convex combination of distributions $\psi_0, \psi_1, \ldots, \psi_k$, i.e., $\psi = \sum_i q_i \cdot \psi_i$ (for $\sum_i q_i = 1$) such that $q_0 = o(1/r^2)$ and for all $i \geq 1$,

$$|\text{supp}(\psi_i)| \geq 2^{\left(\log n - o(\text{ilog}^{(r)}(n))\right)} \geq \frac{n}{\left(\text{ilog}^{(r-1)}(n)\right)^{o(1)}} \tag{17}$$

$$\|\psi_i - \mathcal{U}_i\|_{tvd} = o(1/r^2) \tag{18}$$

where $\mathcal{U}_i$ is the uniform distribution on $\text{supp}(\psi_i)$. With this notation,

$$\delta + o(1/r^2) \geq_{\text{Claim 7}} \Pr\left(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}\right) = \sum_i q_i \cdot \Pr\left(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}, \mathsf{I} \sim \psi_i\right)$$

As $q_0 = o(1/r^2)$, by an averaging argument, we have that there exists a distribution ψ_i for some $i \geq 1$ such that $\Pr\left(\mathcal{A}_r \text{ errs} \mid \mathsf{T} = T_{\mathsf{ui}}, \mathsf{I} \sim \psi_i\right) \leq \delta + o(1/r^2)$. Without loss of generality let this distribution be ψ_1 and define $m := |\text{supp}(\psi_1)|$. We now use the fact that ψ_1 is close to a uniform distribution on $\text{supp}(\psi_1)$ (in total variation distance) together with an embedding argument to show that,

Claim 2 *There exists a deterministic $(r-1)$-round $\delta + o(1/r^2)$-error algorithm for the best coins problem on $\mathcal{D}_m^{\Delta,p}$ with query complexity at most equal to $\mathsf{cost}(\mathcal{A}_r)$ on $\mathcal{D}_n^{\Delta,p}$.*

Proof Let $\mathcal{A}_{r,T_{\mathsf{ui}}}$ be an $(r-1)$-round algorithm obtained by running $\mathcal{A}_r$ from the second round onwards assuming that the outcome profile in the first round was T_{ui}. We use $\mathcal{A}_{r,T_{\mathsf{ui}}}$ to design a randomized algorithm $\mathcal{A}'$ for $\mathcal{D}_m^{\Delta,p}$.

Given any instance sampled from $\mathcal{D}_m^{\Delta,p}$, $\mathcal{A}'$ maps $[m]$ to $\text{supp}(\psi_1)$ (using any arbitrary bijection). Next, it runs $\mathcal{A}_{r,T_{\mathsf{ui}}}$ as follows: if $\mathcal{A}_{r,T_{\mathsf{ui}}}$ choose to toss a coin in $\text{supp}(\psi_1)$, $\mathcal{A}'$ also choose the corresponding coin in $[m]$; otherwise, if $\mathcal{A}_{r,T_{\mathsf{ui}}}$ choose to toss a coin in $[n] \setminus \text{supp}(\psi_1)$, $\mathcal{A}'$ simply toss a coin from the distribution $\mathcal{B}(p)$ and return the result to $\mathcal{A}_{r,T_{\mathsf{ui}}}$. Finally, if $\mathcal{A}_{r,T_{\mathsf{ui}}}$ outputs a coin from $\text{supp}(\psi_1)$, $\mathcal{A}'$ returns the corresponding coin in $[m]$ and otherwise $\mathcal{A}'$ simply return an arbitrary coin in $[m]$ as the answer.

It is trivially true that $\mathsf{cost}(\mathcal{A}') \leq \mathsf{cost}(\mathcal{A})$. Hence, in the following we prove the correctness of $\mathcal{A}'$. Let $\mathcal{D}'$ be the distribution of underlying instances on $[n]$ created by $\mathcal{A}'$. Let $\mathcal{U}_1$ be the uniform distribution on $\text{supp}(\psi_1)$. It is straightforward to verify that $\mathcal{D}' = \mathcal{D}_n^{\Delta,p} \mid \mathsf{I} \sim \mathcal{U}_1$, and that $\mathcal{D}'$ is a deterministic function of I. As such,

$$\begin{aligned}
\Pr_{\mathcal{D}_m^{\Delta,p}}\left(\mathcal{A}' \text{ errs}\right) = \Pr_{\mathcal{D}'}\left(\mathcal{A}_{r,T_{\mathsf{ui}}} \text{ errs}\right) &= \Pr_{\mathcal{D}_n^{\Delta,p}}\left(\mathcal{A}_{r,T_{\mathsf{ui}}} \text{ errs} \mid \mathsf{I} \sim \mathcal{U}_1\right) \\
&\leq_{\text{Fact 5}} \Pr_{\mathcal{D}_n^{\Delta,p}}\left(\mathcal{A}_{r,T_{\mathsf{ui}}} \text{ errs} \mid \mathsf{I} \sim \psi_1\right) + \|\psi_1 - \mathcal{U}_1\|_{tvd} \\
&=_{\text{Eq (18)}} \Pr_{\mathcal{D}_n^{\Delta,p}}\left(\mathcal{A}_r \text{ errs} \mid \mathsf{I} \sim \psi_1, \mathsf{T} = T_{\mathsf{ui}}\right) + o(1/r^2) \\
&\leq \delta + o(1/r^2)
\end{aligned}$$

To finalize the proof, note that by an averaging argument, there exists a fixing of the randomness in $\mathcal{A}'$ that results in the same error guarantee. But by fixing the randomness in $\mathcal{A}'$ we obtain a deterministic $(r-1)$-round algorithm $\mathcal{A}''$ that errs on $\mathcal{D}_m^{\Delta,p}$ w.p. at most $\delta + o(1/r^2)$. ■

We are now ready to conclude the proof of Lemma 6. By Claim 2, there exists an $(r-1)$-round algorithm $\mathcal{A}_{r-1}$ that errs w.p. at most $\delta + o(1/r^2) = 1/3 - e(r) + o(1/r^2) = 1/3 - e(r-1)$ on instances of $\mathcal{D}_m^{\Delta,p}$ such that $\mathsf{cost}(\mathcal{A}_{r-1}) \leq \mathsf{cost}(\mathcal{A}_r)$. But by induction hypothesis (as $\mathcal{A}_{r-1}$ is an $(r-1)$-round algorithm), we know,

$$\begin{aligned}
\mathsf{cost}(\mathcal{A}_{r-1}) &= \Omega\left(\frac{m}{\Delta^2 \cdot (r-1)^4} \cdot \text{ilog}^{(r-1)}(m)\right) \\
&=_{\text{Eq (17)}} \Omega\left(\frac{1}{\Delta^2 \cdot (r-1)^4} \cdot \frac{n}{\left(\text{ilog}^{(r-1)}(n)\right)^{o(1)}} \cdot \text{ilog}^{(r-1)}(n)\right) \\
&= \Omega\left(\frac{n}{\Delta^2 \cdot r^4} \cdot \left(\text{ilog}^{(r-1)}(n)\right)^{1-o(1)}\right) \\
&= \Omega\left(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(n)\right) \qquad \left(\text{as } \text{ilog}^{(r)}(n) = \log(\text{ilog}^{(r-1)}(n))\right)
\end{aligned}$$

which is in contradiction with $\mathsf{cost}(\mathcal{A}_{r-1}) \leq \mathsf{cost}(\mathcal{A}_r) = o\left(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(n)\right)$. This finalizes the proof of Lemma 6.

We can now conclude the proof of Theorem 4.

Proof [of Theorem 4] Fix parameters Δ and integers n, k. For simplicity, we assume k divides n. We further pick a constant $p < 1 - \Delta$. Create distribution $\mathcal{D}$ as follows:

1. Partition the set $[n]$ of coins into k equal size subsets $N_1, \ldots, N_k$ each of size $t := n/k$.

2. For any $j \in [k]$, we sample the bias of the coins in N_j from $\mathcal{D}_t^{\Delta,p}$.

Notice that in any instance sampled from distribution $\mathcal{D}$, there are k coins with bias $p+\Delta$ and $n-k$ coins with bias p, and hence $\Delta_k = \Delta$. Additionally, each of the coins with bias $p+\Delta$ belongs to a separate subset N_j. Hence, finding the top k most biased coins in distribution $\mathcal{D}$ amounts to finding the most biased coins in k *independent* instances sampled from $\mathcal{D}_t^{\Delta,p}$. We now use this to prove the lower bound.

Let $\mathcal{A}$ be a $(1/4)$-error r-round algorithm for finding the top k most biased coins in $\mathcal{D}$, and assume by contradiction that $\mathsf{cost}(\mathcal{A}) = o\left(\frac{n}{\Delta^2} \cdot \text{ilog}^{(r)}(\frac{n}{k})\right)$. This means that there exists at least one index $j \in [k]$, such that in expectation, only $o(\frac{n}{k \cdot \Delta^2} \cdot \text{ilog}^{(r)}(\frac{n}{k}))$ coins are being tossed in the coins in N_j. By Markov inequality, we have that w.p. $1 - o(1)$, only $o(\frac{n}{k \cdot \Delta^2} \cdot \text{ilog}^{(r)}(\frac{n}{k}))$ coins are being tossed in N_j; for brevity, let $\mathcal{E}$ denote this event. We have,

$$\Pr\left(\mathcal{A} \text{ finds the most biased coin in } N_j \mid \mathcal{E}\right) \geq 3/4 - o(1)$$

This means that $\mathcal{A}$ when restricted to the coins in N_j, finds the top most biased coin w.p. at least $3/4 - o(1)$ using at most $o(\frac{t}{\Delta^2} \cdot \text{ilog}^{(r)}(t))$ many coin tosses (recall that $t = n/k$). On the other hand, by Lemma 6, any algorithm for finding the most biased coins in instances sampled from $\mathcal{D}_t^{\Delta,p}$ w.p. at least $2/3 + e(r) = 2/3 + \sum_{i=1}^{r} o(1/i^2) = 2/3 + o(1) < 3/4 - o(1)$, requires $\Omega(\frac{t}{\Delta^2} \cdot \text{ilog}^{(r)}(t))$ many coin tosses, a contradiction. ■

Appendix D. Extension to Multi-Armed Bandits with Sub-Gaussian Rewards

In this section we discuss the problem of best arms identification in multi-armed bandits with sub-gaussian reward distributions defined as:

Definition 10 *(Sub-Gaussian Distributions) For any $b > 0$, we say a distribution $\mathcal{D}$ on $\mathbb{R}$ is b-sub-gaussian if for the random variable X drawn from $\mathcal{D}$ and any $t \in \mathbb{R}$, we have that*

$$\mathbb{E}\left[\exp(t \cdot \mathsf{X} - t\,\mathbb{E}[\mathsf{X}])\right] \leq \exp(b^2 \cdot t^2/2)\,.$$

The Bernoulli distribution is a special case of the 1-sub-Gaussian distribution. Any distribution with support in $[0, b]$ is a b-sub-Gaussian distribution. The b-sub-Gaussian family also contains many unbounded distributions such as the Gaussian distribution. We next give a version of Hoeffding's inequality for b-sub-Gaussian distributions.

Lemma 11 *(Hoeffding's inequality) Let $\mathsf{X}_1, \ldots, \mathsf{X}_\mathsf{m}$ be an i.i.d. sequence of random variables drawn from a b-sub-Gaussian distribution $\mathcal{D}$ with $\mu = \mathbb{E}_{\mathsf{X} \sim \mathcal{D}}[\mathsf{X}]$. Then for any $\epsilon > 0$, we have*

$$\Pr\left(\left|\frac{1}{m}\sum_{i=1}^{m} \mathsf{X}_i - \mu\right| \geq \epsilon\right) \leq 2\exp\left(-\frac{m\epsilon^2}{2b^2}\right)$$

We are given n arms, and the reward that we get on pulling each arm is a b-sub-Gaussian random variable with unknown mean. Let μ_i be the mean reward of arm $i \in [n]$. We define the problem of k best arms identification as:

Problem 1 (Best Arms) *Given arms $[n]$ with (unknown) mean rewards $\{\mu_i\}_{i=1}^n$, a parameter $k \in [n]$, the goal is to identify a set of k best arms in terms of mean rewards. We will assume that the set of k best arms is unique.*

For any $0 < \delta < 1$, a δ-error algorithm $\mathcal{A}$ for solving this problem is allowed to pull *the arms in $[n]$ and based on the outcomes of these pulls, return a set of arms which is the set of top-k arms w.p. at least $1 - \delta$.*

We now define the *gap parameter* for an instance of this problem in terms of the differences in mean rewards. For any $i \in [n]$, let,

$$\Delta_i = \begin{cases} \mu_{[i]} - \mu_{[k+1]} & \text{if } i \leq k \\ \mu_{[k]} - \mu_{[i]} & \text{otherwise} \end{cases} .$$

The gap parameter is then Δ_k, which is the difference between the mean rewards of k-th and $(k+1)$-th best arms.

We consider algorithms that in each round chooses a *multi-set* of arms to pull. The choice of this multi-set is *adaptive*, i.e. it is dependent on the history of rewards in previous rounds. Following the coin tossing problem, we denote by $\mathsf{deg}(\mathcal{A})$ the round complexity of algorithm $\mathcal{A}$, and by $\mathsf{cost}(\mathcal{A})$ the total number of arms pulled. We are interested in algorithms for solving this problem which have low round complexity. In particular, given a parameter r we are interested in δ-error algorithms $\mathcal{A}$ which have $\mathsf{deg}(\mathcal{A}) \leq r$.

D.1. Related Work

Multi-armed bandit problems have been widely studied in the machine learning and operations research community. These problems find applications in settings where there are a number of alternatives (arms) with unknown reward and the goal is to *explore* and then *exploit* the alternatives with the best rewards. This has mostly been studied under different objectives- 1) maximize the reward over the sequence of exploration (regret minimization), 2) find the best set of arms (best arm identification). Our focus is on the latter objective.

The best arms identification problem has also been widely studied over the past few years, starting from the work of Even-Dar et al. (2006). This paper gave a δ-error algorithm that finds the best arm in $O(\frac{n}{\Delta_k^2}\log(\frac{1}{\delta}))$ pulls. While their setting is a fully adaptive in which the algorithm pulls 1 arm in each round, we note that their algorithm can be *parallelized* with $\Theta(\log(n))$ rounds of adaptivity. Later, Kalyanakrishnan and Stone (2010) generalized this to best k arm identification which uses $O(\frac{n}{\Delta_k^2}\log(\frac{k}{\delta}))$ pulls and $\Theta(\log(n))$ rounds of adaptivity. Subsequently, Kalyanakrishnan et al. (2012) showed that the number of pulls achieved by the algorithm of Kalyanakrishnan and Stone (2010) is optimal in terms of worst-case query complexity. We give an algorithm for this problem that is optimal in terms of worst-case query complexity but also in terms of round complexity.

The latter work in this area has focussed on designing *instance-optimal* algorithms Kalyanakrishnan et al. (2012); Gabillon et al. (2012); Jamieson et al. (2013); Bubeck et al. (2013); Karnin et al. (2013); Chen and Li (2015); Kaufmann et al. (2016); Chen et al. (2017), i.e. algorithms whose sample complexity is closely tied to the bandit instance and is better than the worst case complexity for 'easy' bandit instances. While there has been significant progress on this front, these the algorithms typically have $\Omega(\log(n))$ rounds of adaptivity. Some of this work also studies a limited

budget setting, but for a reasonably good confidence δ the number of rounds of adaptivity again needs to be $\Omega(\log(n))$.

There are other multi-armed bandit settings which focus on limiting the adaptivity of an algorithm. One example is the delayed feedback setting in which the reward of pulling an arm in round t is shown in some later round τ_t. We note that our setting of limited adaptivity can be simulated in this setting, by taking an appropriately high value of τ_t. However, to the best of our knowledge most of the results in this setting focus on regret minimization and not on identifying the top arms.

Another example is the semi-bandit feedback setting in which in each round the algorithm is allowed to chose a subset of the arms to pull. However, in each round a particular arm can be pulled at most once. A trivial lower bound on the number of rounds required in this setting is $\Omega(\Delta_k^{-2})$ since one need these many pulls of the k-th arm to distinguish whether it is one of the top-k arm or not. This is not interesting when the number of arms is not large.

A recent work by Jun et al. (2016) is the most closely related to our work. The authors consider algorithms that pull a multi-set of arms in each round. There is bound on the number of pulls that the algorithm is allowed to pull in each round, and also the number of times an individual arm is pulled in each round. However, the number of rounds required by their algorithm in the worst case is at least $\Omega(\log(n))$ irrespective of the bound we set on the number of pulls.

Table 3 summarizes some of these results, and Table 4 gives related lower-bounds.

D.2. Our results on Best Arms Identification

We show that Algorithm 1 can be extended to solve the problem of best-arms identification in multi-armed bandits. This is easy to see if the reward distributions are Bernoulli. For sub-Gaussian reward distributions we prove the following theorem:

Theorem 12 *There exists an algorithm that given any number of rounds $r \geq 1$, integer $k \geq 1$, n arms with b-sub-Gaussian rewards with $b > 0$, and the gap parameter $\Delta_k \in (0,1)$, and confidence parameter $\delta \in (0,1)$, finds the set of the k best arms w.p. $1-\delta$ in r rounds with $O\left(\frac{b^2 n}{\Delta_k^2} \cdot \left(\text{ilog}^{(r)}(n) + \log\left(k/\delta\right)\right)\right)$ pulls.*

To prove the above theorem, the only change required in Algorithm 1 is that the number of pulls in each round also depends on the parameter b of the sub-Gaussian distribution. Specifically, we set

$$t_r := \frac{8b^2}{\Delta^2} \cdot \left(\text{ilog}^{(r)}(m) + \log\left(8k/\delta\right)\right) ,$$

in step 2 of Algorithm 1, while all the other steps remain the same. We first prove a claim on the estimation of rewards of sub-Gaussian rewards. This is similar to Claim 1 for the coin problem and we define ϵ in the same way as done in the proof of Theorem 1.

Claim 8 *For any round $r \geq 1$, and any arm $i \in S_r$, $\Pr\left(|\hat{\mu}_i - \mu_i| \geq \varepsilon\right) \leq \frac{\delta}{4k \cdot \text{ilog}^{(r-1)}(m)}$.*

Proof By Hoeffding's inequality for b sub-Gaussians Lemma 11, we have,

$$\begin{aligned}\Pr\left(|\hat{\mu}_i - \mu_i| \geq \varepsilon\right) &\leq 2\exp\left(-\frac{\epsilon^2 \cdot t_r}{2b^2}\right) \\ &\leq 2\exp\left(-\left(\text{ilog}^{(r)}(m) + \log(8k/\delta)\right)\right) \leq \frac{\delta}{4k \cdot \text{ilog}^{(r-1)}(m)}\end{aligned}$$

Table 3: Summary of some multi-armed bandit results for top-k arm identification

	Algorithm	# Rounds of Adaptivity	Sample Complexity
$k = 1$	Median Elimination Even-Dar et al. (2002)	$\Theta(\log(n))$	$O(\frac{n\log(1/\delta)}{\Delta_1^2})$
	Successive Rejects Audibert and Bubeck (2010)	$\Theta(n)$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log^2(\frac{n}{\delta})\right)$
	Exp-Gap Elimination Karnin et al. (2013)	$\Omega(\log(n))$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\frac{\log(\Delta_i^{-1})}{\delta})\right)$
	PRISM Jamieson et al. (2013)	$\Omega(\log(n))$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\Delta_i^{-1} \cdot \log(1/\delta))\right)$
	Distr-Based Elimination Chen and Li (2015)	$\Omega(\log(n))$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\frac{\log(\min\{n,\Delta_i^{-1}\})}{\delta})\right)$
Any $k \in [n]$	Halving Kalyanakrishnan and Stone (2010)	$\Theta(\log(n))$	$O(\frac{n\log(k/\delta)}{\Delta_k^2})$
	LUCB Kalyanakrishnan et al. (2012)	$\Omega\left(\Delta_k^{-2}\log(\frac{\sum_{i=1}^n \Delta_i^{-2}}{\delta})\right)$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\frac{\sum_{j=1}^n \Delta_j^{-2}}{\delta})\right)$
	UGapE Gabillon et al. (2012)	$\Omega\left(\Delta_k^{-2}\log(\frac{\sum_{i=1}^n \Delta_i^{-2}}{\delta})\right)$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\frac{\sum_{j=1}^n \Delta_j^{-2}}{\delta})\right)$
	Successive Accepts & Rejects Bubeck et al. (2013)	$\Theta(n)$	$O\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log^2(\frac{n}{\delta})\right)$
	This paper	$\log^*(n)$	$O(\frac{n\log(k/\delta)}{\Delta_k^2})$

as $\text{ilog}^{(r)}(m) = \log \text{ilog}^{(r-1)}(m)$. ■

The rest of the proof is exactly the same as the proof of Theorem 1. The lower bound follows from the fact that Bernoulli distributions are a special case of the 1-sub-Gaussian distributions.

Appendix E. Extension to Top-k Ranking Using Pairwise Comparison

We show here how to extend the lower bound in Theorem 4 (and Corollary 5) for best coins problem to the problem of finding top-k elements according to Borda scores. Consider any instance of the best coins problem with probabilities $\{p_i\}_{i=1}^n$ and gap parameter Δ_k. As shown in Section 3, not all instances of best coins problem can be mapped to an instance of the ranking problem. However,

Table 4: Lower bounds for top-k arm identification in multi-armed bandits

	Algorithm	Setting	Lower Bound
$k = 1$	Mannor and Tsitsiklis (2004)	Fully Adaptive	$\Omega(\frac{n\log(1/\delta)}{\Delta_1^2})$
	Mannor and Tsitsiklis (2004)	Fully Adaptive	$\Omega\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(1/\delta)\right)$ Instance-wise
	Chen and Li (2015)	Fully Adaptive	$\Omega\left(\sum_{i=1}^n \Delta_i^{-2} \cdot \log(\frac{\log(n)}{\delta})\right)$
Any $k \in [n]$	Kalyanakrishnan et al. (2012)	Fully Adaptive	$\Omega(\frac{n\log(k/\delta)}{\Delta_k^2})$
	This paper	$\Theta(\log^*(n))$ rounds of adaptivity	$\Omega(\frac{n\log(k/\delta)}{\Delta_k^2})$

in the following, we show that if for all coin i the bias $p_i \in [\frac{1}{2}, \frac{3}{4})$[5], then we can indeed map the instance of coin tossing problem on n coins to an instance of the ranking problem on $2n$ items such that each coin toss can be simulated via a single comparison and further on finding the top k coins is equivalent to finding the top k items according to Borda score. Note that by setting the parameter $p = 1/2$ in the distribution $\mathcal{D}^{\Delta,p}$ used in proving the lower bound for best coins problem in Section 2.2, we obtain a family of instances satisfying the assumption above. As such we can apply our lower bound in Theorem 4 (and Corollary 5) in a black-box fashion to obtain that,

Theorem 13 *For any parameter $\Delta \in (0, 1/4)$ and any integers $n, k \geq 1$, any r-round algorithm for finding the top-k items according to Borda score that succeed w.p. at least $3/4$ on instances with n items and gap parameter $\Delta_k = \Delta$, has query complexity $\Omega\left(\frac{n}{\Delta^2 \cdot r^4} \cdot \text{ilog}^{(r)}(\frac{n}{k})\right)$.*

In particular, $(\log^*(n) - \log^*(\Theta(\log^* n)))$ rounds of adaptivity is required by any algorithm that uses only $O(\frac{n}{\Delta^2})$ pairwise comparisons, i.e., has the optimal worst-case query complexity for constant k.

Proof Consider the following mapping between any instance $\{p_i\}_{i=1}^n$ of best k coins problem to an instance of the ranking problem on $2n$ items: we create two disjoint sets of items S and T each of size n. For each $i \in [n]$, suppose $p_i = 1/2 + \varepsilon_i$ for some $\varepsilon_i \in [0, 1/4)$. We define the preference matrix $\mathbf{P} \in \mathbb{R}^{2n \times 2n}$ as follows:

$$P_{i,j} = \begin{cases} 1/2 & \text{if } i \in S, j \in S \\ 1/2 & \text{if } i \in T, j \in T \\ 1/2 + 2(1 - \frac{1}{2n}) \cdot \varepsilon_i & \text{if } i \in S, j \in T \end{cases}$$

5. We note that the upper bound of $3/4$ in this range is not necessary and can be switched to any other constant smaller than 1 by increasing the size of the constructed ranking instance by a constant factor. However, the lower bound of $1/2$ is crucial for the purpose of the reduction.

Table 5: Summary of some top-k ranking results. In this table by passive we mean settings where there is an oracle that generates a (random) comparison graph $G = ([n], E)$ which is an undirected graph on n vertices. It then samples ℓ comparisons between items i and j if there is an edge $(i, j) \in E$, for some $\ell \geq 1$. It then reports all these comparisons to the algorithm. Note that the algorithm has no control on which items will be compared, and therefore, the comparison graph must satisfy certain properties (for example connectivity) to infer meaningful rankings from these comparisons. Note that this is different from non-adaptive settings, i.e. $r = 1$, where the algorithms is at least able to choose the samples in one round.

	Pairwise Comparison Model	**# Rounds of Adaptivity**	**Sample Complexity**
Chen and Suh (2015)	BTL	Non-adaptive	$O\left(\frac{n\log(n/\delta)}{(w_{[k]}-w_{[k+1]})^2}\right)$
Jang et al. (2016)	BTL	Non-adaptive	$O\left(\frac{n\log(n/\delta)}{(w_{[k]}-w_{[k+1]})^2}\right)$
Braverman et al. (2016)	NP	4	$O\left(\frac{n\log(n/\delta)}{(1-2p)^2}\right)$
Shah and Wainwright (2015)	General	Non-adaptive	$O(\frac{n\log(n/\delta)}{\Delta_k^2})$
Busa-Fekete et al. (2013)	General	$\Omega\left(\Delta_k^{-2}\cdot\log(n)\right)$	$O\left(\sum_{i=1}^n \Delta_i^{-2}\cdot\log(\frac{n}{\delta\Delta_i})\right)$
Heckel et al. (2016)	General	$\Omega\left(\Delta_k^{-2}\cdot\log(n)\right)$	$O\left(\sum_{i=1}^n \Delta_i^{-2}\cdot\log(\frac{n}{\delta\Delta_i})\right)$
This paper	General	$\Theta(\log^*(n))$	$O(\frac{n\log(k/\delta)}{\Delta_k^2})$

Under this transformation, the Borda score of each item $i \in S$ is

$$\tau_i = \frac{1}{2n-1}\cdot\sum_{j\neq i} P_{i,j} = \frac{1}{2n-1}\cdot\left((|S|-1)\cdot\frac{1}{2} + |T|\cdot(\frac{1}{2}+2\varepsilon_i-\frac{1}{n})\right)$$
$$= \frac{1}{2n-1}\cdot\left(\frac{n-1}{2}+\frac{n}{2}+(2n-1)\cdot\varepsilon_i\right) = \frac{1}{2}+\varepsilon_i$$

Moreover, it is straightforward to verify that for each item $j \in T$, $\tau_j < 1/2$. Consequently, the set of top k items in $[2n]$ coincides with the set of top k most biased coins in $[n]$.

We now argue that one can construct this instance implicitly given the set of n coins with unknown biases. Indeed we map the set of n coins to the set S of items and create n new items T. Then, any comparison between i and j can be simulated by (i) independently sampling from the

Bernoulli distribution $\mathcal{B}(\frac{1}{2})$ whenever i, j are both in S, or are both in T, or (ii) tossing the i-th coin in $[n]$ and declaring i beats j iff the coins tosses head whenever $i \in S$ and $j \in T$.

With this transformation, one can use any algorithm for the ranking problem to solve the instances of best coins problem that satisfy the aforementioned property with the same query and round complexity. Theorem 13 now follows immediately from Theorem 4 as the instances in the distribution $\mathcal{D}$ in Theorem 4 satisfy the needed property (by setting the parameter $p = 1/2$). ■