Finding a most biased coin with fewest flips

Karthekeyan Chandrasekaran * Richard Karp †

Abstract

We study the problem of learning a most biased coin among a set of coins by tossing the coins adaptively. The goal is to minimize the number of tosses to identify a coin i^* such that Pr (coin i^* is most biased) is at least $1-\delta$ for any given δ. Under a particular probabilistic model, we give an *optimal* algorithm, i.e., an algorithm that minimizes the expected number of tosses, to learn a most biased coin. The problem is equivalent to finding the best arm in the multi-armed bandit problem using adaptive strategies. Even-Dar et al. (2002) [14] and Mannor and Tsitsiklis (2004) [22] show upper and lower bounds matching up to constant factors on the number of coin tosses for several underlying settings of the bias probabilities. For a class of such settings we bridge the constant factor gap by giving an optimal adaptive strategy – a strategy that performs the best possible action under any given history of outcomes. For any given history, tossing the coin chosen by our strategy minimizes the expected number of tosses needed to learn a most biased coin. To our knowledge, this is the first algorithm that employs an optimal adaptive strategy under a Bayesian setting for this problem.

*karthe@gatech.edu, Georgia Institute of Technology
†karp@icsi.berkeley.edu, University of California, Berkeley

1 Introduction

The multi-armed bandit problem is a classical decision-theoretic problem with applications in bioinformatics, medical trials, stochastic algorithms, etc. [18]. The input to the problem is a set of arms, each associated with an unknown stochastic reward. At each step, an agent chooses an arm and receives a reward. The objective is to find a strategy for choosing the arms in order to achieve the best expected reward asymptotically. This problem has spawned a rich literature on the tradeoff between exploration and exploitation while choosing the arms [6, 21, 2, 3].

The motivation to identify the best bandit arm arises from problems where one would like to minimize regret within a fixed budget. In the models considered in [8, 1, 17], the goal is to choose an arm after a finite number of steps to minimize regret, where regret is defined to be the difference between the expected reward of the chosen arm and the reward of the optimal arm. The work of Bubeck et al. [8] suggested that the exploration-exploitation tradeoffs for this setting are much different from the setting where the number of steps is asymptotic. Following this, Audibert et al. [1] proposed exploration strategies to perform essentially as well as the best strategy that knows all distributions up to permutations of the coins. Gabillon et al. [17] addressed the problem of identifying the best arm for each bandit among a collection of bandits within a fixed budget. They proposed strategies that focus on arms whose expected rewards are closer to that of the optimal arm and show an upper bound on the probability of error for these strategies that decreases exponentially with the number of steps allowed.

In contrast, one could also attempt to optimize the budget subject to the quality of the arm to be identified. This is identical to racing and action elimination algorithms [23, 15, 14] which address the sample complexity of the pure exploration problem – given any $\delta > 0$, identify the arm with maximum expected reward with error probability at most δ while minimizing the total number of steps needed. This PAC-style learning formulation was introduced by Even Dar et al. [14]. Given a collection of n arms, Even Dar et al. [14] showed that a total of $O((n/\epsilon^2)\log(1/\delta))$ steps is sufficient to identify an arm whose expected reward is at most ϵ away from the optimal arm with correctness at least $1 - \delta$. Mannor and Tsitsiklis [22] showed lower bounds matching up to constant factors under various settings of the rewards. We attempt to bridge the constant factor gap by addressing the problem from a decision-theoretic perspective. Given the history of outcomes, does there exist a strategy to choose an arm so that the expected number of steps needed to learn the best arm is minimized?

Although the PAC-style learning problem appears to have garnered the interest of the learning theory community only over the past decade [14, 22, 11, 8, 1, 17], it has been actively studied in operations research as the "ranking and selection problem". It was introduced for normally distributed rewards by Bechhofer (1954) [4]. Adaptive strategies for this problem, known as "sequential selection", can be traced back to Paulson (1964) [24]. Variants of the problem find applications in minimizing the number of experimental simulations to achieve a given confidence level [24, 5, 20, 7, 25]. A simple and interesting case of the problem is when the most rewarding arm and the second-most rewarding arm differ in their mean rewards by at least $\epsilon > 0$. This special case is known as the "indifference-zone" assumption [4]. Strategies and their measure of optimality are known for various relaxations of independence, normality, equal and known variances and indifference-zone assumptions [20]. In the Bayesian setting, the mean rewards of the normal distributions are chosen from some underlying distribution [19, 10, 16, 9]. In this work, we address a particular Bayesian setting for Bernoulli rewards satisfying the indifference-zone assumption.

If the rewards from the bandit arms are Bernoulli, then learning the arm with the maximum

expected reward is equivalent to learning the most biased coin by tossing them adaptively. So, we will focus on this problem for the rest of the paper. Under the indifference zone assumption, a Chernoff bound leads to a trivial upper bound on the number of tosses in the non-adaptive setting – toss each coin $(4/\epsilon^2)\log(n/\delta)$ times and output the coin with the maximum number of heads outcomes. Let $\hat{p_i}$ denote the empirical probability of heads for the ith coin. By the Chernoff bound, $|\hat{p_i} - p_i| \leq \epsilon/2$ with probability at least $1 - \delta/n$. Therefore, by the union bound, it follows that this trivial toss-each-coin-k-times method outputs the most biased coin with probability at least $1 - \delta$.

In this work, we give a simple yet optimal strategy for choosing coins to toss in the pure exploration problem in a particular Bayesian setting. We show that our strategy is "optimal" – it performs the best possible action for any given history. Given the current history of outcomes of all coins, tossing the coin chosen by our strategy minimizes the expected number of tosses needed to learn a most biased coin. To the best of our knowledge, this is the first provably optimal strategy under a Bayesian setting of the problem with indifference zone assumption. Our strategy also improves on the number of coin tosses needed in comparison to the simple toss-each-coin-k-times method.

Setting. We are given a collection of n coins with $p_i := \mathsf{Pr}(\text{Heads for the } i\text{th coin})$. A coin i is said to be *heavy* if $p_i = p + \epsilon$ and *not-heavy* if $p_i = p - \epsilon$ for some given $\epsilon \in (0, 1/2)$ and $p \in [\epsilon, 1-\epsilon]$. Each coin in the collection is heavy with probability α and not-heavy with probability $1-\alpha$. Similar to most learning theory problems, given any $\delta > 0$, we would like to identify a heavy coin with correctness probability at least $1 - \delta$. The algorithm is allowed to toss coins adaptively. The goal is to minimize the expected number of tosses required. Since the collection of coins may not contain a heavy coin for any finite n with positive probability, we assume that $n = \infty$ so that $\mathsf{Pr}(\exists \text{ a heavy-coin}) = 1$.

An adaptive strategy is allowed to choose a coin to toss based on the current history of outcomes of the coin tosses. Given the history of outcomes of coin tosses, the cost of an adaptive strategy is equal to the expected number of further coin tosses needed to identify a most biased coin with correctness probability at least $1 - \delta$ by following the strategy. An adaptive strategy is said to be optimal if it has the minimum cost. Thus, an optimal adaptive strategy minimizes the expected number of tosses to identify a heavy coin with error at most δ.

1.1 Results

Our main result is an optimal adaptive algorithm for the above setting.

Theorem 1. *Given any $\delta > 0$, there exists an algorithm A that employs an optimal adaptive strategy to identify a heavy-coin with correctness at least $1 - \delta$ in the above setting. At any step, the time taken by A to identify the coin to toss is $O(\log n_0)$, where n_0 is the number of coins that have been tossed before the current step.*

We also quantify the improvement over the non-adaptive toss-each-coin-k-times method given by Chernoff bound. We give an upper bound on the expected number of tosses performed by our algorithm. We assume an infinite supply of coins under the same probabilistic setting. Then the toss-each-coin-k-times method is the following: repeatedly toss fresh coins $(4/\epsilon^2)\log(1/\delta)$ times; if the empirical probability of heads of the coin is at least $p - \epsilon/2$, then output that coin and terminate. This trivial algorithm will find a heavy coin after examining $1/\alpha$ coins in expectation.

Thus, the expected number of tosses performed by this trivial algorithm is

$$\left(\frac{1}{\alpha}\right)\left(\frac{4}{\epsilon^2}\right)\log\frac{1}{\delta}.$$

In contrast, our algorithm requires much smaller number of tosses.

Theorem 2. *Given any $\delta > 0$, there exists an algorithm A such that the expected number of tosses performed by A to identify a heavy-coin with correctness at least $1-\delta$ in the above setting, is at most*

$$\frac{16}{\epsilon^2}\left(\frac{1-\alpha}{\alpha} + \log\left(\frac{(1-\alpha)(1-\delta)}{\alpha\delta}\right)\right).$$

The implications of our upper bound when n is much larger than $1/\alpha$ but bounded needs to be contrasted with the lower bounds by Mannor and Tsitsiklis [22]. In this case, setting $n = c/\alpha$ seems to give a better upper bound on the expected number of tosses than the lower bound shown in Theorem 9 of [22]. We observe that Theorem 9 of [22] shows a lower bound in the most general Bayesian setting – there exists a prior distribution of the probabilities of the n coins so that any algorithm requires at least $O((n/\epsilon^2)\log(1/\delta))$ tosses in expectation. However, our algorithm works in a particular Bayesian setting by exploiting prior knowledge about this setting.

1.2 Algorithm

Algorithm Likelihood-Toss

1. `Initialize` $L_i \leftarrow 1$ `for every` $i \in [n]$.
2. `While` $(L_i < (1-\alpha)(1-\delta)/\alpha\delta \ \forall \ i \in [n])$
   (a) `Toss coin` i^* `such that` $i^* = \arg\max\{L_i : i \in [n]\}$. `(Break ties arbitrarily). Let`

$$b_{i^*} = \begin{cases} 1 & \text{if outcome is heads,} \\ 0 & \text{if outcome is tails.} \end{cases}$$

   (b) `Update` $L_{i^*} \leftarrow L_{i^*}\left(\frac{p+\epsilon}{p-\epsilon}\right)^{b_{i^*}}\left(\frac{1-p-\epsilon}{1-p+\epsilon}\right)^{1-b_{i^*}}$.
3. `Output` $\arg\max\{L_i : i \in [n]\}$.

Notation. Let $q = 1-p$, $\Delta_H = \log((p+\epsilon)/(p-\epsilon))$, $\Delta_T = \log((q+\epsilon)/(q-\epsilon))$, $B = \log((1-\alpha)(1-\delta)/\alpha\delta)$. At any stage of the algorithm, let the history of outcomes of a coin i be $D_i = (h_i, t_i)$ where h_i and t_i refer to the number of outcomes that were heads and tails respectively. Given the history D_i, the likelihood ratio of the coin is defined to be

$$L_i = \frac{\mathsf{Pr}(\text{Coin } i \text{ is heavy}|D_i)}{\mathsf{Pr}(\text{Coin } i \text{ is not-heavy}|D_i)} = \left(\frac{p+\epsilon}{p-\epsilon}\right)^{h_i}\left(\frac{q-\epsilon}{q+\epsilon}\right)^{t_i}.$$

2 Preliminaries

Our proof of optimality is based on an optimal strategy for multitoken Markov games. We now formally define the multitoken Markov game and state the optimal strategy that has been studied for this game. We use the notation and results from [12].

A *Markov system* $S = (V, P, C, s, t)$ consists of a state space V, a transition probability function $P : V \times V \to [0, 1]$, a positive real cost C_v associated with each state v, a start state s and a target state t. Let $v(0), v(1), \ldots, v(k)$ denote a set of states taken by following the Markov system for k steps. The cost of such a trip on S is the sum $\sum_{i=0}^{k-1} C_{v(i)}$ of the costs of the exited states.

Let $S_1, \ldots, S_n$ be n Markov systems, each of which has a token on its starting state. A *simple multitoken Markov game* $G = S_1 \circ S_2 \circ \cdots \circ S_n$ consists of a succession of steps in which we choose one of the n tokens, which takes a random step in its system (i.e., according to its P_i). After choosing a token i on state u say, we pay the cost $C_i(u)$ associated with the state u of the system S_i. We terminate as soon as one of the tokens reaches its target state for the first time. A strategy denotes the policy employed to pick a token given the state of the n Markov systems. The cost of such a game $\mathbb{E}[G]$ is the minimum expected cost taken over all possible strategies. The strategy that achieves the minimum expected cost is said to be *optimal*. A strategy is said to be *pure* if the choice of the token at any step is deterministic (entirely determined by the state of all Markov systems).

Theorem 3. *[12] Every Markov game has a pure optimal strategy.*

For any strategy π for a Markov game G, we denote the expected cost incurred by playing π on G by $\mathbb{E}_\pi[G]$.

The pure optimal strategy in the multitoken Markov game is completely determined by the *grade* γ of the states of the systems. The grade γ of a state is defined as follows: Given a Markov system $S = (V, P, C, s, t)$ and state u, let $S(u) = (V, P, C, u, t)$ denote the Markov system whose starting state is u. Consider the Markov game $S_g(u)$ – where at any step of the game one is allowed to either play in $S(u)$ or quit. Quitting incurs a cost of g. Playing in $S(u)$ is equivalent to taking a step following the Markov system S incurring the cost associated with the state of the system. The game stops once the target state is reached or once we quit. The grade $\gamma(u)$ of state u is defined to be the smallest real value g such that there exists an optimal strategy σ that plays in $S(u)$ in the first step. We note that, by definition, the cost of the game $S_{\gamma(u)}(u)$ is $\mathbb{E}[S_{\gamma(u)}(u)] = \gamma(u) = \mathbb{E}_\sigma[S_{\gamma(u)}(u)]$.

Theorem 4. *[12] Given the states $u_1, \ldots, u_n$ of the Markov systems in the multitoken Markov game the unique optimal strategy is to pick the token i such that $\gamma(u_i)$ is minimal.*

We use the following results from [13] to bound the number of tosses.

Theorem 5. *[13] Let $X \in [-\nu, \mu]$ be the random variable that determines the step-sizes of a one dimensional random walk with absorbing barriers at $-L$ and W. Let $L^* = L + \nu$, $W^* = W + \mu$ and $\phi(\rho) := \mathbb{E}\left(\rho^X\right)$.*

1. *The function $\phi(\rho)$ is convex. If $\mathbb{E}(X) \neq 0$, there exists a unique $\rho_0 \in (0, 1) \cup (1, \infty)$ such that $\phi(\rho_0) = 1$. If $\mathbb{E}(X) < 0$, then $\rho_0 > 1$ and if $\mathbb{E}(X) > 0$, then $\rho_0 < 1$.*

2. $$\mathsf{Pr}\,(\textit{Absorption at } W) \geq \frac{1 - \rho_0^L}{1 - \rho_0^{L+W^*}}.$$

3. If $\mathbb{E}(X) < 0$, *then*

$$\mathbb{E}(\textit{Number of steps to absorption}) \leq \frac{L^*}{|\mathbb{E}(X)|}.$$

4. If $\mathbb{E}(X) > 0$, *then*

$$\mathbb{E}(\textit{Number of steps to absorption}) \leq \frac{(L+W^*)}{\mathbb{E}(X)}\left(\frac{1-\rho_0^{L^*}}{1-\rho_0^{L^*+W}}\right).$$

3 Correctness

We first argue the correctness of the algorithm.

Lemma 6. *Given the history* D_i *for a coin* i,

$$\mathsf{Pr}(\textit{Coin } i \textit{ is heavy}|D_i) \geq 1-\delta \textit{ if and only if } L_i \geq \left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right).$$

Proof. The lemma is a straightforward application of Bayes' theorem.

$$\begin{aligned}
\mathsf{Pr}(\text{Coin } i \text{ is heavy}|D_i) &= \frac{\mathsf{Pr}(D_i|\text{Coin } i \text{ is heavy})\,\mathsf{Pr}(\text{Coin } i \text{ is heavy})}{\mathsf{Pr}(D_i)} \\
&= \frac{\alpha(p+\epsilon)^{h_i}(q-\epsilon)^{t_i}}{\alpha(p+\epsilon)^{h_i}(q-\epsilon)^{t_i} + (1-\alpha)(p-\epsilon)^{h_i}(q+\epsilon)^{t_i}} \\
&= \frac{\alpha L_i}{\alpha L_i + (1-\alpha)}.
\end{aligned}$$

Thus, it follows that

$$\mathsf{Pr}(\text{Coin } i \text{ is heavy}|D_i) \geq 1-\delta \text{ if and only if } L_i \geq \left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right).$$

□

The algorithm computes the likelihood ratio L_i for each coin i based on the history of outcomes of the coin. It immediately follows that the Algorithm Likelihood-Toss outputs a coin i^* such that

$$\mathsf{Pr}(\text{Coin } i^* \text{ is heavy}) \geq 1-\delta.$$

4 Optimality of the Algorithm

Consider the log-likelihood of a coin i defined as $X_i := \log L_i$. Given the history of a coin, the log-likelihood of the coin is determined uniquely. In the beginning, the history is empty and hence all log-likelihoods are identically zero. The influence of a toss on the log-likelihood is a random step for X_i – if the outcome of the toss is a head, then $X_i \leftarrow X_i + \Delta_H$ and if the outcome is a tail, then $X_i \leftarrow X_i - \Delta_T$. Thus, the toss outcomes of the coin leads to a 1-dimensional random-walk of the log-likelihood function associated with the coin. Further, since we stop tossing as soon as the log-likelihood of a coin is greater than $B = \log(1-\alpha)(1-\delta)/\alpha\delta$, the random-walk has an absorbing

barrier at B. We observe that the random walks performed by the n coins are independent since each coin being heavy is independent of the rest of the coins.

Thus, we have n identical Markov systems $S_1, \ldots, S_n$ each starting in state $X_i = 0$. Each Markov system also has a target state, namely the boundary B. A strategy to pick a coin to toss is equivalent to picking a Markov system $i \in [n]$. Each toss outcome is equivalent to the corresponding system taking a step following the transition probability and step size of the system. The goal to minimize the expected number of tosses is equivalent to minimizing the expected number of steps for one of the Markov systems to reach the target state.

Therefore, we are essentially seeking an optimal strategy to play a multitoken Markov game. We show that the strategy employed by Algorithm Likelihood-Toss is an optimal strategy to play the multitoken Markov game arising in our setting.

Let the Markov system associated with the one-dimensional random walk of the log-likelihood function of the history of the coin be $S = (V, P, C, s, t)$. Here, the state space V consists of every possible real value that is at most B. The target state is a special state determined by $t = B$. The starting state is $s = 0$. Given the current state X, the transition cost incurred is one while transition probabilities are defined as follows:

$$X \to \begin{cases} \min\{X + \Delta_H, B\} \text{ with probability } \mathsf{Pr}\,(\text{Heads}|X), \\ X - \Delta_T \text{ with probability } 1 - \mathsf{Pr}\,(\text{Heads}|X) \end{cases}$$

where

$$\begin{aligned} \mathsf{Pr}\,(\text{Heads}|X) &= \mathsf{Pr}\,(\text{Heads}|\text{Heavy coin})\,\mathsf{Pr}\,(\text{Heavy coin}|X) \\ &\quad + \mathsf{Pr}\,(\text{Heads}|\text{Non-heavy coin})\,\mathsf{Pr}\,(\text{Non-heavy coin}|X) \\ &= \frac{(p+\epsilon)\alpha e^X}{\alpha e^X + (1-\alpha)} + \frac{(p-\epsilon)(1-\alpha)}{\alpha e^X + (1-\alpha)}. \end{aligned}$$

We observe that the transition probabilities in this random-walk vary with the state of the system (as opposed to the well-known random-walk under uniform transition probability). In this modeling of the Markov System for the log-likelihood of each coin, we donot condition on the coin being heavy or not-heavy. We are postponing this decision by conditioning based on the history.

4.1 Proof of Optimality

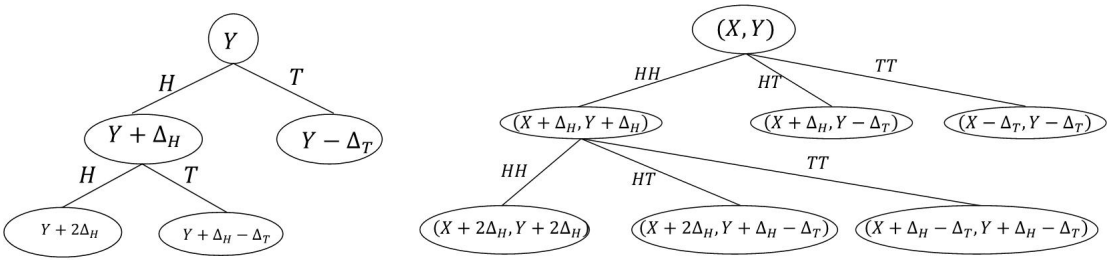
We now show that the grade is a monotonically non-increasing function of the log-likelihood.

Lemma 7. *Consider the Markov System $S = (V, P, C, s, t)$ associated with the log-likelihood function. Let $X, Y \in V$ such that $X \geq Y$. Then $\gamma(X) \leq \gamma(Y)$.*

Proof. Let $\gamma(Y) = g$. Then, by definition of grade, it follows that there exists a pure optimal strategy σ that chooses to toss the coin in the first step in $S_g(Y)$ and $E_\sigma[S_g(Y)] = g$. We will specify a mixed strategy π for $S_g(X)$ such that $\mathbb{E}_\pi[S_g(X)] \leq g$ and π chooses to play in the system $S(X)$ in the first step. It follows by definition that $\gamma(X) \leq g$.

The pure strategy σ can be expressed by a (possibly infinite) binary decision tree D_σ as follows: Each node u has an associated label $l(u) \in \mathbb{R}$. Each edge has a label from $\{H, T\}$. The root node v is labeled $l(v) = Y$. On reaching $l(u) < B$, if σ chooses to play in the system, then u has two children - the left and right children u_L, u_R are labeled $l(u_L) = l(u) + \Delta_H$ and $l(u_R) = l(u) - \Delta_T$

respectively. The edges $(u, u_L), (u, u_R)$ are labeled H and T respectively. On reaching $l(u) < B$, if σ decides to quit, then u is a leaf node. Finally, if $l(u) \geq B$, then u is a leaf node. We observe that since σ plays in the system $S_g(Y)$ in the first step, the root of D_σ is not a leaf.



Binary Decision Tree D_σ Ternary Decision Tree D_π

Figure 1: An example of a strategy σ represented as a binary decision tree D_σ for the Markov game $S_g(Y)$ where $B - 2\Delta_H < Y < B - \Delta_H$; the strategy σ is to continue playing in the system $S_g(Y)$ on reaching states Y and $Y + \Delta_H$ and to quit on reaching states $Y - \Delta_T$ and $Y + \Delta_H - \Delta_T$. The corresponding ternary decision tree D_π derived from D_σ is also shown.

We obtain a mixed strategy π for $S_g(X)$ by considering the following ternary tree D_π derived from D_σ: Each node u in D_π has an associated label $(l_X(u), l_Y(u)) \in \mathbb{R}^2$. Each edge in D_π has a label from $\{HH, HT, TT\}$. There is an onto mapping $m(u)$ from each node $u \in D_\pi$ to a node in D_σ. The root node u is labeled $(l_X(u) = X, l_Y(u) = Y)$ and $m(u) =$Root(D_σ). For any node u, if $m(u)$ is a leaf, then u is a leaf. Let u be a node such that $v = m(u)$ is not a leaf. Let v_H and v_T denote the left and right children of v. Create children u_{HH}, u_{HT}, u_{TT} as nodes adjacent to edges labeled HH, HT, TT respectively. Define the mapping $m(u_{HH}) = v_H$, $m(u_{HT}) = v_T$, $m(u_{TT}) = v_T$ and set

$$l_X(u_{HH}) = l_X(u) + \Delta_H,\ l_X(u_{HT}) = l_X(u) + \Delta_H,\ l_X(u_{TT}) = l_X(u) - \Delta_T,$$
$$l_Y(u_{HH}) = l_Y(u) + \Delta_H,\ l_Y(u_{HT}) = l_Y(u) - \Delta_T,\ l_Y(u_{TT}) = l_Y(u) - \Delta_T.$$

By construction of D_π, it follows that if $X \geq Y$, then at any node u in D_π, $l_X(u) \geq l_Y(u)$ and hence, $\mathsf{Pr}\,(\text{Heads}|l_X(u)) \geq \mathsf{Pr}\,(\text{Heads}|l_Y(u))$.

Our mixed strategy π for $S_g(X)$ is based on D_π. The strategy at any step maintains a pointer to some node u in D_π. Initialize the pointer to the root node u. If the pointer is at a non-leaf node u, then π chooses to play in the system. If the step in the system is a backward step (outcome of coin toss is a tail), then π moves the pointer to u_{TT}. If the step in the system is a forward step (outcome of coin toss is a head), then π generates a random number $r \in [0, 1]$ and moves the pointer to the node u_{HH} if $r < \mathsf{Pr}\,(\text{Heads}|l_Y(u))\,/\mathsf{Pr}\,(\text{Heads}|l_X(u))$ and to the node u_{HT} if $r \geq \mathsf{Pr}\,(\text{Heads}|l_Y(u))\,/\mathsf{Pr}\,(\text{Heads}|l_X(u))$. If the pointer is at a leaf node u such that $l_Y(u) < B$, then π quits the system. Otherwise, $l_Y(u) \geq B$ and hence $l_X(u) \geq B$. Thus, the strategy π is a valid mixed strategy for $S_g(X)$ and π plays in the system $S_g(X)$ in the first step.

It only remains to show that $\mathbb{E}_\pi[S_g(X)] \leq g$. This is shown in Claim 8. □

Claim 8.

$$\mathbb{E}_\pi[S_g(X)] \leq g.$$

Proof. The cost of using σ for $S_g(Y)$ can be simulated by running a random process in D_σ and considering an associated cost. For each non-leaf node in D_σ associate a cost of 1 and for each leaf node u in D_σ such that $l(u) < B$, associate a cost of g. Consider the following random process RP_1: Begin at the root u of D_σ. Repeatedly traverse the tree D_σ by taking the left child with probability $\mathsf{Pr}\,(\text{Heads}|l(u))$ and the right child with the remaining probability until a leaf node is reached. The cost of the random process is the sum of the cost incurred along the nodes in the path traversed by the random process. Let $\mathbb{E}[D_\sigma]$ denote the expected cost. Then, by construction of D_σ, it follows that $\mathbb{E}[D_\sigma] = \mathbb{E}_\sigma[S_g(Y)] = g$.

Next, we give a random process RP_2 on D_π that relates the expected cost of following strategy π on $S_g(X)$ and the expected cost of following strategy σ on $S_g(Y)$. We first associate a cost with each node u in D_π: For each non-leaf node u, if $l_X(u) < B$, then cost $c_X(u) = 1$, and if $l_Y(u) < B$, then cost $c_Y(u) = 1$. For each leaf node u, if $l_X(u) < B$, then cost $c_X(u) = g$ and if $l_Y(u) < B$, then cost $c_Y(u) = g$. The remaining costs are zero. Here, we observe that $c_X(u) \leq c_Y(u)$ for every node $u \in D_\pi$.

Now, the cost incurred by following strategy π for $S_g(X)$ is the same as the cost incurred by the following random process RP_2: Begin at the root node and repeatedly traverse the tree D_π by taking one of the three children at each non-leaf node until a leaf node is reached. On reaching a non-leaf node u, traverse to u_{HH} with probability $\mathsf{Pr}\,(\text{Heads}|l_Y(u))$, to u_{HT} with probability $\mathsf{Pr}\,(\text{Heads}|l_X(u)) - \mathsf{Pr}\,(\text{Heads}|l_Y(u))$ and to u_{TT} with the remaining probability.

Let P be the set of nodes in the path traversed by the random process RP_2. Let the cost incurred be $\bar{c}_X = \sum_{u \in P} c_X(u)$ and $\bar{c}_Y = \sum_{u \in P} c_Y(u)$. Due to the construction of D_π from D_σ, it follows that the expected cost incurred by RP_1 is equal to $\mathbb{E}[\bar{c}_Y]$. Hence, $\mathbb{E}[\bar{c}_Y] = \mathbb{E}[D_\sigma] = g$. Next, since $c_X(u) \leq c_Y(u)$ for every node u, it follows that $\mathbb{E}[\bar{c}_X] \leq \mathbb{E}[\bar{c}_Y] = g$. Finally, the expected cost incurred by following mixed strategy π for $S_g(X)$ is exactly equal to $\mathbb{E}[\bar{c}_X]$. □

Proof of Theorem 1. We use Algorithm Likelihood-Toss. By Lemma 6, the optimal adaptive strategy also minimizes the expected number of tosses to identify a coin i such that the log-likelihood $X_i \geq B$.

The strategy adopted by Algorithm Likelihood-Toss at any stage is to toss the coin with maximum log-likelihood. Let the Markov system associated with the one-dimensional random walk of the log-likelihood function of the history of the coin be $S = (V, P, C, s, t)$. We have n independent and identical Markov systems $S_1 = S_2 = \ldots = S_n = S$ associated with the log-likelihood function of the respective coin. By Theorem 4, the optimal strategy to minimize the expected number of tosses to identify a coin i such that the log-likelihood $X_i \geq B$ is to toss the coin i such that $\gamma(X_i)$ is minimal. Lemma 7 shows that the grade function $\gamma(X)$ is monotonically non-increasing. Thus, tossing the coin with maximum log-likelihood is an optimal strategy.

By the description of the algorithm, it is clear that updating the likelihood values after seeing the outcome of a coin toss takes constant time (since it is updated for only one coin). Thus, the optimal choice of coin to toss in the next step is determined by the algorithm in time linear in the number of coins. We observe that this can in fact be made logarithmic in the number of coins by maintaining a sorted ordering of the coins based on their likelihoods. □

5 Number of Coin Tosses

In this section, we give an upper bound on the number of coin tosses performed by Algorithm Likelihood-Toss. We assume an infinite supply of coins. Thus, the algorithm repeatedly tosses a coin while the log-likelihood of the coin is at least zero and starts with a fresh coin if the log-likelihood of the coin is less than zero. The algorithm terminates if the log-likelihood of a coin is at least B.

Consider the random walk of the log-likelihood function. The random walk has absorbing barriers at B and at every state less than 0.

Lemma 9. *Let C and D denote the expected number of tosses to get absorbed for a non-heavy and heavy coin respectively. Let π denote the probability that a heavy coin gets absorbed at B. Then,*

1. $$\pi \geq \frac{\Delta_H(p+\epsilon) - \Delta_T(q-\epsilon)}{2(\Delta_H + \Delta_T)}.$$

2. $$\frac{D}{\pi} \leq \left(\frac{8B}{\Delta_H(p+\epsilon) - \Delta_T(q-\epsilon)}\right)\left(\frac{\Delta_H + \Delta_T}{\Delta_H(p+\epsilon)}\right).$$

3. $$C \leq \frac{2(\Delta_H + \Delta_T)}{\Delta_T(q+\epsilon) - \Delta_H(p-\epsilon)}.$$

Proof. Consider a modified random walk where the starting state is Δ_H as opposed to zero. Let C' and D' denote the expected number of tosses for the modified walk to get absorbed using a non-heavy and heavy coin respectively. Let π' denote the probability that the modified walk gets absorbed at B using a heavy coin. Then, $D \leq D' + 1 \leq 2D'$, $C \leq C' + 1 \leq 2C'$, $\pi = (p+\epsilon)\pi'$.

We use Theorem 5. For the modified random walk, we have that $L = \Delta_H$, $W = B - \Delta_H$, $\nu = \Delta_T$, $\mu = \Delta_H$. For the modified random walk using a heavy coin, the step sizes are

$$X = \begin{cases} \Delta_H \text{ with probability } p+\epsilon \\ -\Delta_T \text{ with probability } q-\epsilon, \end{cases}$$

and for the modified random walk using a non-heavy coin, the step sizes are

$$Y = \begin{cases} \Delta_H \text{ with probability } p-\epsilon \\ -\Delta_T \text{ with probability } q+\epsilon, \end{cases}$$

For $\epsilon > 0$, we have that $\mathbb{E}(Y) < 0$. Therefore,

$$C' \leq \frac{\Delta_H + \Delta_T}{\Delta_T(q+\epsilon) - \Delta_H(p-\epsilon)}$$

and hence we have the bound on C.

Now consider the modified random walk using a heavy coin. For $\epsilon > 0$, we have that $\mathbb{E}(X) > 0$. Let $\rho_0 < 1$ be the unique real value such that $\mathbb{E}\left(\rho_0^X\right) = 1$. Thus,

$$\pi' \geq \frac{1-\rho_0^{\Delta_H}}{1-\rho_0^{B+\Delta_H}}$$
$$D' \leq \frac{(\Delta_H + B)}{\mathbb{E}(X)} \left(\frac{1-\rho_0^{\Delta_H+\Delta_T}}{1-\rho_0^{B+\Delta_T}}\right).$$

Since $\phi(\rho)$ is convex, it can be shown that the minimum value of $\phi(\rho)$ occurs at

$$\rho_{\min} = \left(\frac{\Delta_T(q-\epsilon)}{\Delta_H(p+\epsilon)}\right)^{\frac{1}{\Delta_H+\Delta_T}}$$

and hence, $\rho_0 < \rho_{\min} < 1$. Thus,

$$\begin{aligned}
\frac{D'}{\pi'} &\leq \frac{(\Delta_H + B)}{\mathbb{E}(X)} \left(\frac{1-\rho_0^{B+\Delta_H}}{1-\rho_0^{B+\Delta_T}}\right) \left(\frac{1-\rho_0^{\Delta_H+\Delta_T}}{1-\rho_0^{\Delta_H}}\right) \\
&\leq \frac{2B}{\mathbb{E}(X)} \left(\frac{1-\rho_0^{\Delta_H+\Delta_T}}{1-\rho_0^{\Delta_H}}\right) \qquad \text{(for } B \text{ larger than a constant)} \\
&< \frac{2B}{\mathbb{E}(X)} \left(\frac{1-\rho_{\min}^{\Delta_H+\Delta_T}}{1-\rho_{\min}^{\Delta_H}}\right) \qquad \text{(since } \rho_0 < \rho_{\min}\text{)} \\
&= \frac{2B}{\Delta_H(p+\epsilon)} \left(\frac{1}{1-\left(\frac{\Delta_T(q-\epsilon)}{\Delta_H(p+\epsilon)}\right)^{\frac{\Delta_H}{\Delta_H+\Delta_T}}}\right) \\
&\leq \frac{4B(\Delta_H+\Delta_T)}{\mathbb{E}(X)\,\Delta_H}.
\end{aligned}$$

and we obtain the bound on the ratio D/π. Finally, to lower bound π', we observe that

$$\begin{aligned}
\pi' &\geq \frac{1-\rho_0^{\Delta_H}}{1-\rho_0^{B+\Delta_H}} \\
&\geq \frac{1-\rho_{\min}^{\Delta_H}}{1-\rho_{\min}^{B+\Delta_H}} \\
&\geq 1-\rho_{\min}^{\Delta_H} \\
&\geq \frac{\mathbb{E}(X)}{2(\Delta_H+\Delta_T)(p+\epsilon)}.
\end{aligned}$$

□

Proof of Theorem 2. We use Algorithm Likelihood-Toss. Consider the one-dimensional random walk of the log-likelihood function. The random walk has absorbing barriers at B and at every state less than 0. Let C and D denote the expected number of tosses to get absorbed for a non-heavy and heavy coin respectively. Let π denote the probability that a heavy coin gets absorbed

at B. Let D_0 and D_1 denote the expected number of tosses of a heavy coin to get absorbed at 0 and B respectively. Then, $D = (1-\pi)D_0 + \pi D_1$.

Let E denote the expected number of tosses performed by algorithm Likelihood-Toss. Then,

$$\begin{aligned} E &\leq (1-\alpha)(C+E) + \alpha((1-\pi)(D_0+E) + \pi D_1) \\ \Rightarrow E &\leq \frac{(1-\alpha)}{\alpha}\frac{C}{\pi} + \frac{D}{\pi}. \end{aligned}$$

By Lemma 9, we have that

$$E \leq \left(\frac{4(\Delta_H+\Delta_T)}{\Delta_H(p+\epsilon) - \Delta_T(q-\epsilon)}\right)\left(\left(\frac{1-\alpha}{\alpha}\right)\left(\frac{\Delta_H+\Delta_T}{\Delta_T(q+\epsilon) - \Delta_H(p-\epsilon)}\right) + \left(\frac{2B}{\Delta_H(p+\epsilon)}\right)\right).$$

The final upper bound follows by substituting for Δ_H, Δ_T and B and using the following inequalities (derived by straightforward calculus),

$$\begin{aligned} \frac{2}{\epsilon} &\geq \max\left\{\frac{\Delta_H+\Delta_T}{\Delta_H(p+\epsilon) - \Delta_T(q-\epsilon)}, \frac{\Delta_H+\Delta_T}{\Delta_T(q+\epsilon) - \Delta_H(p-\epsilon)}\right\}, \\ \Delta_H &\geq \frac{\epsilon}{p-\epsilon}. \end{aligned}$$

□

Acknowledgment. We thank Santosh Vempala for valuable comments.

References

[1] J.-Y. Audibert, S. Bubeck, and R. Munos. Best Arm Identification in Multi-Armed Bandits. In *Proceedings of the Twenty-third Conference on Learning Theory*, COLT '10, pages 41–53, 2010.

[2] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47:235–256, May 2002.

[3] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The Nonstochastic Multiarmed Bandit Problem. *SIAM Journal of Computing*, 32:48–77, Jan 2003.

[4] R. E. Bechhofer. A Single-Sample Multiple Decision Procedure for Ranking Means of Normal Populations with known Variances. *The Annals of Mathematical Statistics*, 25:16–39, 1954.

[5] R. E. Bechhofer, T. J. Santner, and D. M. Goldsman. *Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons.* Wiley-Interscience, 1995.

[6] D. A. Berry and B. Fristedt. *Bandit Problems: Sequential Allocation of Experiments (Monographs on Statistics and Applied Probability).* Chapman & Hall, Oct 1985.

[7] J. Boesel, B. L. Nelson, and S.-H. Kim. Using Ranking and Selection to "Clean Up" after Simulation Optimization. *Operations Research*, 51(5):814–825, 2003.

[8] S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in multi-armed bandits problems. In *Proceedings of the 20th international conference on Algorithmic learning theory*, ALT'09, pages 23–37, 2009.

[9] S. E. Chick and N. Gans. Economic Analysis of Simulation Selection Problems. *Management Science*, 55(3):421–437, 2009.

[10] S. E. Chick and K. Inoue. New Two-Stage and Sequential Procedures for Selecting the Best Simulated System. *Operations Research*, 49(5):732–743, 2001.

[11] V. Cicirello and S. Smith. The Max k-Armed Bandit: A New Model for Exploration Applied to Search Heuristic Selection. In *20th National Conference on Artificial Intelligence*, AAAI '05, pages 1355–1361, 2005.

[12] I. Dumitriu, P. Tetali, and P. Winkler. On Playing Golf with Two Balls. *SIAM Journal of Discrete Mathematics*, 16:604–615, Apr 2003.

[13] S. N. Ethier and D. Khoshnevisan. Bounds on Gambler's Ruin Probabilities in Terms of Moments. *Methodology and Computing in Applied Probability*, 4(1):55–68, Mar 2002.

[14] E. Even-Dar, S. Mannor, and Y. Mansour. PAC Bounds for Multi-armed Bandit and Markov Decision Processes. In *Proceedings of the 15th Annual Conference on Computational Learning Theory*, COLT '02, pages 255–270, 2002.

[15] E. Even-Dar, S. Mannor, and Y. Mansour. Action Elimination and Stopping Conditions for the Multi-Armed Bandit and Reinforcement Learning Problems. *J. Mach. Learn. Res.*, 7:1079–1105, Dec. 2006.

[16] P. I. Frazier, W. B. Powell, and S. Dayanik. A Knowledge-Gradient Policy for Sequential Information Collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

[17] V. Gabillon, M. Ghavamzadeh, A. Lazaric, and S. Bubeck. Multi-Bandit Best Arm Identification. In *Advances in Neural Information Processing Systems*, NIPS '11, pages 2222–2230, 2011.

[18] J. Gittins, K. Glazebrook, and R. Weber. *Multi-armed Bandit Allocation Indices*. Wiley, 2nd edition, 2011.

[19] S. S. Gupta and K. J. Miescke. Bayesian look ahead one-stage sampling allocations for selection of the best population. *Journal of Statistical Planning and Inference*, 54(2):229–244, 1996.

[20] S. H. Kim and B. L. Nelson. Selecting the best system. *Handbooks in Operations Research and Management Science: Simulation*, pages 501–534, 2006.

[21] T. Lai and H. Robbins. Asymptotically Efficient Adaptive Allocation Rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

[22] S. Mannor and J. N. Tsitsiklis. The Sample Complexity of Exploration in the Multi-Armed Bandit Problem. *Journal of Machine Learning Research*, 5:623–648, Dec 2004.

[23] O. Maron and A. Moore. Hoeffding Races: Accelerating Model Selection Search for Classification and Function Approximation. In *Advances in Neural Information Processing Systems*, volume 6, pages 59–66, April 1994.

[24] E. Paulson. A Sequential Procedure for Selecting the Population with the Largest Mean from k Normal Populations. *The Annals of Mathematical Statistics*, 35:174–180, 1964.

[25] J. Pichitlamken and B. L. Nelson. Selection-of-the-best procedures for optimization via simulation. In *Proceeding of the 2001 Winter Simulation Conference*, pages 401–407, 2001.