An Instance-Based Algorithm for Deciding the Bias of a Coin

Luís Fernando Schultz Xavier da Silveira[*] Michiel Smid[†]

March 29, 2022

Abstract

Let $q \in (0,1)$ and $\delta \in (0,1)$ be real numbers, and let C be a coin that comes up heads with an unknown probability p, such that $p \neq q$. We present an algorithm that, on input C, q, and δ, decides, with probability at least $1-\delta$, whether $p < q$ or $p > q$. The expected number of coin flips made by this algorithm is $O\left(\frac{\log\log(1/\varepsilon)+\log(1/\delta)}{\varepsilon^2}\right)$, where $\varepsilon = |p-q|$.

1 Introduction

Let $q \in (0,1)$ and $\varepsilon \in (0, \min\{q, 1-q\})$ be real numbers. Consider a coin that comes up heads with an unknown probability p and, thus, comes up tails with probability $1-p$. Assume we know that $p \in \{q+\varepsilon, q-\varepsilon\}$.

Let $\delta \in (0,1)$ be a real number. The following algorithm decides, with probability at least $1-\delta$, whether $p = q - \varepsilon$ (this corresponds to the output *YES*) or $p = q + \varepsilon$ (this corresponds to the output *NO*):

- Flip the coin k times, where
$$k = \left\lceil \frac{\ln(1/\delta)}{2\varepsilon^2} \right\rceil .$$
- Let X be the number of heads in this sequence of k coin flips.
  - If $X \leq qk$, then return *YES*.
  - If $X > qk$, then return *NO*.

[*] School of Electrical Engineering and Computer Science, University of Ottawa, Canada. Supported by NSERC.

[†] School of Computer Science, Carleton University, Ottawa, Canada. Supported by NSERC.

To prove correctness, assume first that $p = q - \varepsilon$. By the Chernoff–Hoeffding bound (see Lemma 1 below), we have

$$\begin{aligned}
\Pr(\text{ the algorithm returns } NO\,) &= \Pr\left(X > qk\right) \\
&\leq \Pr\left(X \geq qk\right) \\
&= \Pr\left(X \geq pk + \varepsilon k\right) \\
&\leq e^{-2k\varepsilon^2} \\
&\leq \delta
\end{aligned}$$

and, therefore, with probability at least $1 - \delta$, the algorithm correctly returns *YES*. By a symmetric argument, in case $p = q + \varepsilon$, the algorithm correctly returns *NO* with probability at least $1 - \delta$.

Observe that this algorithm must know the values of q, ε, and δ. The number of coin flips made by the algorithm is $O(\frac{\log(1/\delta)}{\varepsilon^2})$, which is optimal for the case when $q = 1/2$: Any algorithm that determines, with probability at least $1-\delta$, whether $p = 1/2-\varepsilon$ or $p = 1/2+\varepsilon$, must flip the coin $\Omega(\frac{\log(1/\delta)}{\varepsilon^2})$ times in the worst case. For a proof of this claim, see Lemma 5.1 in Anthony and Bartlett [1]. The results by Mannor and Tsitsiklis [4] imply the same lower bound for the expected number of coin flips made by any algorithm that uses, besides flipping the coin, randomization to decide when to terminate.

In this paper, we consider a more general version of this problem. Besides the coin having an unknown probability $p \in (0, 1)$ of coming up heads, we are given a real number $q \in (0, 1)$ such that $p \neq q$ and a real number $\delta \in (0, 1)$. How can we decide whether p is smaller than or larger than q?

More formally, we consider the problem of designing an algorithm that takes as input the above coin and the real numbers $q \in (0, 1)$ and $\delta \in (0, 1)$, and outputs *YES* or *NO*, such that

1. if $p < q$, then the output is *YES* with probability at least $1 - \delta$,

2. if $p > q$, then the output is *NO* with probability at least $1 - \delta$.

Any such algorithm will repeatedly flip the coin and determine its output based on the resulting sequence of heads and tails. The goal is to minimize the number of coin flips made by the algorithm. Intuitively, this number should depend on the absolute value $|p - q|$ of the difference between p and q: The smaller this value is, the more coin flips are needed to decide which of p and q is larger.

An obvious approach is the following. For a given value $\varepsilon \in (0, 1)$, the algorithm flips the coin k times, where

$$k = \left\lceil \frac{\ln(1/\delta)}{2\varepsilon^2} \right\rceil . \tag{1}$$

Let X be the number of heads in this sequence of coin flips. The Chernoff–Hoeffding bound implies the following two claims: If $p < q$, then

$$\Pr\left(X \geq qk + \varepsilon k\right) \leq \Pr\left(X \geq pk + \varepsilon k\right) \leq e^{-2k\varepsilon^2} \leq \delta.$$

If $p > q$, then

$$\Pr\left(X \leq qk - \varepsilon k\right) \leq \Pr\left(X \leq pk - \varepsilon k\right) \leq e^{-2k\varepsilon^2} \leq \delta.$$

Based on the value of X, the algorithm does the following:

- If $X \leq qk - \varepsilon k$, it returns *YES*.
- if $X \geq qk + \varepsilon k$, it returns *NO*.
- Otherwise, the algorithm does not have enough information to decide which of p and q is larger. In this case, the algorithm chooses a smaller value of ε, recomputes the value of k according to (1), and repeats.

Observe that this approach is similar to searching for a value, say x, in an infinite sorted array $A[1\ldots]$. If x is stored at $A[n]$, where n is unknown at the start, the algorithm first finds an index, say m, such that $x \leq A[m]$ and m is polynomial in n. Given this index m, the algorithm then performs a binary search for x in the bounded sorted array $A[1\ldots m]$. For a detailed exposition of this technique, see Bentley and Yao [2].

A natural choice for the values of ε is the sequence $1/2^i$ for $i = 1, 2, 3, \ldots$. Since each iteration of this algorithm depends on the outcomes of all previous iterations, it is not clear that this algorithm is correct with probability at least $1 - \delta$.

In this paper, we show that we do obtain a correct algorithm, if we take a slightly larger value for k: In (1), we replace $\ln(1/\delta)$ by $\ln(\pi^2 i^2/(6\delta))$.

Let

$$d = \left\lceil \log\left(\frac{1}{|p-q|}\right)\right\rceil,$$

where log is the logarithm to the base 2. If, for example, $q = 1/2$ and $p > q$, then d is the position of the leftmost bit in which the (infinite) binary representations of p and q differ. We can think of d as being the degree of "difficulty": The larger d is, the closer p and q are to each other and, thus, the more "difficult" it is to decide whether $p < q$ or $p > q$.

Using the new value for k, we prove the following:

1. The output of the algorithm is correct with probability at least $1 - \delta$.
2. The expected number of iterations made by the algorithm is at most

$$d + 1.2 = \log\left(\frac{1}{|p-q|}\right) + O(1).$$

3. The expected total number of coin flips made by the algorithm and its expected running time are

$$O\left(4^d \cdot \log(d/\delta)\right) = O\left(\left(\log\log\left(\frac{1}{|p-q|}\right) + \log(1/\delta)\right) \cdot \frac{1}{(p-q)^2}\right).$$

2 The Algorithm

Below, we give a formal description of the algorithm. In Section 3, we will analyze the success probability, the expected number of iterations, and the expected total number of coin flips.

Algorithm CoinFlipper(C, q, δ)

Comment: C is a coin with an unknown probability of coming up heads, and $q \in (0,1)$ and $\delta \in (0,1)$ are real numbers. All coin flips are mutually independent.

```
$i = 1$;
while $true$
do $\varepsilon = 1/2^i$;
    $k = \lceil \ln\left(\frac{\pi^2 i^2}{6\delta}\right) / \left(2\varepsilon^2\right) \rceil$;
    flip the coin $k$ times;
    $X$ = number of heads;
    if $X \leq qk - \varepsilon k$
    then return $YES$ and terminate
    else if $X \geq qk + \varepsilon k$
        then return $NO$ and terminate
        else $i = i + 1$
        endif
    endif
endwhile
```

3 The Analysis of Algorithm CoinFlipper

Our analysis will use the additive version of the well known Chernoff–Hoeffding bound (see, e.g., inequalities (1.6) of Theorem 1.1 in Dubhashi and Panconesi [3]):

Lemma 1 (Chernoff–Hoeffding) *Let $k \geq 1$ be an integer and let $p \in (0,1)$ be a real number. Consider a coin that comes up heads with probability p. Let X be the random variable that counts the number of heads in a sequence of k mutually independent coin flips. Then, for any real number $\varepsilon \in (0,1)$,*

$$\Pr\left(X \geq pk + \varepsilon k\right) \leq e^{-2k\varepsilon^2}$$

and

$$\Pr\left(X \leq pk - \varepsilon k\right) \leq e^{-2k\varepsilon^2}.$$

Throughout the rest of this section, p denotes the (unknown) probability that the coin C comes up heads, and q and δ are the real numbers that are the input to algorithm

$\textsc{CoinFlipper}(C, q, \delta)$. We will assume throughout that $p > q$. The analysis for the case when $p < q$ is symmetric.

3.1 The Success Probability

We have to prove that, with probability at least $1 - \delta$, algorithm $\textsc{CoinFlipper}(C, q, \delta)$ returns *NO*. Thus, if we let A be the event

$$A = \text{“algorithm } \textsc{CoinFlipper}(C, q, \delta) \text{ returns } \textit{YES}\text{”},$$

then we have to prove that $\Pr(A) \leq \delta$.

For each integer $i \geq 1$, define the events

$$\begin{aligned} A_i &= \text{“algorithm } \textsc{CoinFlipper}(C, q, \delta) \text{ returns } \textit{YES} \text{ in iteration } i\text{”}, \\ B_i &= \text{“iteration } i \text{ of algorithm } \textsc{CoinFlipper}(C, q, \delta) \text{ takes place”}. \end{aligned}$$

Observe that the events A_i and $A_i \cap B_i$ are the same. Using this, together with the fact that the event A is the pairwise disjoint union of the A_i's, we have

$$\begin{aligned} \Pr(A) &= \Pr\left(\bigcup_{i=1}^{\infty} A_i\right) \\ &= \sum_{i=1}^{\infty} \Pr\left(A_i\right) \\ &= \sum_{i=1}^{\infty} \Pr\left(A_i \cap B_i\right) \\ &= \sum_{i=1}^{\infty} \Pr\left(A_i \mid B_i\right) \cdot \Pr\left(B_i\right) \\ &\leq \sum_{i=1}^{\infty} \Pr\left(A_i \mid B_i\right). \end{aligned}$$

Let $i \geq 1$ be an integer. We will derive an upper bound on $\Pr\left(A_i \mid B_i\right)$. Consider iteration i of algorithm $\textsc{CoinFlipper}(C, q, \delta)$, and the values of ε, k, and X during this iteration. It follows from the algorithm that

$$\begin{aligned} \Pr\left(A_i \mid B_i\right) &= \Pr\left(X \leq qk - \varepsilon k\right) \\ &\leq \Pr\left(X \leq pk - \varepsilon k\right), \end{aligned}$$

where the inequality follows from the assumption that $p > q$. Lemma 1 implies that

$$\Pr\left(A_i \mid B_i\right) \leq e^{-2k\varepsilon^2}.$$

Since

$$2k\varepsilon^2 \geq \ln\left(\frac{\pi^2 i^2}{6\delta}\right),$$

it follows that

$$\Pr\left(A_i \mid B_i\right) \leq \frac{6\delta}{\pi^2} \cdot \frac{1}{i^2}.$$

Using the well known identity $\sum_{i=1}^{\infty} 1/i^2 = \pi^2/6$, we conclude that

$$\begin{aligned} \Pr(A) &\leq \frac{6\delta}{\pi^2} \sum_{i=1}^{\infty} \frac{1}{i^2} \\ &= \frac{6\delta}{\pi^2} \cdot \frac{\pi^2}{6} \\ &= \delta. \end{aligned}$$

3.2 The Expected Number of Iterations

Let Y be the random variable that counts the number of iterations made when running algorithm $\textsc{CoinFlipper}(C, q, \delta)$. For each integer $i \geq 1$, define the indicator random variable

$$Y_i = \begin{cases} 1 & \text{if iteration } i \text{ takes place,} \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$Y = \sum_{i=1}^{\infty} Y_i.$$

Let

$$d = \left\lceil \log\left(\frac{1}{|p-q|}\right) \right\rceil.$$

Observe that $d \geq 1$ and, because of our assumption that $p > q$,

$$q + 1/2^d \leq p < q + 1/2^{d-1}.$$

Using the Linearity of Expectation, we have

$$\begin{aligned} \mathbb{E}(Y) &= \mathbb{E}\left(\sum_{i=1}^{\infty} Y_i\right) \\ &= \sum_{i=1}^{d+1} \mathbb{E}\left(Y_i\right) + \sum_{j=1}^{\infty} \mathbb{E}\left(Y_{d+j+1}\right) \\ &\leq d + 1 + \sum_{j=1}^{\infty} \mathbb{E}\left(Y_{d+j+1}\right). \end{aligned}$$

Let $j \geq 1$ be an integer. Consider iteration $d+j$ of algorithm $\textsc{CoinFlipper}(C, q, \delta)$, and the values of ε, k, and X during this iteration. We have

$$\begin{aligned}
\mathbb{E}\left(Y_{d+j+1}\right) &= \Pr\left(B_{d+j+1}\right) \\
&= \Pr\left(B_{d+j+1} \cap B_{d+j}\right) \\
&= \Pr\left(B_{d+j+1} \mid B_{d+j}\right) \cdot \Pr\left(B_{d+j}\right) \\
&\leq \Pr\left(B_{d+j+1} \mid B_{d+j}\right) \\
&= \Pr\left(qk - \varepsilon k < X < qk + \varepsilon k\right) \\
&\leq \Pr\left(X < qk + \varepsilon k\right).
\end{aligned}$$

Since $q \leq p - 1/2^d$, we have

$$\begin{aligned}
\mathbb{E}\left(Y_{d+j+1}\right) &\leq \Pr\left(X \leq \left(p - 1/2^d\right) k + \varepsilon k\right) \\
&= \Pr\left(X \leq pk - \left(1/2^d - \varepsilon\right) k\right).
\end{aligned}$$

Observe that

$$1/2^d - \varepsilon = 1/2^d - 1/2^{d+j} \geq 1/2^d - 1/2^{d+1} = 1/2^{d+1},$$

implying that

$$\mathbb{E}\left(Y_{d+j+1}\right) \leq \Pr\left(X \leq pk - k/2^{d+1}\right).$$

Using Lemma 1, we obtain

$$\mathbb{E}\left(Y_{d+j+1}\right) \leq e^{-k/2^{2d+1}}.$$

It follows from the algorithm that

$$\begin{aligned}
\frac{k}{2^{2d+1}} &\geq \frac{\ln\left(\frac{\pi^2 (d+j)^2}{6\delta}\right)}{2\varepsilon^2} \cdot \frac{1}{2^{2d+1}} \\
&\geq \frac{\ln\left(\frac{4\pi^2}{6\delta}\right)}{2\varepsilon^2} \cdot \frac{1}{2^{2d+1}} \\
&\geq \frac{\ln(6/\delta)}{2\varepsilon^2} \cdot \frac{1}{2^{2d+1}} \\
&= 4^{j-1} \cdot \ln(6/\delta) \\
&\geq 4^{j-1} \cdot \ln 6.
\end{aligned}$$

Therefore,

$$\mathbb{E}\left(Y_{d+j+1}\right) \leq (1/6)^{4^{j-1}}. \tag{2}$$

Thus,

$$\mathbb{E}(Y) \leq d + 1 + \sum_{j=1}^{\infty} (1/6)^{4^{j-1}}.$$

The infinite series converges and its value is approximately 0.167438, which is less than 0.2. We conclude that

$$\begin{aligned}
\mathbb{E}(Y) &\leq d + 1.2 \\
&= \left\lceil \log\left(\frac{1}{|p-q|}\right) \right\rceil + 1.2.
\end{aligned}$$

3.3 The Expected Total Number of Coin Flips

Let Z be the random variable that counts the total number of coin flips made when running algorithm $\textsc{CoinFlipper}(C, q, \delta)$. Using the indicator random variables Y_i of Section 3.2, and denoting the value of k in iteration i by k_i, we have

$$\begin{aligned}
\mathbb{E}(Z) &= \mathbb{E}\left(\sum_{i=1}^{\infty} Y_i \cdot k_i\right) \\
&= \sum_{i=1}^{d+1} \mathbb{E}\left(Y_i\right) \cdot k_i + \sum_{j=1}^{\infty} \mathbb{E}\left(Y_{d+j+1}\right) \cdot k_{d+j+1} \\
&\leq \sum_{i=1}^{d+1} k_i + \sum_{j=1}^{\infty} \mathbb{E}\left(Y_{d+j+1}\right) \cdot k_{d+j+1}. \qquad (3)
\end{aligned}$$

Since, for $1 \leq i \leq d+1$,

$$\begin{aligned}
k_i &\leq 1 + \frac{1}{2} \cdot 4^i \cdot \ln\left(\frac{\pi^2 i^2}{6\delta}\right) \\
&\leq 1 + \frac{1}{2} \cdot 4^i \cdot \ln\left(\frac{\pi^2 (d+1)^2}{6\delta}\right),
\end{aligned}$$

we obtain the following upper bound on the first summation in (3):

$$\begin{aligned}
\sum_{i=1}^{d+1} k_i &\leq d + 1 + \frac{1}{2} \cdot \ln\left(\frac{\pi^2 (d+1)^2}{6\delta}\right) \sum_{i=1}^{d+1} 4^i \\
&= O\left(4^d \cdot \log(d/\delta)\right). \qquad (4)
\end{aligned}$$

To bound the second summation in (3), let $j \geq 1$. Using (2), we have

$$\mathbb{E}\left(Y_{d+j+1}\right) \cdot k_{d+j+1} \leq \left(\frac{1}{6}\right)^{4^{j-1}} \left(1 + \frac{1}{2} \cdot 4^{d+j+1} \cdot \ln\left(\frac{\pi^2 (d+j+1)^2}{6\delta}\right)\right).$$

Since $d + j + 1 \leq 3dj$, it follows that

$$\begin{aligned}
\ln\left(\frac{\pi^2 (d+j+1)^2}{6\delta}\right) &\leq \ln\left(\frac{9\pi^2 (dj)^2}{6\delta}\right) \\
&\leq \ln\left(\frac{3\pi^2}{2}\right) + \ln\left(\frac{(dj)^2}{\delta^2}\right) \\
&\leq 3 + 2 \cdot \ln\left(\frac{dj}{\delta}\right).
\end{aligned}$$

Thus, we obtain the following upper bound on the second summation in (3):

$$\begin{aligned}
\sum_{j=1}^{\infty} \mathbb{E}\left(Y_{d+j+1}\right) \cdot k_{d+j+1} &\leq \sum_{j=1}^{\infty} \left(\frac{1}{6}\right)^{4^{j-1}} \left(1 + \frac{1}{2} \cdot 4^{d+j+1} \cdot \left(3 + 2 \cdot \ln\left(\frac{dj}{\delta}\right)\right)\right) \\
&= \sum_{j=1}^{\infty} \left(\frac{1}{6}\right)^{4^{j-1}} + && (5) \\
&\quad 4^{d+1} \left(\frac{3}{2} + \ln\left(\frac{d}{\delta}\right)\right) \sum_{j=1}^{\infty} 4^j \cdot \left(\frac{1}{6}\right)^{4^{j-1}} + && (6) \\
&\quad 4^{d+1} \sum_{j=2}^{\infty} 4^j \cdot \left(\frac{1}{6}\right)^{4^{j-1}} \ln j. && (7)
\end{aligned}$$

The infinite series in (5), (6), and (7) converge, and their values are approximately 0.167438, 0.679012, and 0.00855737, respectively. Thus,

$$\sum_{j=1}^{\infty} \mathbb{E}\left(Y_{d+j+1}\right) \cdot k_{d+j+1} = O\left(4^d \cdot \log(d/\delta)\right). \tag{8}$$

By combining (3), (4) and (8), we obtain our upper bound on the expected total number of coin flips made by algorithm CoinFlipper(C, q, δ):

$$\begin{aligned}
\mathbb{E}(Z) &= O\left(4^d \cdot \log(d/\delta)\right) \\
&= O\left(\left(\log\log\left(\frac{1}{|p-q|}\right) + \log(1/\delta)\right) \cdot \frac{1}{(p-q)^2}\right).
\end{aligned}$$

Since the running time of algorithm CoinFlipper(C, q, δ) is proportional to Z, we obtain the same upper bound on its expected running time.

The following theorem summarizes our result.

Theorem 1 *Let $q \in (0,1)$ and $\delta \in (0,1)$ be real numbers, and let C be a coin that comes up heads with an unknown probability p, such that $p \neq q$. Algorithm* CoinFlipper(C, q, δ) *has the following properties:*

1. *If $p < q$, then the output is YES with probability at least $1 - \delta$.*

2. *If $p > q$, then the output is NO with probability at least $1 - \delta$.*

3. *Let $\varepsilon = |p - q|$. The expected total number of coin flips is*

$$O\left(\frac{\log\log(1/\varepsilon) + \log(1/\delta)}{\varepsilon^2}\right).$$

As we mentioned in Section 1, if the algorithm gets as input the values of q, δ, and ε (i.e., the algorithm knows that $p \in \{q - \varepsilon, p + \varepsilon\}$), the upper bound on the number of coin flips can be improved to $O(\frac{\log(1/\delta)}{\varepsilon^2})$. As shown by Anthony and Bartlett [1] and Mannor and Tsitsiklis [4], this is the best possible upper bound.

Open Problem *Does there exist an algorithm that gets as input only the values of q and δ, together with the promise that $p \neq q$, and solves the problem by making, in expectation, $O\left(\frac{\log(1/\delta)}{\varepsilon^2}\right)$ coin flips?*

References

[1] M. Anthony and P. L. Bartlett. *Neural Network Learning: Theoretical Foundations.* Cambridge University Press, Cambridge, UK, 1999.

[2] J. L. Bentley and A. C.-C. Yao. An almost optimal algorithm for unbounded searching. *Information Processing Letters*, 5:82–87, 1976.

[3] D. P. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, Cambridge, UK, 2009.

[4] 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, 2004.