Exploration with Limited Memory: Streaming Algorithms for Coin Tossing, Noisy Comparisons, and Multi-Armed Bandits

Sepehr Assadi, Chen Wang
Rutgers University

Abstract

- Finding the most biased coin by tossing – a classical exploration problem in computer science and machine learning.
- Assuming a gap parameter \( \Delta \), elimination-based algorithms have provided solution with \( O\left(\frac{n}{\Delta^2}\right) \) coin tosses which matches the lower bound.
- However, these algorithms inherently require storing all the coins, which is not memory-efficient.
- We studied the sample-space trade-off under the streaming coin tossing model: the algorithm can only toss an incoming or stored coin.
- We designed an algorithm which only stores a single extra coin, which means the sample-space trade-off does not exist.
- In route to the one-coin algorithm, we also proposed preliminary memory-efficient algorithms with \( O\left(\log(n)\right) \), \( O\left(\log(n)\log(n)\right) \) and \( O\left(\log(n) \right) \) stored coins.
- Extensions of our main algorithm includes finding the k most biased coins and other exploration problems. E.g., Finding top-k elements using noisy comparisons; Finding an \( \varepsilon \)-best arm in stochastic multi-armed bandits.

Preliminary Algorithms

The \( O\left(\log(n)\right) \)-Coin memory Algorithm:
- Multiple levels: 4-coin memory per level
- Level 1: toss each coin \( \frac{n}{2^4} \) times; send the most biased to the level 2.
- Level 2+: increase the number of tosses by 1.5x
  - Correctness: Probability of losing coin* exponentially decreases.
  - Sample complexity: \( i \)-th level: \( \frac{n}{2^i} \) \((1.5)^{i-1} \frac{n}{2^i}\), overall \( O\left(\frac{n}{2^i}\right) \)
  - Space Complexity: \( O\left(\log(n)\right) \) levels; each level 4 coins.

The \( O\left(\log(n)\log(n)\right) \) and \( O\left(\log(n)\right) \) Coin Algorithms:
- \( O\left(\log(\log(n))\right) \) memory: stopping at the \( \log\log(n) \) level
- \( O\left(\log(n)\right) \) memory: aggressive selections of coins (iterative logarithm factor) and increments of coin tosses (tower factor) (cf. [Agarwal et al., 2017])

Main Algorithm – One Coin Suffices

Idea:
- Pick only one coin to store, name as King.
- Worst case \( \Theta(n) \) coins challenge the King -- give the King privilege: only be def enforced if lost multiple levels of challenge.
- Bound the sample complexity: limit the tosses of the King by budget.

Algorithm GAME-OF-COINS:
- For each arriving coin give the King a budget of \( O\left(\frac{1}{n}\right) \).
- To challenge the King, toss both coins \( \frac{n^2}{2} \) \( (1.5)^{i-1} \) times at level \( i \);
- A King is defeated only if it exhausts all its budget.

Analysis:
- Sample Complexity: At most \( 2n \cdot O\left(\frac{1}{n}\right) \) budgets \( \rightarrow O\left(\frac{n^2}{2}\right) \) coin tosses.
- Space Complexity: Only store 1 coin.
- Correctness:
  1. The coin* can exhaust the budget of other King (soundness)
  2. If coin* as the King -- budget sufficient in expectation.
  3. Control the variance:
     a) The budget behaves like random walks (but with flexible length).
     b) The challenging rule → budget distribution sub-exponential.
     c) Beating the union bound by Bernstein inequality (completeness).

Extensions of the Algorithm

Algorithm for top-\( k \) coins:
- Main technical contribution -- a delayed challenging rule & a potential function argument.
- Avoid eliminating any top-\( k \) coin -- use a buffer to swap defeated coins (correctness).
- Number of coins eventually decreases -- bounded sample complexity.

Noisy Comparisons and \( \varepsilon \)-PAC Multi-Armed Bandit (MAB):
- Noisy comparison -- \( O(k) \) space algorithm for finding top-\( k \) elements.
- No gap guarantee -- a \( O\left(\log^2(n)\right) \) space algorithm. Most recently, an extension to a 2-armed algorithm.

Extensions and open problems:
- The instance-sensitive sample complexity: \( H_2: O\left(\sum_{i>2} \frac{i}{2^i} \log\log\left(\frac{n}{2^{i-1}}\right)\right) \).
- Single-pass: achievable with random arrival of coins and a value \( O(H_2) \).
- Single-pass with lower bounds; arbitrary stream with \( O\left(\frac{1}{2^i}\right) \) passes [Jin et al., 2021].
- Open: tight number of passes to achieve \( O(H_2) \) sample complexity.

Our Contribution

Main Theorem (Assadi and Wang, 2020)

There exists a streaming algorithm that given \( n \) coins arriving in a stream with the gap parameter \( \Delta \) and confidence parameter \( \delta \), finds the most biased coin with probability at least \( 1 - \delta \) using \( O\left(\frac{\Delta}{\delta^2} \cdot \log\left(\frac{1}{\delta}\right)\right) \) coin tosses and a memory of a single coin.

- No sample-space trade-off!
- Preliminary: \( O\left(\log(n)\right) \), \( O\left(\log(n)\log(n)\right) \) and \( O\left(\log^2(n)\right) \) coins memory algorithms.
- Additional result: Top-\( k \) coin exploration with \( O(k) \) coin memory.
- Additional results: Noisy comparisons and Multi-Armed Bandits.

References
