CSC2420 - Algorithm Design, Analysis and Theory **Fall 2022**

Lecture 10: Streaming Algorithms

Lecturer: Allan Borodin *Scribe: Kevin Gao*

10.1 Heavy Hitter with Hashing

10.2 Frequency Moment

Let m_i denote the number of occurrences of the value i in the stream A. For $k \geq 0$, the kth frequency moment is $F_k = \sum_{i \in [n]} (m_i)^k$. The frequency moments are most often studied for integral k. Given an error bound ϵ and confidence bound δ, the objective of the frequency moment problem is to compute an estimate F'_k such that

$$\Pr[|F_k - F'_k| > \epsilon F_k] \leq \delta.$$

AMS(A, k)
1 $s_1 = \left(\frac{8}{\epsilon^2} m^{1-1/k}\right) / \delta^2$
2 $s_2 = 2 \log(1/\delta)$.
3 **for** $i = 1$ **to** s_2
4 $\quad j = 1$
5 $\quad$ **repeat** s_1 times
6 $\quad\quad$ randomly chooce $p \in [1, \ldots, m]$
7 $\quad\quad r = |\{q \mid q \geq p,\, a_p = a_q\}|$
8 $\quad\quad X_{ij} = m(r^k - (r-1)^k)$
9 $\quad\quad j = j + 1$
10 $\quad Y_i =$ mean of $\{X_{i,1}, X_{i,2}, \ldots, X_{i,s_1}\}$
11 $Y =$ median of $\{Y_1, \ldots, Y_{s_2}\}$
12 **return** Y

For simplicity, we assume that input stream length m is known but in reality, we can estimate and update it as the stream goes. For the analysis, we need to show that $\mathbb{E}[X] = F_k$ and that the variance $\mathrm{Var}[X]$ is small enough such that by the Chebyshev inequality, $\Pr[|Y_i - F_k| > \epsilon F_k]$ is also small.

10.3 Semi-streaming

Feigenbaum et al. introduced **semi-streaming** in order to consider sparse graph problem in the streaming model. Semi-streaming is an online model where we are not required to make immediate irrevocable decisions but can only maintain limited amount of information about the input stream. The goal is to maintain $\widetilde{O}(n)$ space.

For semi-streaming in the context of graph theory problems, we have a graph $G = (V, E)$ with $n = |V|$, $m = |E|$, and vertices or edges arrive online in sequence.

An online algorithm does not have to limit itself to $\widetilde{O}(n)$ memory. It could remember all edges seen so far and the order in which they have arrived, and a regular online algorithm can use any amount of space to make decisions.

10.3.1 Bipartite Matching in Semi-streaming Model

In the edge arrival model, there is no randomized semi-streaming algorithm with worst-case competitive ratio better than 1/2. The ratio is optimal since any greedy algorithm achieves 1/2 in streaming as well as in the online model.

In the vertex arrival model, the randomized KVV ranking algorithm also easily carries over to the semi-streaming model (either deterministic random order or randomized adversarial order). Surprisingly, Goel et al showed that there is a deterministic semi-streaming algorithm with adversarial order input sequences.

10.4 Weighted Majority Algorithm

The **weighted majority algorithm** and generalizations: We consider the problem of **finding the best expert**. Suppose we have say k "expert" (e.g. weatherman) and at every time t, they give a binary prediction (e.g. rain v.s. no rain). The predictions from different "experts" can be highly correlated. Without any knowledge of the subject matter, we want to construct an algorithm that tries to make predictions that will be nearly as good over time t as the best "expert".

```
WM(experts = [a_1, ...])
 1  w_i(0) = 1 for all i
 2  for t = 0 to ...
 3       if Σ_{i predicts 0 at t+1} w_i(t) ≥ (1/2) Σ_i w_i(t)
 4            predict 0 for time (t + 1)
 5       elseif Σ_{i predicts 1 at t+1} w_i(t) ≥ (1/2) Σ_i w_i(t)
 6            predict 1 for time (t + 1)
 7       for i = 1 to k
 8            if i made a mistake on (t + 1)st prediction
 9                 w_i(t + 1) = (1 − ε) w_i(t)
10            else
11                 w_i(t + 1) = w_i(t)
```

Theorem 10.1 (Weight majority algorithm)**.** *Let $m_i(t)$ be the number of mistakes of expert i after the first t expert predictions, and let $M(t)$ be the number of our mistakes. Then for any expert i (including the best expert),*

$$M(t) \leq \frac{2 \ln k}{\epsilon} + 2(1+\epsilon) m_i(t)$$

Proof: TBD ■

We can randomize the algorithm and remove the factor of 2. Instead of taking the weighted majority opinion, in each iteration t, choose the prediction of the ith expert with probability $w_i(t) / \sum_i w_i(t)$. This is a so-called natural randomization, as we have seen in monotone sublinar optimization.

Theorem 10.2 (Randomized weighted majority algorithm)**.** *For any expert,*

$$\mathbb{E}[M(t)] \leq \frac{\ln k}{\epsilon} + (1+\epsilon) m_i(t).$$

10.5 Multiplicative Weights Algorithm

The weighted majority algorithm can be generalized to the multiplicative weights algorithm. If the ith expert or decision is chosen on time t, it incurs a real valued cost/profit $m_i(t) \in [-1, 1]$. The algorithm then updates the weights

$$w_i(t+1) = (1 - \epsilon m_i(t))w_i(t)$$

Let $\epsilon \leq 1/2$ and $\Phi(t) = \sum_t w_i(t)$. On time t, we randomly select expert i with probability $w_i(t)/\Phi(t)$.

Theorem 10.3. *The expected cost of the multiplicative weight algorithm after T rounds is*

$$\sum_{t=1}^{T} \mathbf{m}(t) \cdot \mathbf{p}(t) \leq \frac{\ln k}{\epsilon} + \sum_{t=1}^{T} m_i(t) + \epsilon \sum_{t=1}^{T} |m_i(t)|$$

for any expert i.

10.6 Primality Testing

Let

$$\text{PRIMES} = \{N \mid N \text{ is prime}\}$$

where N is represented in binary so that $n = O(\log n)$.

It's easy to show that PRIMES is in co-NP and Vaughan showed that PRIMES is in NP. In 2022, Agarwal, Kayal, and Saxena (**AKS**) showed that primality is in deterministic polynomial time.