Lecture 11: Frequency Moments and Sketch Algorithms

October 1, 2025

1 Introduction to Heavy Hitters

F_0 is the most frequently occurring item in the stream. It is a brittle measure. In most applications we want to know the **heavy hitters**: items that occur very frequently.

1.1 Definition

From a theoretical perspective, we call an index $i \in [n]$ a heavy hitter if

$$F_i \geq \frac{m}{c}$$

for some sufficiently large constant c.

Alternatively, fix $F_i \geq \frac{m}{k}$ for some integer k.

2 Misra-Gries Algorithm

A classical algorithm shows that one can identify items i with $F_i \geq \frac{m}{k}$.

2.1 Algorithm Description

We have a data structure D that stores k items along with a counter for each. D is initialized to the empty set.

Implicitly, it defines an estimate $\hat{F}_i$ for each i:

- If $i \in D$ at the end, then $\hat{F}_i$ is the counter value
- Otherwise, it is 0

Theorem 1. *$|F_i - \hat{F}_i| \leq \frac{m}{k}$. Hence, if i is a heavy hitter, it will be in D. Space usage is $O(k)$.*

Although Misra-Gries is nice, it does not allow deletion and also does not provide a sketch.

3 Count-Min Sketch

Count-Min and Count sketches use hashing to identify heavy hitters and they have led to many applications.

Algorithm 1 Misra-Gries-k

```
D ← ∅
while stream is not empty do
    e is current item
    if e ∈ D then
        increment counter for e in D
    else
        if |D| < k then
            add e to D with counter value 1
        else
            decrease counter value by 1 for all current elements
            delete from D any element with counter set to 0
        end if
    end if
end while
Output: values stored in D and the counter values
```

3.1 Basic Idea

Suppose we use a hash function $h : [n] \to [ck]$ for some sufficiently large constant c. Then h spreads the n items into ck buckets. Suppose the heavy hitters are $i_1, \ldots, i_k$. We expect that they will not collide and we can use separate counts in each bucket.

We will use amplification as usual by considering multiple hash functions rather than a single one.

3.2 Algorithm (Cormode-Muthukrishnan)

Let $h_1, h_2, \ldots, h_d$ be d independent (pairwise independent) hash functions from $[n]$ to $[w]$.

Algorithm 2 Count-Min Sketch

```
Initialize C[ℓ][j] = 0 for all ℓ ∈ [d], j ∈ [w]
while stream is not empty do
    let (i, Δ) be current item
    for ℓ = 1 to d do
        C[ℓ][h_ℓ(i)] ← C[ℓ][h_ℓ(i)] + Δ
    end for
end while
for i = 1 to n do
    x̂_i = min_{ℓ=1}^{d} C[ℓ][h_ℓ(i)]
end for
```

where w is the width of the sketch and d is the number of independent hash functions we use. We use $w = \frac{e}{\varepsilon}$ and $d = \log \frac{1}{\delta}$.

Lemma 1. *Consider the strict turnstile model ($\bar{x} \geq 0$). Let $d = \log \frac{1}{\delta}$ and $w = \frac{e}{\varepsilon}$. Then:*

1. $\hat{x}_i \geq x_i$
2. $\Pr[\hat{x}_i - x_i \geq \varepsilon \|\bar{x}\|_1] \leq \delta$

Proof. Fix i. For $\ell \in [d]$:

$$
\begin{aligned}
Z_\ell = C[\ell][h_\ell(i)] - x_i &= \sum_{i': i' \neq i, h_\ell(i') = h_\ell(i)} x_{i'} \\
&= x_i + \sum_{i' \neq i, h_\ell(i') = h_\ell(i)} x_{i'}
\end{aligned}
$$

$$E[Z_\ell] = x_i + \sum_{i' \neq i} \Pr[h_\ell(i') = h_\ell(i)] \cdot x_{i'}$$

By pairwise independence: $\Pr[h_\ell(i') = h_\ell(i)] = \frac{1}{w}$

$$E[Z_\ell] = x_i + \frac{1}{w}\|\bar{x}\|_1$$

By Markov's inequality:

$$\Pr[Z_\ell \geq x_i + \varepsilon\|\bar{x}\|_1] \leq \frac{1}{e}$$

Thus:

$$\Pr[\min_\ell Z_\ell \geq x_i + \varepsilon\|\bar{x}\|_1] \leq \left(\frac{1}{e}\right)^d$$

by independence.
Choosing $d = \log \frac{1}{\delta}$, we have $\hat{x}_i - x_i \leq \varepsilon\|\bar{x}\|_1$ with high probability for all $i \in [n]$. □

Count-Min gives overestimates. Total space is $O(dw)$ counters $= O\left(\frac{1}{\varepsilon} \log \frac{1}{\delta} \log n\right)$.
Advantages: Simple, handles dependencies.
Disadvantages: Only handles $\bar{x} \geq 0$.
Exercise: Show that Count-Min is a linear sketch.

4 Count Sketch

Count Sketch is similar to Count-Min in using d independent hash functions but uses F_2 estimation ideas and median estimator instead of min.

4.1 Algorithm (Charikar-Chen-Farach-Colton)

Let $h_1, h_2, \ldots, h_d$ be independent hash functions from $[n]$ to $[w]$.
Let $g_1, g_2, \ldots, g_d$ be independent functions from $[n]$ to $\{-1, +1\}$.
$\hat{x}_i$ can be negative even if $\bar{x} \geq 0$. Cancellation can happen like in F_2 estimation.

Lemma 2. *Let $d = O(\log \frac{1}{\delta})$ and $w = O\left(\frac{1}{\varepsilon^2}\right)$. Then for any $i \in [n]$:*

1. $E[\hat{x}_i] = x_i$
2. $\Pr[|\hat{x}_i - x_i| \geq \varepsilon\|\bar{x}\|_2] \leq \delta$

Proof. Fix i. For $\ell \in [d]$, to make analysis easier, let $Y_{i'} = \mathbb{1}_{h_\ell(i') = h_\ell(i)}$ be the indicator for $h_\ell(i') = h_\ell(i)$.

Algorithm 3 Count Sketch

Initialize $C[\ell][j] = 0$ for all ℓ, j
while stream is not empty **do**
  let (i, Δ) be current item
  for $\ell = 1$ to d **do**
    $C[\ell][h_\ell(i)] \leftarrow C[\ell][h_\ell(i)] + g_\ell(i) \cdot \Delta$
  end for
end while
for $i \in [n]$ **do**
  $\hat{x}_i = \text{median}_{\ell=1}^{d}\{g_\ell(i) \cdot C[\ell][h_\ell(i)]\}$
end for

$$\begin{aligned} Z_\ell &= g_\ell(i) \cdot C[\ell][h_\ell(i)] \\ &= g_\ell(i) \cdot \left(\sum_{i' : h_\ell(i') = h_\ell(i)} g_\ell(i') \cdot Y_{i'} \cdot x_{i'} \right) \\ &= \sum_{i'} g_\ell(i) \cdot g_\ell(i') \cdot Y_{i'} \cdot x_{i'} \end{aligned}$$

$E[Z_\ell] = x_i$ by pairwise independence of g_ℓ.
We note that $E[Y_{i'}] = \frac{1}{w}$ and $E[g_\ell(i) \cdot g_\ell(i')] = 0$ for $i' \neq i$ by pairwise independence of h_ℓ.

$$\begin{aligned} \text{Var}(Z_\ell) &= E[Z_\ell^2] - (E[Z_\ell])^2 \\ &= E\left[\sum_{i'} g_\ell(i) \cdot g_\ell(i') \cdot Y_{i'} \cdot x_{i'} \right]^2 - x_i^2 \\ &= E\left[\sum_{i'} Y_{i'} \cdot x_{i'}^2 \right] - x_i^2 \\ &\leq \frac{1}{w} \|\bar{x}\|_2^2 \end{aligned}$$

Hence, using Chebyshev's inequality:

$$\Pr[|Z_\ell - x_i| \geq \varepsilon \|\bar{x}\|_2] \leq \frac{1}{w\varepsilon^2}$$

Via Chernoff bounds:

$$\Pr[\text{median}_\ell Z_\ell - x_i \geq \varepsilon \|\bar{x}\|_2] \leq \delta$$

□

5 Finding Heavy Hitters

Important: Sketches do not store directly the identity of the heavy hitters. Given $i \in [n]$, we can estimate $\hat{x}_i$ from the sketch. But outputting all i such that $\hat{x}_i$ is high requires a linear scan through $[n]$. Can maintain multiple data structures and use additional information to find the heavy hitters in $O(k)$ space and time.

6 Sparse Recovery

One nice and powerful application of Count Sketch is for sparse recovery. Suppose $\bar{x} \in \mathbb{R}^n$ is sparse or close to sparse, meaning that only k of the coordinates are non-zero. Can we recover $\bar{x}$ without knowing which of the coordinates are going to be important? Want to use only $O(k)$ space.

6.1 Definition

Given $\bar{x} \in \mathbb{R}^n$, let

$$\text{error}_k(\bar{x}) = \min_{\bar{z}:\|\bar{z}\|_0 \leq k} \|\bar{x} - \bar{z}\|_2$$

That is, what is the best k-sparse approximation to $\bar{x}$?

Offline, it is easy to compute:

$$\bar{z}_i^* = \begin{cases} x_i & \text{if } i \text{ is among the largest absolute value } k \text{ coordinates of } \bar{x} \\ 0 & \text{otherwise} \end{cases}$$

Can we find $\bar{z}^*$ in the streaming setting?

There exists a Count Sketch with $w = O\left(\frac{k}{\varepsilon^2}\right)$ and $d = O(\log n)$ that allows us to find a $\bar{z}$ such that $\|\bar{z}\|_0 \leq O(k)$ and with high probability

$$\|\bar{x} - \bar{z}\|_2 \leq C \cdot \text{error}_k(\bar{x})$$

In particular, if $\bar{x}$ is k-sparse, then we get exact recovery.

7 Compressed Sensing and RIP Matrices

Count Sketch guarantees that we can recover any sparse $\bar{x}$ with high probability. Can we guarantee probability 1 with a linear sketch? Yes!

There exist $\ell \times n$ matrices M for $\ell = O(k \log \frac{n}{k})$ such that given any k-sparse $\bar{x} \in \mathbb{R}^n$, one can recover $\bar{x}$ from $M\bar{x}$.

Note that $M\bar{x}$ takes $O(\ell)$ space, and since $\ell = O(k \log n)$, we are not storing much more than what we want to recover.

Such matrices are called **RIP matrices** (Restricted Isometry Property).

It turns out that a random $\ell \times n$ matrix with each entry chosen independently from a $\mathcal{N}(0, 1)$ Gaussian distribution satisfies the RIP. But we cannot easily verify that a given matrix is RIP.

This area is called **Compressed Sensing** and has several applications in signal processing.