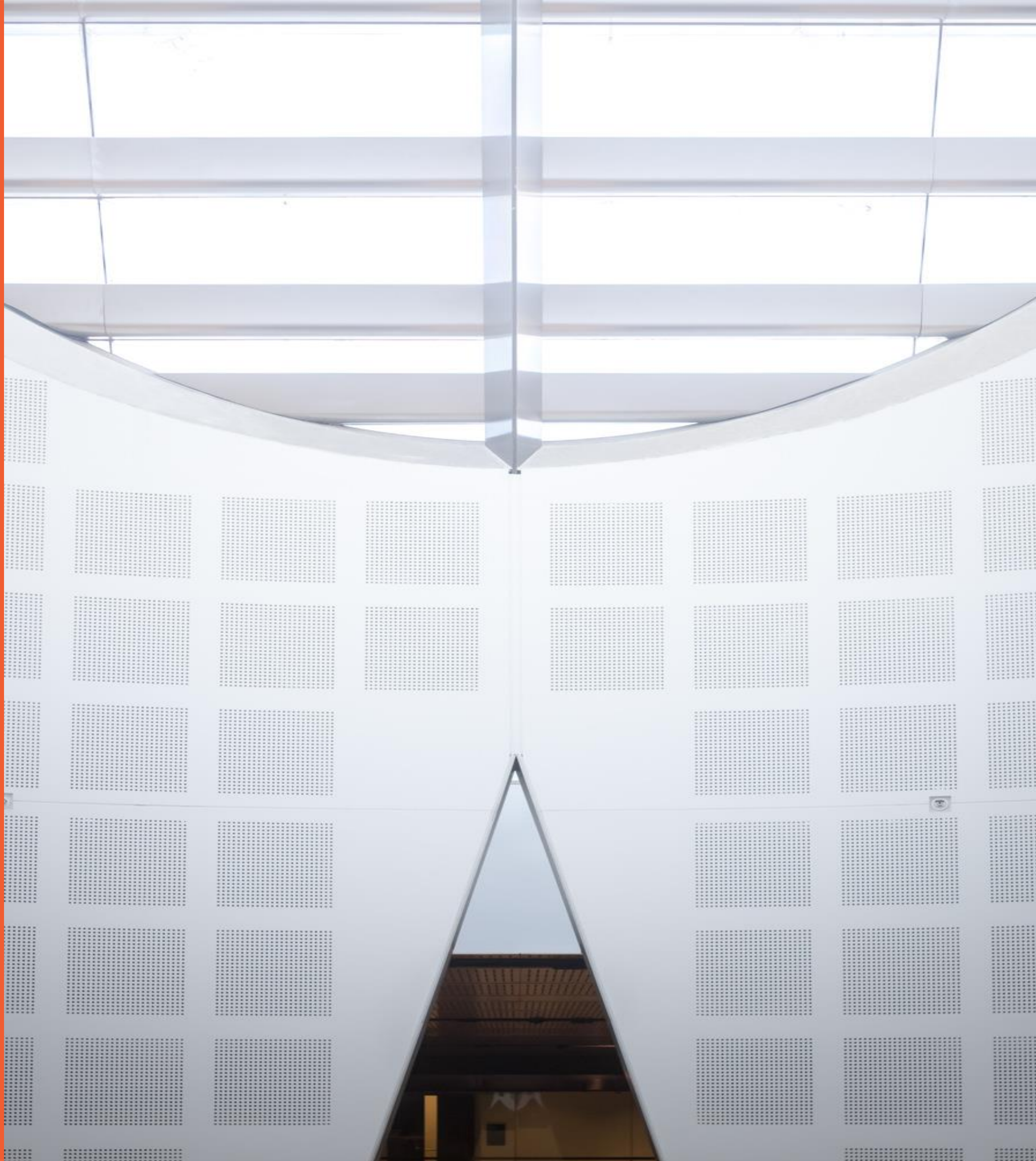
COMPx270: Randomised and Advanced Algorithms
Lecture 9: Streaming and Sketching II

Clément Canonne
School of Computer Science

THE UNIVERSITY OF SYDNEY



A question 🪡

You design a streaming algorithm A to solve some problem. The stream of data arrives.

A question 🪡

You design a streaming algorithm A to solve some problem. The stream of data arrives.

628, 516, 163, 509, 15, 499, 772, 588, 737, 439, 79, 866, 186, 18, 854, 459, 146, 518, 748, 737, 685, 188, 939, 724, 27, 719, 263, 795, 120, 573, 853, 132, 522, 3, 298, 123, 932, 993, 180, 674, 1, 619, 989, 142, 496, 178, 191, 524, 716, 501, 677, 712, 452, 768, 591, 551, 439, 397, 229, 214, 43, 639, 353, 610, 737, 203, 933, 279, 877, 30, 513, 518, 616, 714, 633, 804, 422, 731, 867, 184, 124, 881, 595, 193, 254, 240, 4, 260, 303, 319, 757, 723, 309, 365, 278, 512, 658, 233, 393, 875

A question 🪡

You design a streaming algorithm A to solve some problem. The stream of data arrives. A outputs its answer:

21.5

A question 🪡

Oh, no! That wasn't the end. More data arrives.

A question 🪡

Oh, no! That wasn't the end. More data arrives.

834, 992, 528, 12, 181, 274, 159, 150, 716, 71, 755, 4, 324, 398, 802, 176, 302, 941, 678, 934, 546, 753, 812, 47, 755, 721, 893, 53, 410

A question 🪡

Oh, no! That wasn't the end. More data arrives. A outputs its answer on this:

18.1

A question 🪡

How do you combine 21.5 and 18.1 to get the answer on the whole data stream?

Sketching

$\sigma \xrightarrow{A} S(\sigma)$: sketch (succinct "summary" of the stream σ)

Problem-dependent

$\sigma' \xrightarrow{A} S(\sigma')$

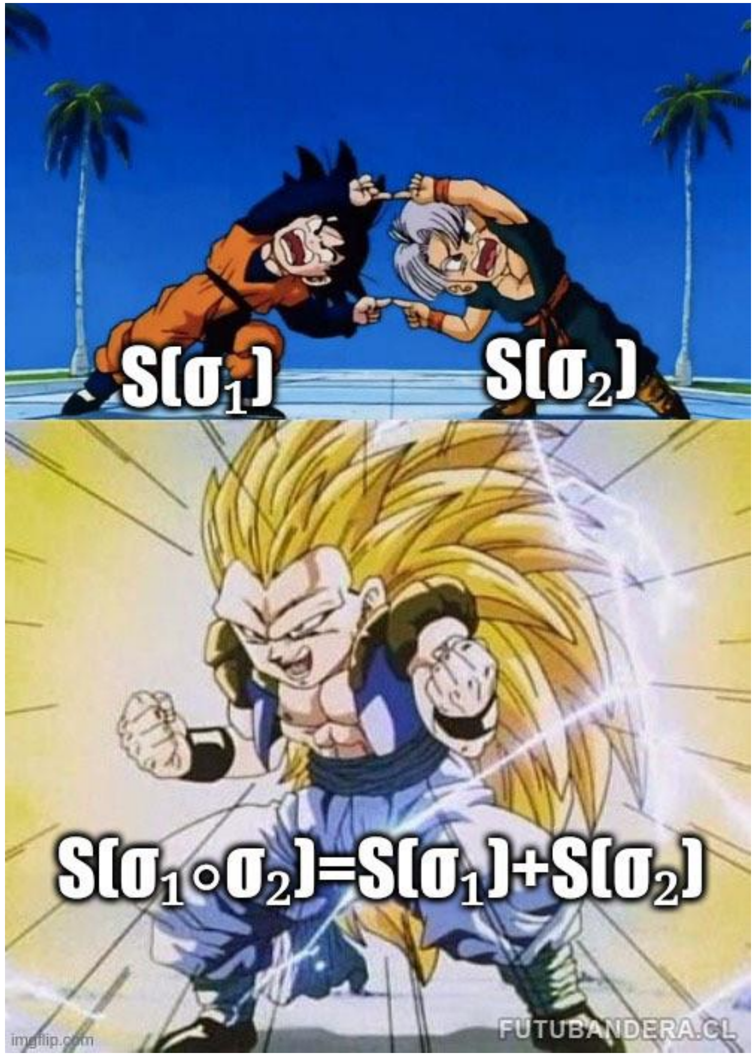
[From $S(\sigma)$, can get answer to problem]

$f(S(\sigma), S(\sigma')) \approx S(\sigma \circ \sigma')$

some way to combine things

(concatenation of σ, σ')

Even better: linear sketching



Even better: linear sketching

"$f = +$"

$$S(\sigma_1) + S(\sigma_2) = S(\sigma_1 \circ \sigma_2)$$

what is addition ?

Typically, addition of vectors in $\mathbb{R}^k$ (and sketches are vectors: $S(\sigma) \in \mathbb{R}^k$)

Equivalent:

$S : \mathbb{R}^n \to \mathbb{R}^k \qquad k \ll n$

$k \updownarrow \begin{pmatrix} \quad S \quad \end{pmatrix} \begin{pmatrix} f \end{pmatrix}$

$\xleftrightarrow{\quad n \quad}$

S is ① random
② not explicitly stored

Frequent Elements (Heavy Hitters)

Theorem 39. *The* MISRA-GRIES *algorithm is a* deterministic *one-pass algorithm which, for any given parameter $\varepsilon \in (0,1]$, provides $\hat{f}_1, \ldots, \hat{f}_n$ of all element frequencies such that*

$$f_j - \varepsilon m \leq \hat{f}_j \leq f_j, \qquad j \in [n]$$

with space complexity $s = O(\log(mn)/\varepsilon)$. (In particular, it can be used to solve the MAJORITY *problem in two passes.)*

Remember Misra-Gries?

3 claims:

1. MG is a sketching algorithm (yay!)
2. ~~You will see that in tutorial~~ Not quite. See [AC lecture notes, p.12, Exercise 1-3.
3. It is not a linear sketching algo

$\|x\|_\infty = \max_{i \in [n]} |x_i|$ + 1:

$\|f\|_1 = \sum_{i=1}^{n} |f_i|$

Guarantee of MG: outputs $\hat{f} \in \mathbb{R}^n$ (succinctly represented)

s.t. $\|\hat{f} - f\|_\infty \leq \varepsilon \|f\|_1$

in the cash register model.

Frequent Elements (Heavy Hitters): ℓ_1, ℓ_2, etc.

$$\|f\|_p = \left(\sum_{i=1}^{n} |f_i|^p\right)^{1/p}$$

$\|f\|_2 \approx$ measure of empirical variance of the stream

- $\|f\|_0$ ← # of distinct elements (F_0)

  $\|f\|_1$, $\|f\|_2$, $\|f\|_\infty$ (fact)

  $$\|f\|_\infty \leq \|f\|_2 \leq \|f\|_1$$

  "$\varepsilon\|f\|_2$ better guarantee than $\varepsilon\|f\|_1$"

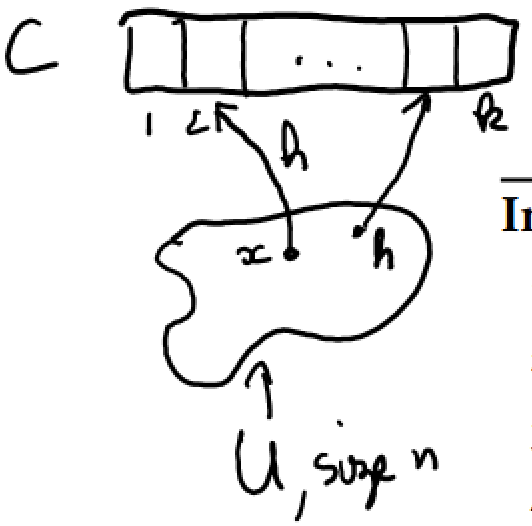
- In the rest: slight generalization:

  $$\sigma = (\underset{\substack{\uparrow\\\text{element}}}{(a_1}, \underset{\substack{\uparrow\\\text{\# of units}}}{c_1)}, \ldots, (a_m, c_m))$$

  $|c_i| \leq B = O(1)$

  $\to \|f\|_1 \leq m \cdot B$

Frequent Elements (Heavy Hitters): CountSketch (1/5)



Input: Parameters $\varepsilon, \delta \in (0, 1]$

Set $k \leftarrow O(1/\varepsilon^2)$, and initialize an array C of size k to zero

Pick $h: [n] \to [k]$ from a strongly universal hashing family

Pick $g: [n] \to \{-1, 1\}$ from a strongly universal hashing family

for all $1 \le i \le m$ **do**

  Get item $a_i = (j, c) \in [n] \times \{-B, \ldots, B\}$ $\triangleright$ Assume $B = O(1)$

  $C[h(j)] \leftarrow C[h(j)] + c \cdot g(j)$

Output: On query $j \in [n]$, **return** $\widehat{f}_j \leftarrow g(j) \cdot C[h(j)]$

Turnstyle model

$a_1 = (25, 3)$ $h(25) = 2$ $g(25) = -1$

$C[2] \leftarrow 0 + 3.(-1) = -3$

$a_{101} = (300, 1)$ $h(300) = 2$ $g(300) = -1$

$C[2] \leftarrow -3 + 1.(-1) = -4$

$a_{1005} = (25, -2)$

$C[2] \leftarrow -4 + (-2).(-1) = -2$

Frequent Elements (Heavy Hitters): CountSketch (2/5)

Input: Parameters $\varepsilon, \delta \in (0,1]$
1: Set $k \leftarrow O(1/\varepsilon^2)$, and initialize an array C of size k to zero
2: Pick $h\colon [n] \to [k]$ from a strongly universal hashing family
3: Pick $g\colon [n] \to \{-1,1\}$ from a strongly universal hashing family
4: **for all** $1 \le i \le m$ **do**
5: Get item $a_i = (j,c) \in [n] \times \{-B,\dots,B\}$ ▷ Assume $B = O(1)$
6: $C[h(j)] \leftarrow C[h(j)] + c \cdot g(j)$
Output: On query $j \in [n]$, **return** $\widehat{f}_j \leftarrow g(j) \cdot C[h(j)]$

$$C[j] = \sum_{i:h(i)=j} \underbrace{g(i)}_{\pm 1} \cdot f_i$$

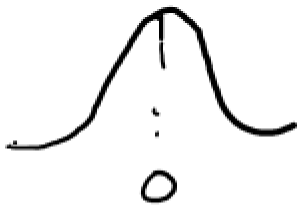
Example: $\underbrace{g(25)C[2]}_{\hat{f}_{25}} = \underbrace{(\underbrace{g(25)}_{-1})^2}_{1} f_{25} + \sum_{\substack{i \neq 25 \\ h(i)=j}} g(25) g(i) f_i$

$$= f_{25} + \underbrace{\sum_{\substack{i \neq 25 \\ h(i)=j}} (\pm 1) f_i}_{\text{noise with expectation } 0}$$

Intuition

$$\sum_{i=1}^{n} X_i a_i \approx \mathcal{N}(0, \|a\|_2^2)$$

X_i: ± 1 uniformly; a_i: fixed vector

$f_{25} +$ 

"Expect" $\hat{f}_i \simeq f_i \pm O(\|f\|_2)$

Frequent Elements (Heavy Hitters): CountSketch (3/5)

Input: Parameters $\varepsilon, \delta \in (0,1]$

1: Set $k \leftarrow O(1/\varepsilon^2)$, and initialize an array C of size k to zero
2: Pick $h: [n] \to [k]$ from a strongly universal hashing family
3: Pick $g: [n] \to \{-1,1\}$ from a strongly universal hashing family
4: **for all** $1 \le i \le m$ **do**
5: Get item $a_i = (j,c) \in [n] \times \{-B,\ldots,B\}$ ▷ Assume $B = O(1)$
6: $C[h(j)] \leftarrow C[h(j)] + c \cdot g(j)$

Output: On query $j \in [n]$, **return** $\hat{f}_j \leftarrow g(j) \cdot C[h(j)]$

Claim CountSketch is a sketching algorithm*.

Proof. By looking at step "output".

* using the same hash functions g, h for all sketches.

Claim. $\mathbb{E}[\hat{f}] = f$.

Proof. Fix $j \in [n]$.

$$\hat{f}_j = g(j)\, C[h(j)] = g(j) \sum_{i:\, h(i)=h(j)} g(i) f_i$$

$$= \underbrace{g(j)\cdot g(j)}_{=1} f_j + \sum_{\substack{i:\, i \neq j \\ h(i)=h(j)}} g(j) g(i) f_i$$

$$\mathbb{E}[\hat{f}_j] = f_j + \sum_{i:\, i\neq j} \mathbb{E}[\mathbb{1}_{h(i)=h(j)}] \underbrace{\mathbb{E}[g(j)g(i)]}_{=0 \text{ (pairwise indep}^t\text{)}} f_i$$

$$= f_j$$

$\square$

Frequent Elements (Heavy Hitters): CountSketch (3/5)

Input: Parameters $\varepsilon, \delta \in (0,1]$
1: Set $k \leftarrow O(1/\varepsilon^2)$, and initialize an array C of size k to zero
2: Pick $h\colon [n] \to [k]$ from a strongly universal hashing family
3: Pick $g\colon [n] \to \{-1,1\}$ from a strongly universal hashing family
4: **for all** $1 \le i \le m$ **do**
5: Get item $a_i = (j,c) \in [n] \times \{-B,\dots,B\}$ ▷ Assume $B = O(1)$
6: $C[h(j)] \leftarrow C[h(j)] + c \cdot g(j)$
Output: On query $j \in [n]$, **return** $\hat{f}_j \leftarrow g(j) \cdot C[h(j)]$

Claim $\forall j \quad \mathrm{Var}[\hat{f}_j] \le \dfrac{\|f\|_2^2}{k}$ ← size of C

Proof
$$\mathrm{Var}\,\hat{f}_j = \mathbb{E}[\hat{f}_j^2] - \mathbb{E}[\hat{f}_j]^2 = \mathbb{E}[\hat{f}_j^2] - f_j^2.$$

$$\mathbb{E}[\hat{f}_j^2] = \mathbb{E}\Big[\Big(\sum_{j'} \mathbb{1}_{h(j')=h(j)}\, g(j')g(j) f_{j'}\Big)^2\Big]$$

$$= \mathbb{E}\Big[\sum_{j_1,j_2} \mathbb{1}_{h(j_1)=h(j)=h(j_2)}\, g(j_1)g(j_2)\cancel{g(j)^2}^{\,1} f_{j_1} f_{j_2}\Big]$$

$$= \sum_{j_1,j_2} \mathbb{E}[\mathbb{1}_{h(j_1)=h(j_2)=h(j)}]\, \underbrace{\mathbb{E}[g(j_1)g(j_2)]}_{=\mathbb{1}_{j_1=j_2}} f_{j_1} f_{j_2}$$

$$= \sum_{j_1} \underbrace{\Pr[h(j_1)=h(j)]}_{1 \text{ if } j_1=j,\ \frac{1}{k} \text{ o/w (pairwise ind}^{t})} f_{j_1}^2$$

$$= f_j^2 + \frac{1}{k}\sum_{j_1\ne j} f_{j_1}^2 \le f_j^2 + \frac{1}{k}\sum_{j'} f_{j'}^2 = f_j^2 + \frac{\|f\|_2^2}{k}.$$

From there, Chebyshev:
$$\Pr[|\hat{f}_j - f_j| > \varepsilon\|f\|_2] \le \frac{\|f\|_2^2/k}{\varepsilon^2\|f\|_2^2} = \frac{1}{k\varepsilon^2} \le \frac{1}{3}.$$

Frequent Elements (Heavy Hitters): CountSketch (5/5)

Theorem 44. *The (median trick version of the)* COUNTSKETCH *algorithm is a* randomised *one-pass sketching algorithm which, for any given parameters* $\varepsilon, \delta \in (0,1]$, *provides a (succinctly represented) estimate* $\widehat{f}$ *of frequency vector* f *of the stream such that, for every* $j \in [n]$

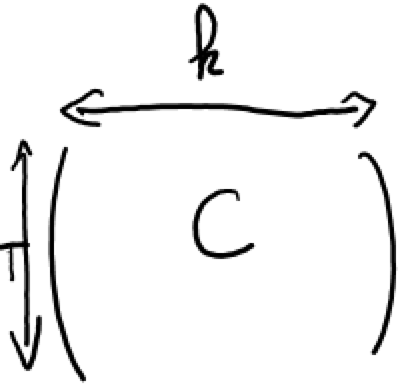
$$\Pr\left[\left| \widehat{f}_j - f_j \right| \leq \varepsilon \|f_{-j}\|_2 \right] \geq 1 - \delta$$

with space complexity

$$s = O\left(\frac{\log(nm)}{\varepsilon^2} \log \frac{1}{\delta} \right).$$

Frequent Elements (Heavy Hitters): CountMinSketch (1/4)

⊛ Cash register model



each row has its own h_t

Input: Parameters $\varepsilon, \delta \in (0, 1]$

Set $k \leftarrow O(1/\varepsilon)$ and $T \leftarrow O(\log(1/\delta))$, and initialize a two-dimensional array C of size $T \times k$ to zero

Pick $h_1, \ldots, h_T : [n] \rightarrow [k]$ independently from a strongly universal hashing family

for all $1 \leq i \leq m$ **do**

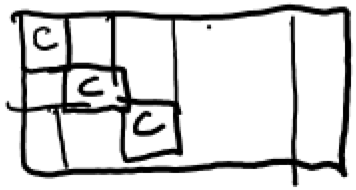
  Get item $a_i = (j, c) \in [n] \times \{0, \ldots, B\}$ ▷ Assume $B = O(1)$

  for all $1 \leq t \leq T$ **do** ← ⊛

    $C[t][h_t(j)] \leftarrow C[t][h_t(j)] + c$

Output: On query $j \in [n]$, **return** $\hat{f}_j \leftarrow \min_{1 \leq t \leq T} C[t][h_t(j)]$

$h_1(25) = 1$
$h_2(25) = 2$
$h_3(25) = 3$



C

"Bloom filter that counts"

Input: Parameters $\varepsilon, \delta \in (0, 1]$

1: Set $k \leftarrow O(1/\varepsilon)$ and $T \leftarrow O(\log(1/\delta))$, and initialize a two-dimensional array C of size $T \times k$ to zero
2: Pick $h_1, \ldots, h_T : [n] \to [k]$ independently from a strongly universal hashing family
3: **for all** $1 \le i \le m$ **do**
4: Get item $a_i = (j, c) \in [n] \times \{0, \ldots, B\}$ ▷ Assume $B = O(1)$
5: **for all** $1 \le t \le T$ **do**
6: $C[t][h_t(j)] \leftarrow C[t][h_t(j)] + c$

Output: On query $j \in [n]$, **return** $\widehat{f}_j \leftarrow \min_{1 \le t \le T} C[t][h_t(j)]$

Frequent Elements (Heavy Hitters): CountMinSketch (2/4)

Claim. Count Min Sketch is a sketching algo.*

Claim. $\hat{f}_j \geq f_j \quad \forall j.$

(Cash register model)

Claim. Space $O\left(\frac{\log(mn)}{\varepsilon}\log(1/\delta)\right)$.

"Better" than CountSketch.

* using the same hash functions for each stream.

Frequent Elements (Heavy Hitters): CountMinSketch (3/4)

Input: Parameters $\varepsilon, \delta \in (0,1]$
1: Set $k \leftarrow O(1/\varepsilon)$ and $T \leftarrow O(\log(1/\delta))$, and initialize a two-dimensional array C of size $T \times k$ to zero
2: Pick $h_1, \dots, h_T : [n] \to [k]$ independently from a strongly universal hashing family
3: **for all** $1 \le i \le m$ **do**
4: Get item $a_i = (j, c) \in [n] \times \{0, \dots, B\}$ ▷ Assume $B = O(1)$
5: **for all** $1 \le t \le T$ **do**
6: $C[t][h_t(j)] \leftarrow C[t][h_t(j)] + c$
Output: On query $j \in [n]$, **return** $\hat{f}_j \leftarrow \min_{1 \le t \le T} C[t][h_t(j)]$

Claim $\forall j,\ \Pr[|\hat{f}_j - f_j| \geq \varepsilon \|f\|_1] \leq \delta.$

Proof:

① $\hat{f}_j = f_j +$ "noise (≥ 0) from collisions"

$\hat{f}_j = \min_{1 \le t \le T} C[t][h_t(j)]$

$= \min_t \Big(f_j + \underbrace{\sum_{\substack{j' \neq j \\ h_t(j') = h(j)}} f_{j'}}_{\text{let's bound this}} \Big)$

② $\mathbb{E}[\text{Ⓐ}] = \dfrac{\|f_{-j}\|_1}{k}$

③ Markov.

$\Pr[\text{Ⓐ} \geq \varepsilon \|f\|_1] \leq \dfrac{1}{\varepsilon k} \leq \dfrac{1}{2}$

④ min bad $\Leftrightarrow$ all T are bad!

Ⓐ $\Pr[\ ] \leq \dfrac{1}{2^T}$

Frequent Elements (Heavy Hitters): CountMinSketch (4/4)

Theorem 45. *The* CountMinSketch *algorithm is a* randomised *one-pass sketching algorithm which, for any given parameters* $\varepsilon, \delta \in (0, 1]$, *provides a (succinctly represented) estimate* $\widehat{f}$ *of frequency vector* f *of the stream such that, for every* $j \in [n]$

$$\Pr\left[\left| \widehat{f}_j - f_j \right| \leq \varepsilon \|f_{-j}\|_1 \right] \geq 1 - \delta$$

with space complexity

$$s = O\left(\frac{\log(nm)}{\varepsilon} \log \frac{1}{\delta} \right).$$

(Moreover, $\widehat{f}_j$ *is always an overestimate:* $\widehat{f}_j \geq f_j$ *for all* $j \in [n]$*.)*

Wait a minute...

This seems strictly worse than Misra-Gries!

- Randomised instead of deterministic!
- Uses more space!
- Also in the cash register model!
- Also an ℓ_1 guarantee!

Wait a minute...

This seems strictly worse than Misra-Gries!

- Randomised instead of deterministic!
- Uses more space!
- Also in the cash register model!
- Also an ℓ_1 guarantee!

Yes, but:

- Linear sketch!
- Much faster per time step!
- Can be extended to the strict turnstile model!

https://sites.google.com/site/countminsketch/

Recap

- CS, CMS, MG solve the same problem
- Different guarantees (CS/CMS, MG)
- Different types of sketches (CS, CMS / MG)
- Different space usage