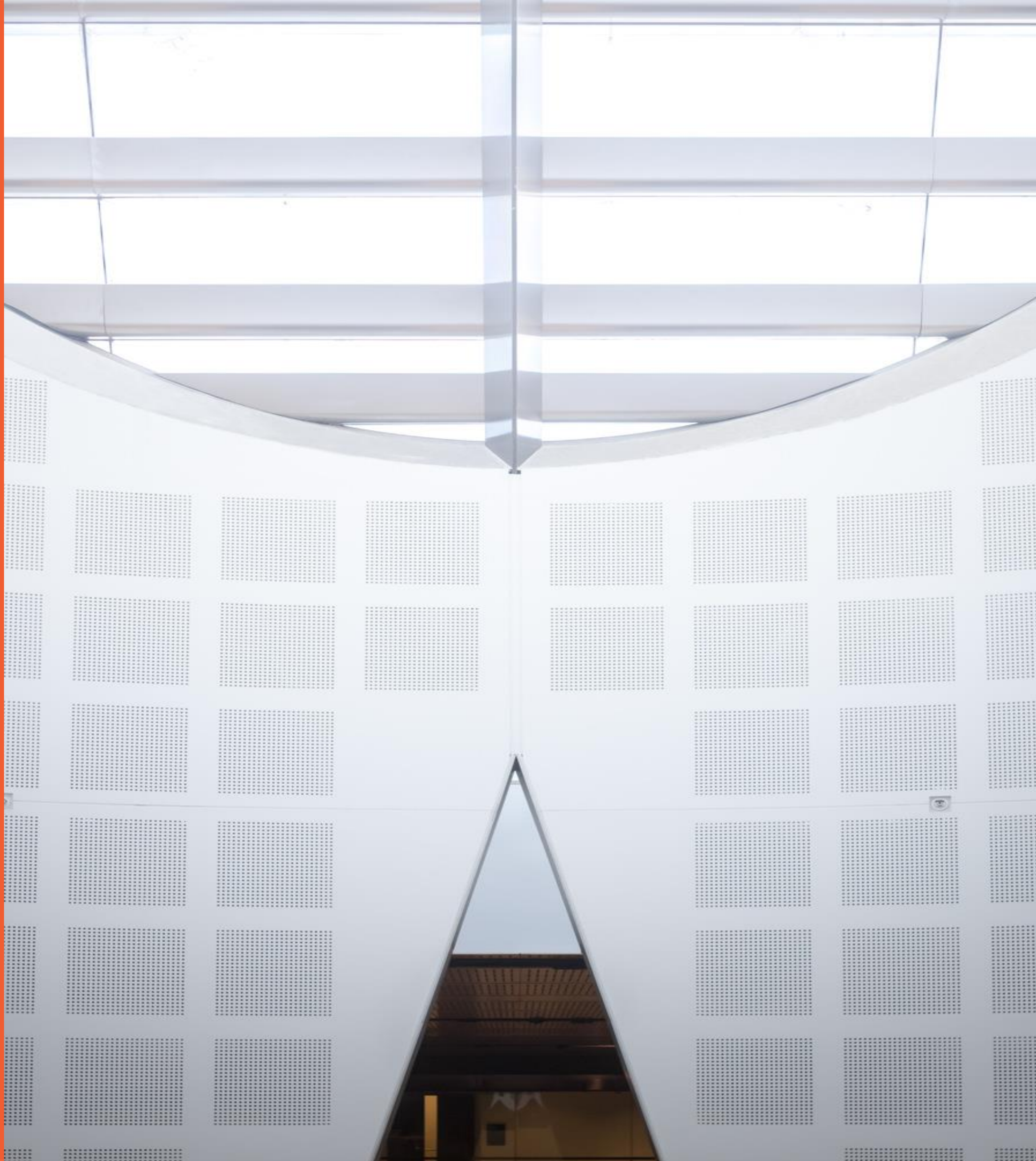
COMPx270: Randomised and Advanced Algorithms
Lecture 8: Streaming and Sketching I

Clément Canonne

School of Computer Science

THE UNIVERSITY OF SYDNEY



A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(1,2)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(2,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(1,2)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,6)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(1,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,6)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,6)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

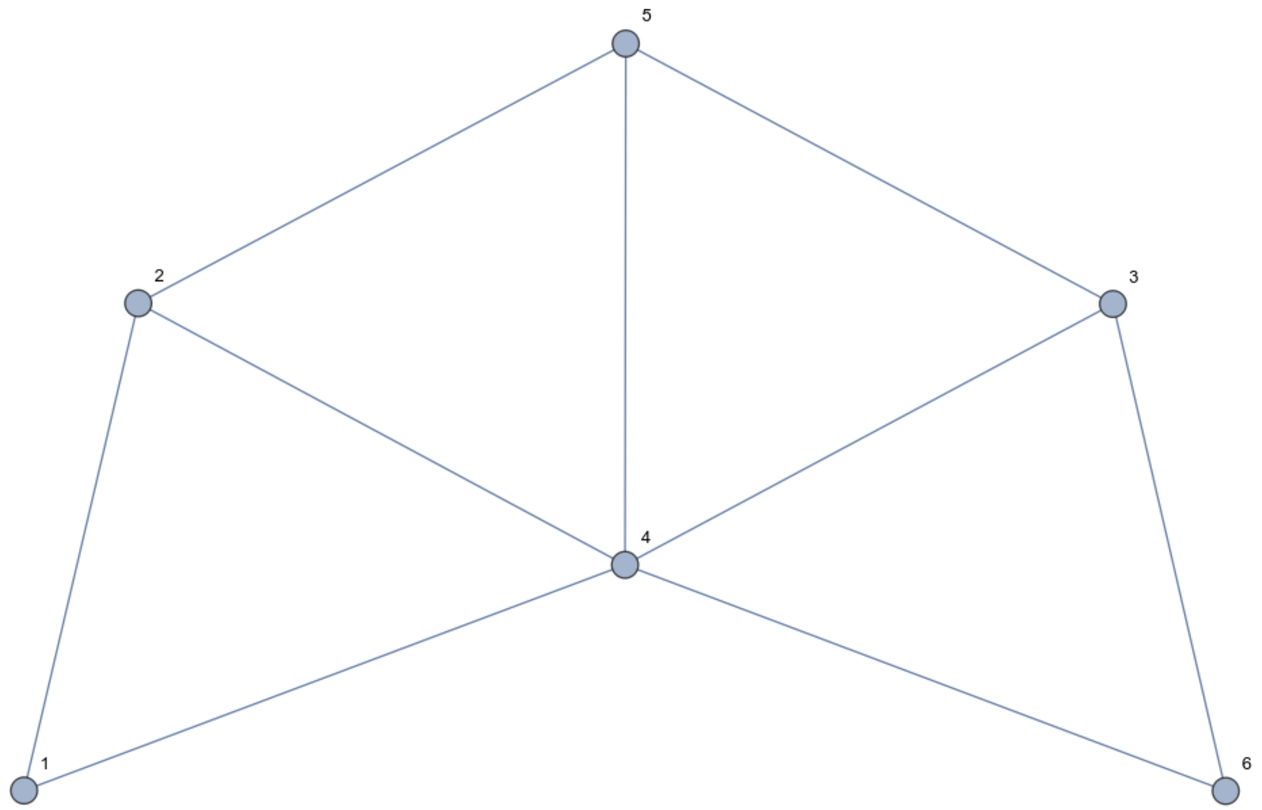
(1,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,6)

A question (an answer)



Streaming algorithms: what? (1/3)

Streaming algorithms: what? (2/3)

so less than $O(m \log n)$ or $O(n \log m)$

- Low memory: cannot store whole input
- Input comes as a stream: sequence of length m

$$\sigma = (a_1, a_2, \dots, a_m)$$

$$a_i \in \mathcal{X}, \quad |\mathcal{X}| = n$$

Worst-case (arbitrary) order.

- p-pass algorithms get to see σ p times ($p = 1$ for us*)

$\xrightarrow{p} \sigma$

- cash register: don't remove parts of the input (that would be "turnstile")

SPACE: $o(\min(n, m))$ ("sublinear")

hope: $O(\log(mn)) = O(\log m + \log n)$

very good: $\mathrm{polylog}(m, n)$

Streaming algorithms: what? (3/3)

- Randomised algorithms
- Approximate: want to compute some value $v \geq 0$

  we're OK with $\hat{v} \overset{\varepsilon}{\approx} v$

  ?

① Multiplicative:

$$\Pr[\,|\hat{v} - v| \geq \varepsilon v\,] \leq \delta$$

$(1 \pm \varepsilon)$ approx

② Additive:

$$\Pr[\,|\hat{v} - v| \geq \varepsilon\,] \leq \delta$$

$\pm\varepsilon$ approx

First example: Majority

A.k.a. "special case of Heavy Hitters"

- MAJORITY : is there an elem^t appearing $\geq 50\%$ of the time in the stream? (If so, which one(s)?)

$$\sigma = (\sigma_1, \dots, \sigma_m) \in [n]^m$$

$\forall j \in [n]$ $\quad f_j = \#(\text{times } j \text{ appears in } \sigma) = \sum_{i=1}^{m} \mathbb{1}_{\sigma_i = j}$ $\quad$ frequency of element j

Frequency vector $\quad \vec{f} = (f_1, \dots, f_n)$

- $0 \leq f_j \leq m \quad \forall j \in [n]$
- $F_1 = \|\vec{f}\|_1 = \sum_{j=1}^{n} f_j = m$

First example: Majority (Frequency Estimation)

MAJORITY: "Is there $j \in [n]$ s.t. $f_j \geq \frac{m}{2}$?" (at most 2 of them)

ε-HH: "Is there $j \in [n]$ s.t. $f_j \geq \varepsilon m$?" (at most $\frac{1}{\varepsilon}$ of them)

Want to solve this in one pass.

We'll see two passes, but <u>deterministically</u>

First example: the Misra-Gries algorithm (1/3)

MISRA-GRIES returns $\hat{f}_1, \ldots, \hat{f}_n$ (a succinct representation of them)

s.t. $f_j - \varepsilon m \leq \hat{f}_j \leq f_j \quad \forall j$

in one pass.

⚠ only $O(\frac{1}{\varepsilon})$ estimates are non-zero: only returns these $\hat{f}_j$

→ TO SOLVE MAJORITY IN TWO PASSES

Pass ① Run M-G on σ with $\varepsilon = \frac{1}{4}$

Pass ② Count exactly the frequency for all j's s.t. $\hat{f}_j \geq \frac{m}{4}$

If $f_j \geq \frac{m}{2}$

$\hat{f}_j \geq \frac{m}{4} > 0$

If $\hat{f}_j \geq \frac{m}{4}$

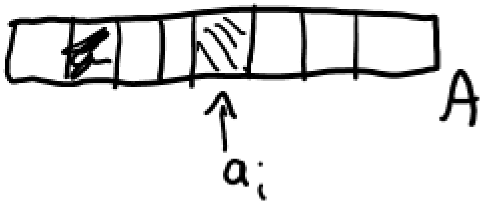
$f_j \geq \frac{m}{4}$

→ at most 4 of them

First example: the Misra-Gries algorithm, alternative view (2/3)

$A \leftarrow n$ zeroes (use a BST to save space)
$k \leftarrow 1/\varepsilon$
At step $1 \le i \le m$:

- get a_i
- If $A[a_i] > 0$
  $A[a_i] += 1$
- If $A[a_i] = 0$ and $|A| < k-1$ ← # of non-zero entries
  $A[a_i] = 1$
- If $A[a_i] = 0$ and $|A| = k-1$
  For all j s.t. $A[j] > 0$ ← $A[a_i]=1$
  $A[j] = A[j] - 1$



At the end: return all j's (and $A[j]$) s.t. $A[j] > 0$

SPACE: $O(k \cdot (\log m + \log n))$
$= O\left(\frac{\log(nm)}{\varepsilon}\right)$

CORRECTNESS $(\hat{f}_j = A[j])$

① $\forall j, \quad \hat{f}_j \le f_j$ "clear"

② $\forall j \quad \hat{f}_j \ge f_j - \frac{m}{k}$

Claim: "can't decrement an element too many times"

Each decrement corresponds to exactly k prior increments
→ at most $\frac{m}{k}$ decrements

First example: the Misra-Gries algorithm (3/3)

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.)*

Second example: Approximate Counting

$n = 2 \quad \mathcal{X} = \{0, 1\}$ Want $d = \sum_{i=1}^{m} a_i$

- $O(\log m)$ trivial: counter (deterministic)
- 2-estimate $O(\log \log m)$ space (Morris)

randomized

Second example: Approximate Counting and the Morris Counter

1: $x \leftarrow 0$
2: **for all** $1 \leq i \leq m$ **do**
3:     Get item $a_i \in \{0, 1\}$
4:     **if** $a_i = 1$ **then**
5:         $r_i \leftarrow \text{Bern}(1/2^x)$ ▷ Independent of previous choices.
6:         $x \leftarrow x + r_i$
7: **return** $\hat{d} \leftarrow 2^x - 1$

$\mathbb{E}[\hat{d}] = ?$

$\mathrm{Var}[\hat{d}] = ?$

Second example: Approximate Counting and the Morris Counter

$C_0 \leftarrow 1$
for all $1 \leq i \leq m$ **do**
    Get item $a_i \in \{0, 1\}$
    if $a_i = 1$ **then**
        $r_i \leftarrow \text{Bern}(1/C_{i-1})$ ▷ Independent of previous choices.
    else $r_i \leftarrow 0$
    $C_i \leftarrow 2^{r_i} C_{i-1}$
return $\widehat{d} \leftarrow C_m - 1$

Claim:

$$\mathbb{E}[C_m] = d+1$$

$$\text{Var}[C_m] = O(d^2) \quad \left(= \binom{d}{2}\right)$$

Throwback: Law of Total Expectation (and Friends)

$$\mathbb{E}[\mathbb{E}[X|Y]] = \mathbb{E}[X]$$

$$\mathbb{E}[f(Y)|Y] = f(Y)$$

$$\mathbb{E}[f(Y)X|Y] = f(Y)\,\mathbb{E}[X|Y]$$

Second example: the Morris Counter (1/3)

```
C_0 ← 1
for all 1 ≤ i ≤ m do
    Get item a_i ∈ {0,1}
    if a_i = 1 then
        r_i ← Bern(1/C_{i-1})        ▷ Independent of previous choices.
    else r_i ← 0
    C_i ← 2^{r_i} C_{i-1}
return d̂ ← C_m − 1
```

- $\mathbb{E}[C_0] = C_0 = 1$
- $\mathbb{E}[C_{i+1} \mid C_i] \overset{?}{=}$ (If $a_{i+1} = 0$, $C_{i+1} = C_i$)

$$\underset{\text{If } a_{i+1}=1}{\uparrow} \quad \frac{1}{C_i} \cdot 2C_i + \left(1 - \frac{1}{C_i}\right) \cdot C_i$$

$$= C_i + 1$$

$$\mathbb{E}[C_{i+1} \mid C_i] = C_i + a_{i+1}$$

$$\rightarrow \mathbb{E}[C_{i+1}] = \mathbb{E}[\mathbb{E}[C_{i+1} \mid C_i]] = \mathbb{E}[C_i] + a_{i+1} = \mathbb{E}[C_{i-1}] + a_i + a_{i+1} = \ldots$$

$$\rightarrow \mathbb{E}[C_m] = 1 + \sum_{i=1}^{m} a_i = 1 + d$$

✓

Second example: the Morris Counter (2/3)

```
C_0 ← 1
for all 1 ≤ i ≤ m do
    Get item a_i ∈ {0,1}
    if a_i = 1 then
        r_i ← Bern(1/C_{i−1})        ▷ Independent of previous choices.
    else r_i ← 0
    C_i ← 2^{r_i} C_{i−1}
return d̂ ← C_m − 1
```

$$\mathrm{Var}[C_m] = \mathbb{E}[C_m^2] - \mathbb{E}[C_m]^2$$

known! $(d+1)^2$

$$\mathbb{E}[C_m^2] = ?$$

- $\mathbb{E}[C_0^2] = C_0^2 = 1$
- $\mathbb{E}[C_{i+1}^2 \mid C_i] = \begin{cases} C_i^2 & \text{if } a_{i+1} = 0 \\ \frac{1}{C_i} \cdot 4C_i^2 + \left(1 - \frac{1}{C_i}\right) C_i^2 = 3C_i + C_i^2 & \text{if } a_{i+1} = 1 \end{cases}$

$$= C_i^2 + a_{i+1}(2 + a_{i+1}) C_i$$

$$+ \sum_{i=1}^{m} (\ldots)$$

$$\rightarrow \mathbb{E}[C_m^2] = \mathbb{E}[\mathbb{E}[C_m^2 \mid C_{m-1}]] = \ldots = 1 + 3\frac{d(d+1)}{2}$$

$$\rightarrow \mathrm{Var}\, C_m = \frac{d(d-1)}{2} \quad \checkmark$$

Second example: the Morris Counter (3/3)

$\mathbb{E}[C_m] = d+1$ ✓

$\mathrm{Var}[C_m] = \Theta(d^2)$ ✗

Chebyshev:

$C_m = \mathbb{E}[C_m] \pm \sqrt{\mathrm{Var}[C_m]}$ w.p. 99%

useless guarantee.

Doom? No

① "Meh" guarantee, w.p. 51% ✓

② "Good" guarantee, w.p. 51%
take average

③ "Good" guarantee w.p. $1-\delta$
median trick

Second example: the Morris Counter, Median-of-Means

Theorem 40. *The medians-of-means version of the* Morris Counter *is a* randomised *one-pass algorithm which, for any given parameters* $\varepsilon, \delta \in (0,1]$, *provides an estimate* $\hat{d}$ *of the number* d *of non-zero elements of the stream such that*

$$\Pr\Big[(1-\varepsilon)d \leq \hat{d} \leq (1+\varepsilon)d \Big] \geq 1-\delta$$

with space complexity

$$s = O\left(\underbrace{\frac{\log\log m}{\varepsilon^2}}_{\text{average over } T=O\left(\frac{1}{\varepsilon^2}\right) \text{ instances}} \cdot \underbrace{\log\frac{1}{\delta}}_{\text{Median trick over } T'=O\left(\log\frac{1}{\delta}\right) \text{ instances}} \right)$$

that is, doubly logarithmic *in* m.

③ Median trick over $T' = O(\log \frac{1}{\delta})$ instances

② average over $T = O(\frac{1}{\varepsilon^2})$ instances to reduce variance, then Chebyshev

Did we need to do that?

No.

No need for median-of-means here!

⊕ better space...

Second example: the Morris Counter, careful version (1/2)

Morris from before

$$C \leftarrow 2C \text{ wp } \frac{1}{C} \quad (\text{when } a := 1)$$

Morris, better

$$C \leftarrow (1+\alpha)C \quad \text{wp} \quad \frac{1}{\alpha C}$$

$$\boxed{\alpha = 2\varepsilon^2\delta}$$

(Estimate: $(1+\alpha)^x - 1$)

Second example: the Morris Counter, careful version (2/2)

Theorem 41. *The "careful" version of* Morris Counter *is a* randomised *one-pass algorithm which, for any given parameters* $\varepsilon, \delta \in (0,1]$, *provides an estimate* $\widehat{d}$ *of the number* d *of non-zero elements of the stream such that*

$$\Pr\Big[\, (1-\varepsilon)d \leq \widehat{d} \leq (1+\varepsilon)d \,\Big] \geq 1-\delta$$

with space complexity

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

that is, doubly logarithmic in m and logarithmic in $1/\varepsilon$.

Third example: Distinct Elements

Approximate $F_0 = \underbrace{\sum_{j=1}^{n} \mathbb{1}_{f_j > 0}}_{\text{"}d\text{" (new name)}}$

Return $\hat{d} = (1 \pm \varepsilon) d$

Third example: Distinct Elements, the Tidemark (AMS) algorithm (1/5)

$\varepsilon = \Theta(1)$

Pick $h\colon [n] \to [n]$ from a strongly universal hashing family
$z \leftarrow 0$
for all $1 \le i \le m$ **do**
  Get item $a_i \in [n]$
  if $\text{zeros}(h(a_i)) \ge z$ **then**
    $z \leftarrow \text{zeros}(h(a_i))$
return $\sqrt{2} \cdot 2^z$

$\text{zeros}(k) =$ largest i s.t. $2^i \mid k$

$\boxed{0\,0\,1\,1\,1\,1\,0\,1\,1\,\underbrace{0\,0\,1\,0}_{\text{zeros}(k)}}$

Third example: Distinct Elements, the Tidemark (AMS) algorithm (2/5)

```
Pick h: [n] → [n] from a strongly universal hashing family
z ← 0
for all 1 ≤ i ≤ m do
    Get item a_i ∈ [n]
    if zeros(h(a_i)) ≥ z then
        z ← zeros(h(a_i))
return √2 · 2^z
```

Space $s = O(\log n)$ (hash function)

$+\ O(\log \log n)$ (storing z)

$= O(\log n)$

Intuition: "Since h behaves like a random function"

→ d values $h(j_1), \ldots, h(j_d)$ uniformly distributed in $[n]$

→ for each, proba to have at least r trailing zeroes in binary is $\frac{1}{2} \cdots \frac{1}{2} = \frac{1}{2^r}$

→ for $r \approx \log_2 d$, by "union bound" we "should" have at least one hash with r trailing zeroes (w. cst proba)

→ for $r \gg \log_2 d$, by union bound very unlikely to have any hash with r trailing zeroes

Third example: Distinct Elements, the Tidemark (AMS) algorithm (3/5)

1: Pick $h: [n] \to [n]$ from a strongly universal hashing family
2: $z \leftarrow 0$
3: **for all** $1 \le i \le m$ **do**
4: Get item $a_i \in [n]$
5: **if** $\text{zeros}(h(a_i)) \ge z$ **then**
6: $z \leftarrow \text{zeros}(h(a_i))$
7: **return** $\sqrt{2} \cdot 2^z$

Analysis

Let

$$Y_r = \sum_{j: f_j > 0} \mathbb{1}_{\text{zeros}(h(j)) \ge r}, \text{ for } r \ge 0$$

Observations:

① $\boxed{Y_r \ge 1 \quad \text{iff} \quad z \ge r}$ (for all $r \ge 0$)

② $\mathbb{E}[Y_r] = \sum_{j: f_j > 0} \Pr[\text{zeros}(h(j)) \ge r] \underset{\substack{\uparrow \\ h(j) \text{ is uniformly distributed}}}{=} \sum_{j: f_j > 0} \frac{1}{2^r} \underset{\substack{\uparrow \\ d \text{ terms}}}{=} \frac{d}{2^r}$

③ $\text{Var}\, Y_r \underset{\substack{\uparrow \\ \text{pairwise independence}}}{=} \sum_{j: f_j > 0} \text{Var}\, \mathbb{1}_{\text{zeros}(h(j)) \ge r}$

$$\le \sum_{j: f_j > 0} \frac{1}{2^r} = \frac{d}{2^r}$$

Third example: Distinct Elements, the Tidemark (AMS) algorithm (4/5)

1: Pick $h: [n] \to [n]$ from a strongly universal hashing family
2: $z \leftarrow 0$
3: **for all** $1 \le i \le m$ **do**
4: Get item $a_i \in [n]$
5: **if** zeros$(h(a_i)) \ge z$ **then**
6: $z \leftarrow$ zeros$(h(a_i))$
7: **return** $\sqrt{2} \cdot 2^z$

So $\mathbb{E}[Y_r] \le \frac{d}{2^r}$, $\mathrm{Var}[Y_r] \le \frac{2^r}{d}$ for all $r \ge 0$

④ Markov!

$$\Pr[z \ge r] = \Pr[Y_r \ge 1] \le \mathbb{E}[Y_r] = \frac{d}{2^r} \quad (*)$$

Chebyshev!

$$\Pr[z \le r] = \Pr[Y_{r+1} = 0] \le \frac{2^{r+1}}{d} \quad (\circledast)$$

⑤ Conclude:

$C = 3\sqrt{2}$

$$\Pr[\hat{d} \ge Cd] = \Pr\left[2^z \ge \frac{C}{\sqrt{2}} d\right] \overset{(*)}{\le} \frac{\sqrt{2}}{Cd} \cdot d = \frac{1}{3}$$

$$\Pr[\hat{d} \le d/C] = \Pr\left[2^z \le \frac{d}{\sqrt{2}C}\right] \underset{(\circledast)}{\le} \frac{2d}{\sqrt{2}C} \cdot \frac{1}{d} = \frac{1}{3}$$

⑥ Amplify (?) Not a very good guarantee! Union bound naively gives $\Pr[\hat{d} \notin [d/C, Cd]] \le \underbrace{2/3}_{> 1/2} \ldots$

How to amplify this?

"Carefully" median trick still applies, and works.

Third example: Distinct Elements, the Tidemark (AMS) algorithm (5/5)

Theorem 42. *The (median trick version of the)* TIDEMARK *(AMS) algorithm is a* randomised *one-pass algorithm which, for any given parameter* $\delta \in (0,1]$, *provides an estimate* $\widehat{d}$ *of the number* d *of distinct elements of the stream such that, for some absolute constant* $C > 0$,

$$\Pr\left[\frac{1}{C} \cdot d \leq \widehat{d} \leq C \cdot d\right] \geq 1 - \delta$$

with space complexity

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

Only issue: $C = \Theta(1)$.
We don't get $1 \pm \varepsilon$
for arbitrary $\varepsilon > 0$.

Can we do better?

Yes.

Third example: Distinct Elements, the BJKST algorithm (1/4)

```
Input: Parameter ε ∈ (0,1]
  Set k ← O(log² n/ε⁴), T ← Θ(1/ε²)
  Pick h: [n] → [n] from a strongly universal hashing family
  Pick g: [n] → [k] from a strongly universal hashing family

  z ← 0, B ← ∅
  for all 1 ≤ i ≤ m do
      Get item a_i ∈ [n]
      if zeros(h(a_i)) ≥ z then
          B ← B ∪ {(g(a_i), zeros(h(a_i)))}
          while |B| ≥ T do
              z ← z + 1
              Remove every (a,b) with b < z from B
  return |B| · 2^z
```

Why this value of k?

Birthday paradox. We want no collision in any of the $\log n$ buckets (with high proba).

$k = O((T \log n)^2)$ means each bucket has no collision under g with proba $\geq 1 - \frac{1}{10 \log n}$

+ union bound to get overall proba $1 - \frac{1}{10} = 9/10$

Third example: Distinct Elements, the BJKST algorithm (2/4)

Space:

- Hash functions: h, g take space $O(\log n + \log k)$ $= O(\log n + \log \frac{1}{\varepsilon})$
- z: space $O(\log\log n)$
- B: space $T \cdot O(\log k + \log\log n) = O\left(\frac{1}{\varepsilon^2}(\log\log n + \log \frac{1}{\varepsilon})\right)$

Total: $O\left(\log n + \frac{\log(1/\varepsilon) + \log\log n}{\varepsilon^2}\right)$

Input: Parameter $\varepsilon \in (0,1]$
```
1: Set k ← O(log² n/ε⁴), T ← Θ(1/ε²)
2: Pick h: [n] → [n] from a strongly universal hashing family
3: Pick g: [n] → [k] from a strongly universal hashing family

4: z ← 0, B ← ∅
5: for all 1 ≤ i ≤ m do
6:     Get item a_i ∈ [n]
7:     if zeros(h(a_i)) ≥ z then
8:         B ← B ∪ {(g(a_i), zeros(h(a_i)))}
9:         while |B| ≥ T do
10:            z ← z + 1
11:            Remove every (a, b) with b < z from B
12: return |B| · 2^z
```

Assumption: no collisions via hashing. Our setting of k ensures this is true with high proba. ($\geq 9/10$), so we can just add $1/10$ of failure proba at the end (union bound) to account for it.

Analysis:

As before, $\mathbb{E}[Y_r] = \frac{d}{2^r}$, $\mathrm{Var}[Y_r] \leq \frac{2^r}{d}$ for all $r \geq 0$.

We return $Y_z 2^z$ on line 12, so we fail when $|Y_z 2^z - d| > \varepsilon d$.

Third example: Distinct Elements, the BJKST algorithm (3/4)

$$\Pr[\text{fail}] = \Pr\left[\left|Y_Z 2^Z - d\right| > \varepsilon d\right] = \sum_{r=1}^{\log n} \Pr\left[\left|Y_r - \frac{d}{2^r}\right| > \varepsilon \frac{d}{2^r},\ Z = r\right]$$

$$\leq \sum_{r=1}^{\log n} \min\left(\Pr\left[\left|Y_r - \frac{d}{2^r}\right| > \varepsilon \frac{d}{2^r}\right], \Pr[Z = r]\right)$$

$$\leq \sum_{r=1}^{s-1} \Pr\left[\left|Y_r - \frac{d}{2^r}\right| > \varepsilon \frac{d}{2^r}\right] + \underbrace{\sum_{r=s}^{\log n} \Pr[Z = r]}_{= \Pr[Z \geq s] = \Pr[Y_{s-1} \geq T]}$$

s is a param. to choose

← If we had the s-th bucket in play, the $(s-1)^{th}$ must have reached capacity before

$$\leq \sum_{r=1}^{s-1} \frac{2^r}{\varepsilon^2 d} + \frac{d}{2^{s-1} T}$$

Chebyshev

Markov!

$$\boxed{\mathbb{E}[Y_r] = \frac{d}{2^r} \qquad \mathrm{Var}[Y_r] \leq \frac{2^r}{d}}$$

$$\leq \frac{2^s}{\varepsilon^2 d} + \frac{\varepsilon^2 d}{100 \cdot 2^s}$$

setting of T

Balance the two terms: $2^s \approx \varepsilon^2 d$

$$\leq \frac{1}{10}$$

and we are done!

Input: Parameter $\varepsilon \in (0,1]$

Set $k \leftarrow O(\log^2 n/\varepsilon^4)$, $T \leftarrow \Theta(1/\varepsilon^2)$
Pick $h\colon [n] \to [n]$ from a strongly universal hashing family
Pick $g\colon [n] \to [k]$ from a strongly universal hashing family

$z \leftarrow 0$, $B \leftarrow \emptyset$
for all $1 \leq i \leq m$ **do**
  Get item $a_i \in [n]$
  if $\mathrm{zeros}(h(a_i)) \geq z$ **then**
    $B \leftarrow B \cup \{(g(a_i), \mathrm{zeros}(h(a_i)))\}$
    while $|B| \geq T$ **do**
      $z \leftarrow z + 1$
      Remove every (a, b) with $b < z$ from B
return $|B| \cdot 2^z$

(★) Key idea: use one bound (+ Chebyshev) for some terms, the other (+ Markov) for the remaining ones.

Third example: Distinct Elements, the BJKST algorithm (4/4)

Theorem 43. *The (median trick version of the)* BJKST *algorithm is a* randomised *one-pass algorithm which, for any given parameters* $\varepsilon, \delta \in (0, 1]$, *provides an estimate* $\widehat{d}$ *of the number* d *of distinct elements of the stream such that, for some absolute constant* $C > 0$,

$$\Pr\left[(1-\varepsilon) \cdot d \leq \widehat{d} \leq (1+\varepsilon) d \right] \geq 1 - \delta$$

with space complexity

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

… Can we do better?

Yes.
(a little bit)

(But it's a much
more complicated
algorithm/analysis)