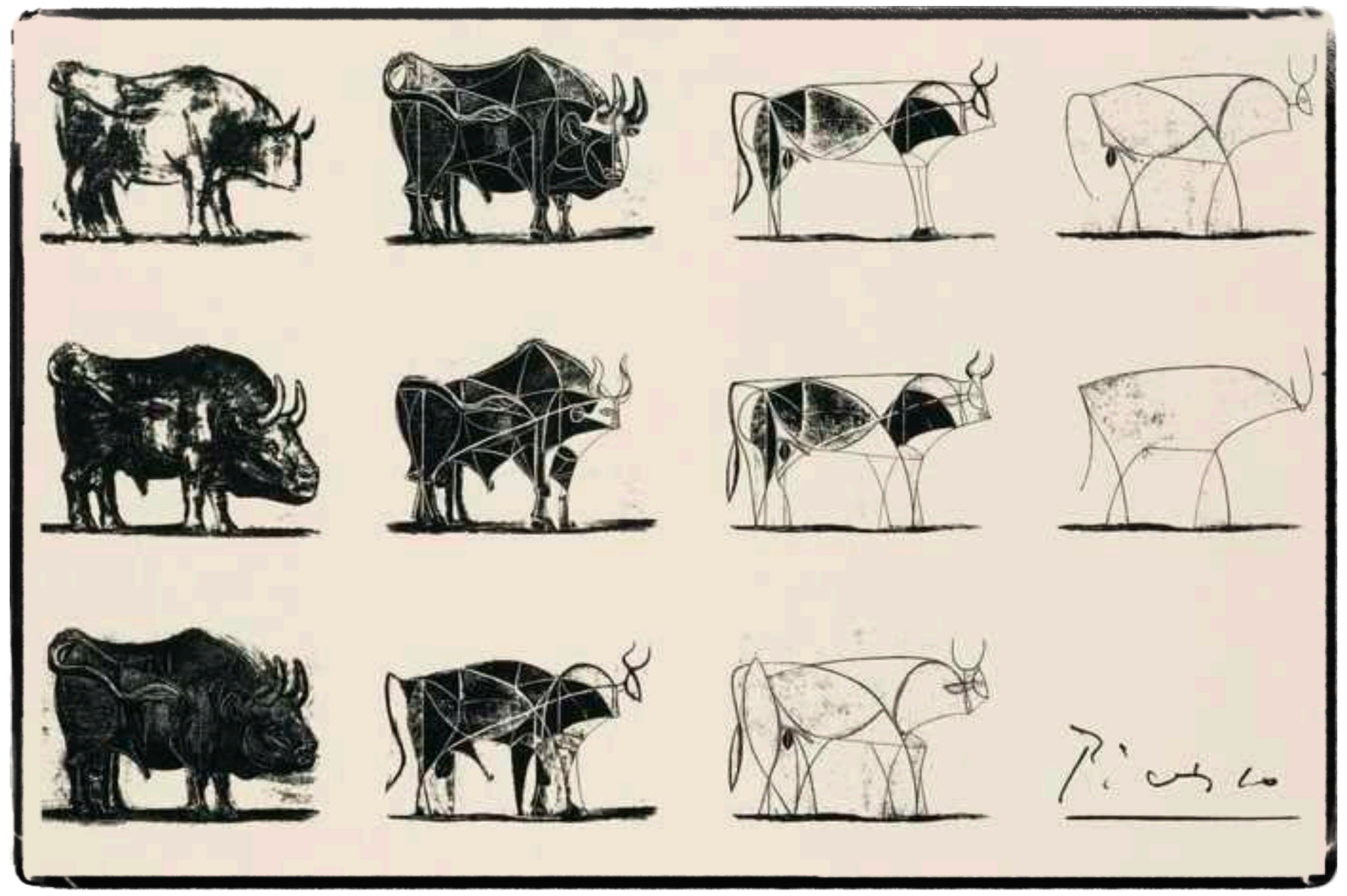
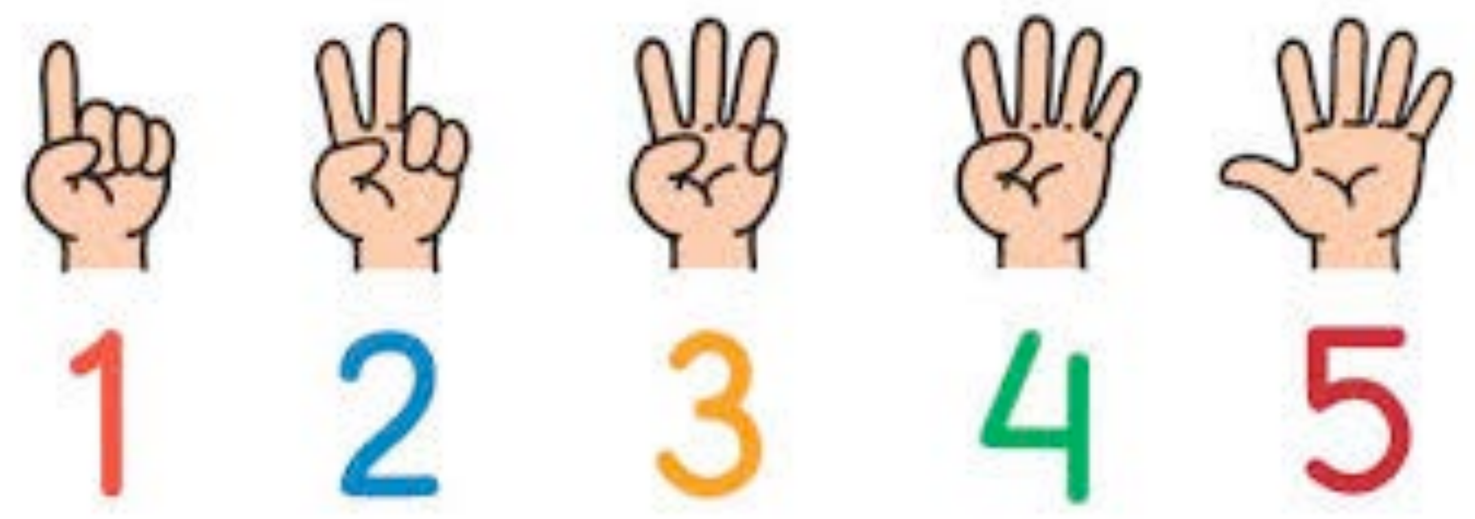


Advanced Algorithms

Sketching

刘明谋 Nanjing University, Suzhou, 2025

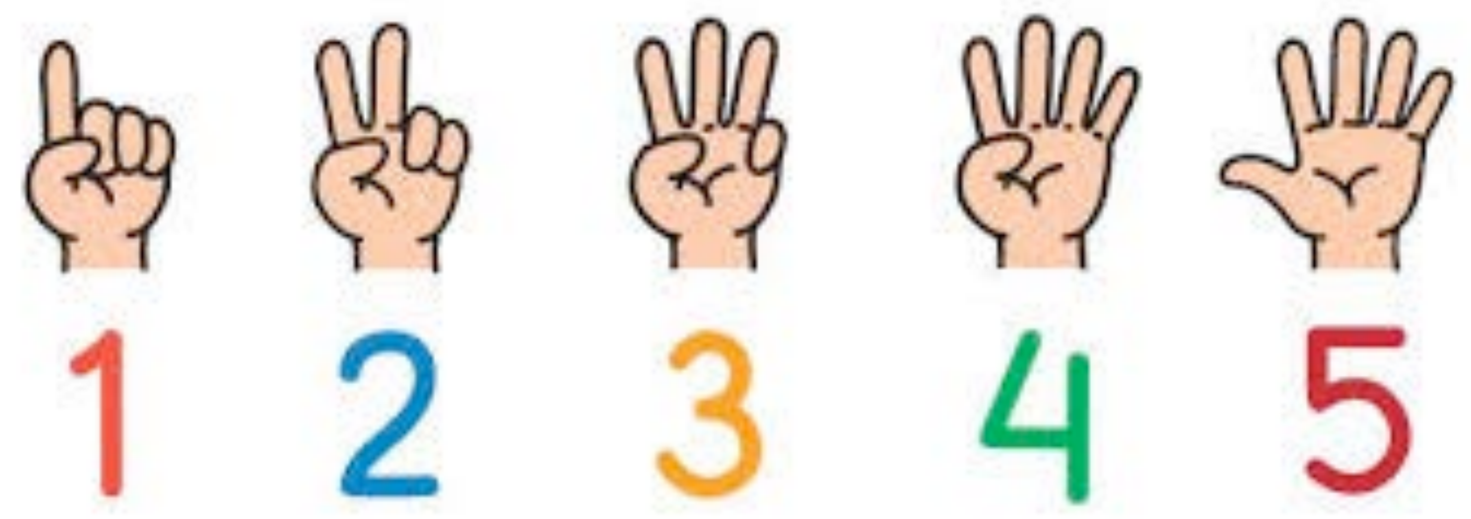
Counting



Counting

- **Maintain a counter n under updating:**
  - **init:** $n \leftarrow 0$
  - **increment:** $n \leftarrow n + 1$
  - **query: return** n

- Goal: use as less space as possible
- Naive solution: encode with $O(\log n)$ bits



Approximate Counting

- **Maintain a counter n under updating:**
  - **init:** $n \leftarrow 0$
  - **increment:** $n \leftarrow n + 1$
  - **query: return** n

- Goal: use as less space as possible
- Naive solution: encode with $O(\log n)$ bits
- "$n = 11451419$ is of length 8", encode with $\log_2 8 = 3$ bits
  - Approximation ratio: 10; Space cost: $O(\log \log n)$

Approximate Counting

- **Maintain a counter n under updating:**
  - **init:** $n \leftarrow 0$
  - **increment:** $n \leftarrow n + 1$
  - **query: return** n

- Goal: use as less space as possible
- "$n = 11451419$ is of length 8", encode with $\log_2 8 = 3$ bits

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}
do nothing w.p. $1 - 2^{-X}$
query: return $2^X - 1$

Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}

do nothing w.p. $1 - 2^{-X}$

query: return $2^X - 1$

- How correct is it? Unbiased: $\mathbb{E}[2^X] = n + 1$ after n updates
- Proof by induction. Base: $X_0 = 0, \mathbb{E}[2^{X_0}] = 1$. I.H. $\mathbb{E}[2^{X_n}] = n + 1$

$$\begin{aligned}
\mathbb{E}[2^{X_n+1}] &= \Sigma_j \Pr[X_n = j] \cdot \mathbb{E}\left[2^{X_{n+1}} \,|\, X_n = j\right] \\
&= \Sigma_j \Pr[X_n = j] \cdot \left((1 - 1/2^j)2^j + (1/2^j)2^{j+1}\right) \\
&= \Sigma_j \Pr[X_n = j]2^j + \Sigma_j \Pr[X_n = j] \\
&= \mathbb{E}\left[2^{X_n}\right] + 1 = n + 2
\end{aligned}$$

Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}
do nothing w.p. $1 - 2^{-X}$
query: return $2^X - 1$

- How space-efficient is it?
- Space cost $\log X$ bits; $\mathbb{E}[2^X] = n + 1$
- Jensen's inequality: $2^{\mathbb{E}[X]} \leq \mathbb{E}[2^X] = n + 1$;

Jensen's Inequality

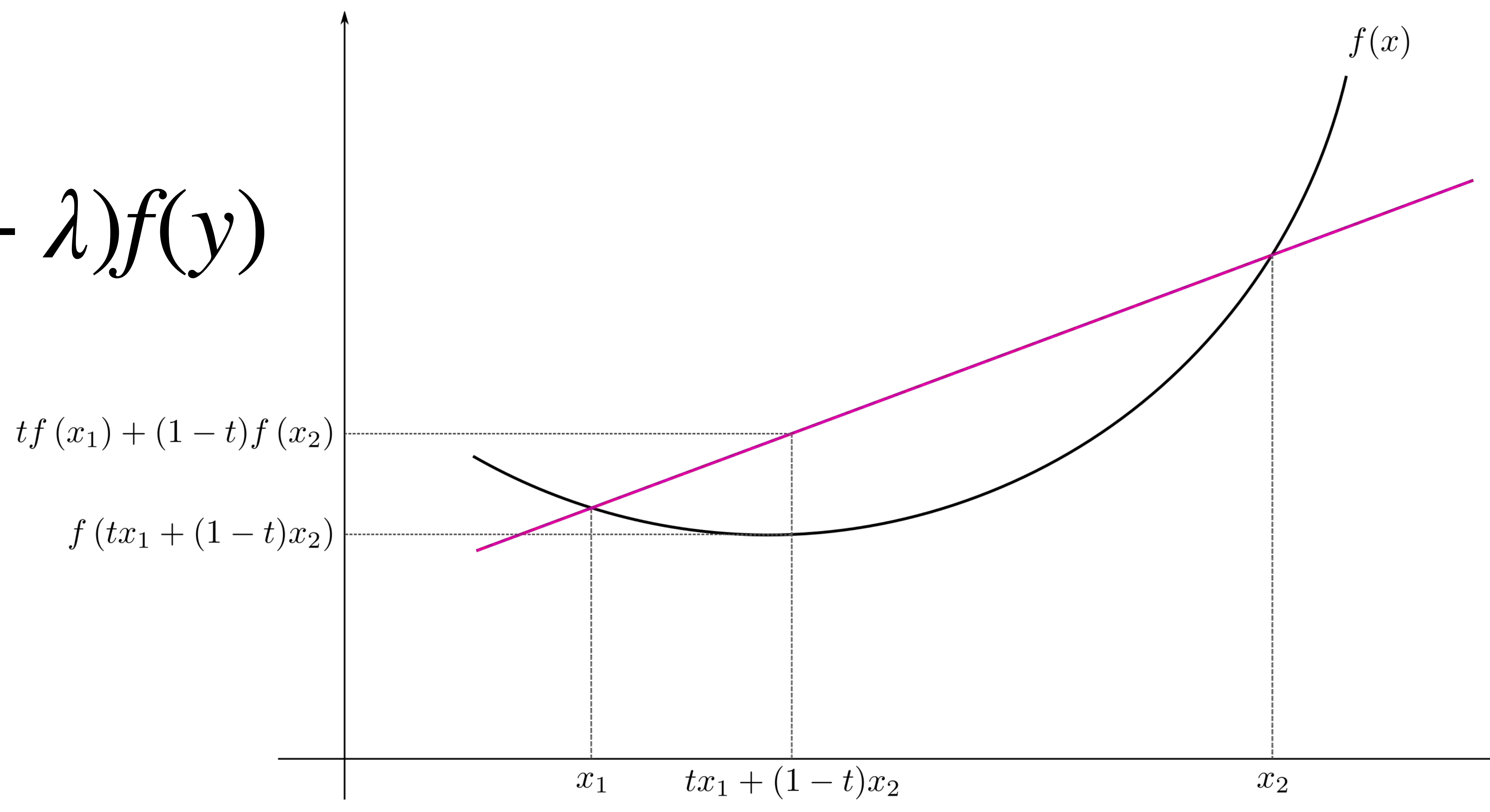
- For general (non-linear) function $f(X)$ of random variable X

  we don't have $\mathbb{E}[f(X)] = f(\mathbb{E}[X])$

- But if the convexity of f is known, then the **Jensen's inequality** applies:
  - f is **convex** $\iff f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$

    $\implies \mathbb{E}[f(X)] \geq f(\mathbb{E}[X])$
  - f is **concave** $\iff f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda)f(y)$

    $\implies \mathbb{E}[f(X)] \leq f(\mathbb{E}[X])$



Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}
do nothing w.p. $1 - 2^{-X}$
query: return $2^X - 1$

- How space-efficient is it?
- Space cost $\log X$ bits; $\mathbb{E}[2^X] = n + 1$
- Jensen's inequality: $2^{\mathbb{E}[X]} \leq \mathbb{E}[2^X] = n + 1$;
  $\mathbb{E}[\log X] \leq \log \mathbb{E}[X] \leq \log \log(n + 1)$
- Claim: $\log X \leq \log \log n + O(1)$

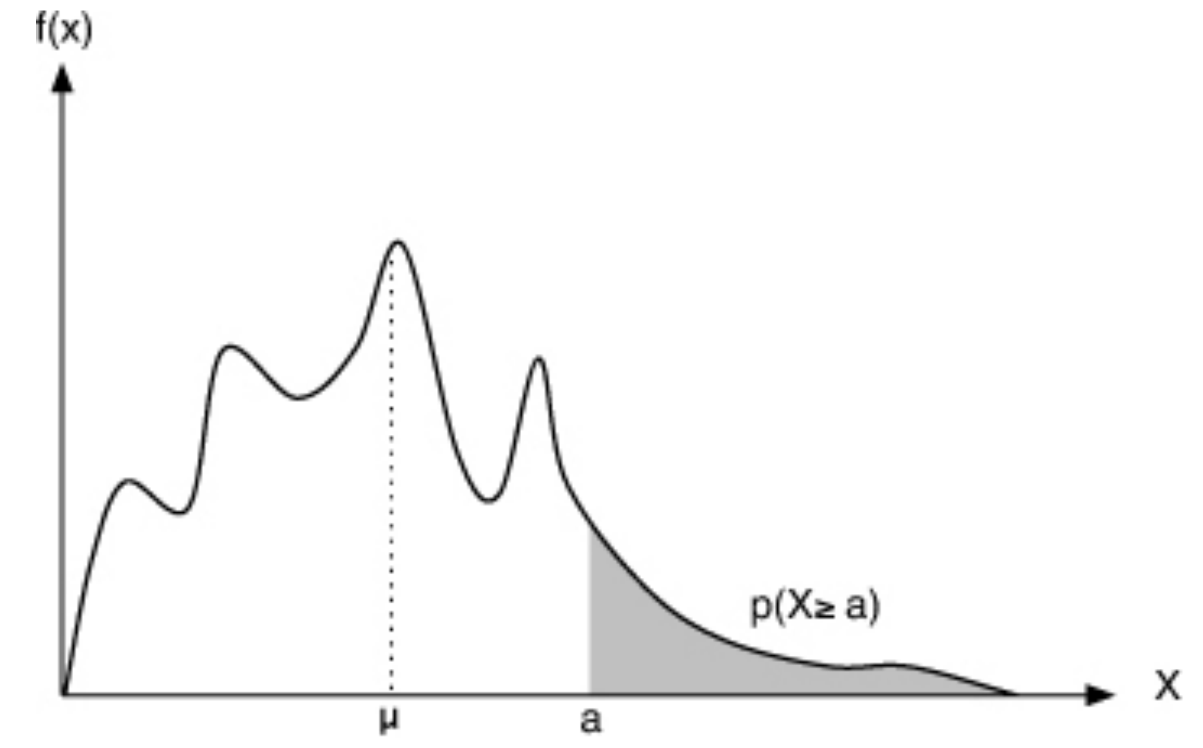
Markov's Inequality

For *nonnegative* random variable X, for any $t > 0$,

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}$$

Markov's Inequality

(马尔可夫不等式, the first Chebyshev inequality**)**



- Markov's inequality: Let X be a *nonnegative-valued* random variable. Then,

$$\text{for any } a > 0, \qquad \Pr(X \geq a) \leq \frac{\mathbb{E}[X]}{a}$$

- **Proof** (by total expectation):

($X \geq a$ is possible) (X is nonnegative)

$$\mathbb{E}[X] = \mathbb{E}[X \mid X \geq a] \cdot \Pr(X \geq a) + \mathbb{E}[X \mid X < a] \cdot \Pr(X < a)$$

$$\geq a \cdot \Pr(X \geq a) + 0 \cdot \Pr(X < a) \qquad = a \cdot \Pr(X \geq a)$$

$$\Longrightarrow \quad \Pr(X \geq a) \leq \frac{\mathbb{E}[X]}{a}$$

Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}
do nothing w.p. $1 - 2^{-X}$
query: return $\hat{n} = 2^X - 1$

Markov's Inequality

For *nonnegative* random variable X, for any $t > 0$,

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}$$

- Space cost $\log X$ bits; $\mathbb{E}[2^X] = n + 1$
- Claim: $\log X \leq \log \log n + O(1)$ w.p. 90%
- Proof: $\Pr[\hat{n} \geq 10n] \leq 0.1$ by Markov's inequality

Now assume $\hat{n} = 2^X - 1 \leq 10n$.

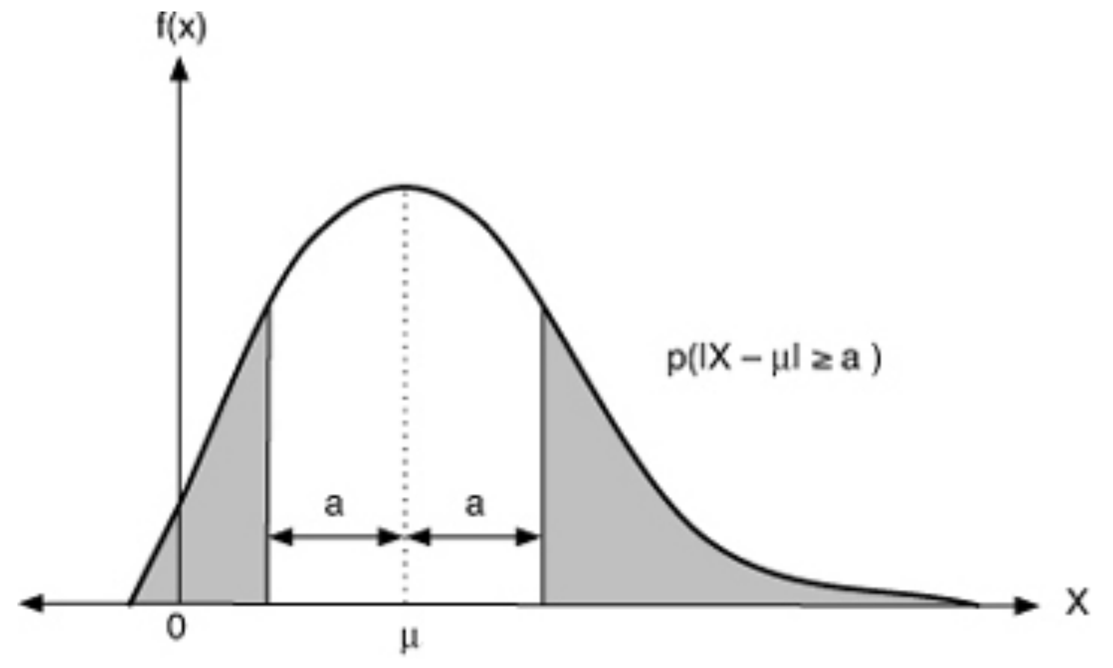
then $\log X = \log \log 2^X \leq \log \log (10n + 1) \leq \log \log n + O(1)$

Better analysis?

Higher moment bound!

Chebyshev's Inequality

(切比雪夫不等式, the second Chebyshev inequality)



- <u>Chebyshev's inequality</u>: Let X be a random variable. For any $a > 0$,

$$\Pr(|X - \mathbb{E}[X]| \geq a) \leq \frac{\mathbf{Var}[X]}{a^2} \quad := \mathbb{E}\left[(X - \mathbb{E}[X])^2\right]$$

- **Proof**: Apply Markov's inequality to $Y = (X - \mathbb{E}[X])^2$.

- **Corollary**: For standard deviation $\sigma = \sqrt{\mathbf{Var}[X]}$, for any $k \geq 1$,

$$\Pr(|X - \mathbb{E}[X]| \geq k\sigma) \leq \frac{1}{k^2}$$

Calculation of Variance

$$\mathbf{Var}[X] := \mathbb{E}\left[(X - \mathbb{E}[X])^2\right] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$$

- **Proof**: $\mathbf{Var}[X] = \mathbb{E}\left[(X - \mathbb{E}[X])^2\right]$

$$= \mathbb{E}\left[X^2 - 2\mathbb{E}[X]X + \mathbb{E}[X]^2\right]$$

$$= \mathbb{E}[X^2] - 2\mathbb{E}[X]\mathbb{E}[X] + \mathbb{E}[X]^2$$

$$= \mathbb{E}[X^2] - \mathbb{E}[X]^2$$

- X is constant ***a.s.*** $(\Pr(X = \mathbb{E}[X]) = 1) \iff \mathbb{E}[X^2] = \mathbb{E}[X]^2 \iff \mathbf{Var}[X] = 0$

Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}

do nothing w.p. $1 - 2^{-X}$

query: return $\hat{n} = 2^X - 1$

Chebyshev's Inequality

For random variable X, for any $t > 0$,

$$\Pr\left[\,|X - \mathbb{E}[X]| \geq t\,\right] \leq \frac{\mathbf{Var}[X]}{t^2}$$

- $\mathbf{Var}[\hat{n}] = \mathbb{E}[(2^{X_n} - 1)^2] - \mathbb{E}[2^{X_n} - 1]^2$

$$= \mathbb{E}[2^{2X_n} - 2 \cdot 2^{X_n} + 1] - n^2 = \mathbb{E}[2^{2X_n}] - n^2 - 2n - 1$$

- Claim: $\mathbb{E}[2^{2X_n}] \leq 3n(n+1)/2 + 1$
- Proof by induction. Base: $\mathbb{E}[2^{2X_0}] = 1$. I.H. $\mathbb{E}[2^{2X_n}] \leq 3n(n+1)/2 + 1$

- $\mathbb{E}\left[2^{2X_{n+1}}\right] = \mathbb{E}\left[2^{-X_n} \cdot 2^{2(X_n+1)} + (1 - 2^{-X_n}) \cdot 2^{2X_n}\right] = \mathbb{E}\left[2^{2X_n} + 3 \cdot 2^{X_n}\right]$ $= n + 1$

$$\leq 3n(n+1)/2 + 1 + 3(n+1) \leq 3(n+1)(n+2)/2 + 1$$

Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}

do nothing w.p. $1 - 2^{-X}$

query: return $\hat{n} = 2^X - 1$

Chebyshev's Inequality

For random variable X, for any $t > 0$,

$$\Pr\left[\,|X - \mathbb{E}[X]| \geq t\,\right] \leq \frac{\mathbf{Var}[X]}{t^2}$$

- $\mathbf{Var}[\hat{n}] = \mathbb{E}[(2^{X_n} - 1)^2] - \mathbb{E}[2^{X_n} - 1]^2$

$$= \mathbb{E}[2^{2X_n} - 2 \cdot 2^{X_n} + 1] - n^2 = \mathbb{E}[2^{2X_n}] - n^2 - 2n - 1$$

$\leq n^2/2 - n/2$

- Claim: $\mathbb{E}[2^{2X_n}] \leq 3n(n+1)/2 + 1$
- $\Pr\left[\,|\hat{n} - n| \geq \epsilon n\right] \leq \dfrac{\mathbf{Var}[\hat{n}]}{\epsilon^2 n^2} \leq \dfrac{1}{2\epsilon^2}$
- Not meaningful since $1/2\epsilon^2 < 1/2 \iff \epsilon > 1$ **:(**

Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}

do nothing w.p. $1 - 2^{-X}$

query: return $\hat{n} = 2^X - 1$

Morris' algorithm +:

Maintain k independent copies of Morris' counters.

query: return $\hat{n} = \Sigma_i \hat{n}_i / k$

- Morris' counter+ is unbiased by linearity of expectation: $\mathbb{E}[\hat{n}] = k \cdot \mathbb{E}[\hat{n}_i]/k = n$
- Space cost: $\Pr\left[\sum_i \hat{n}_i \geq 10kn\right] \leq 0.1$, encode with $\leq k \log\log n + O(k)$ bits w.p. 0.9
- Goal: $\Pr[\,|\hat{n} - n| \geq \epsilon n] \leq \delta$
- $\mathbf{Var}[\hat{n}] = \mathbf{Var}[\hat{n}_1]/k$

- Intuition: higher $k \implies$ higher accuracy

Markov's Inequality

For *nonnegative* random variable X, for any $t > 0$,

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}$$

Variance of Linear Function

- For random variables X, Y and real number $a \in \mathbb{R}$:
  - $\mathbf{Var}[a] = 0$
  - $\mathbf{Var}[X + a] = \mathbf{Var}[X]$ (variance is a central moment)
  - $\mathbf{Var}[aX] = a^2\mathbf{Var}[X]$ (variance is quadratic)
  - $\mathbf{Var}[X + Y] = \mathbf{Var}[X] + \mathbf{Var}[Y] + \underbrace{2(\mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y])}_{=0 \text{ if } X, Y \text{ are ind.}}$
- **Proof**: All can be verified through $\mathbf{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$.

Covariance of Independent Variables

- If random variables X and Y are **independent**, then

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

- If random variables $X_1, X_2, \ldots, X_n$ are **mutually independent**, then

$$\mathbb{E}\left[\prod_{i=1}^{n} X_i\right] = \mathbb{E}\left[\prod_{i=1}^{n-1} X_i\right] \cdot \mathbb{E}[X_n] = \prod_{i=1}^{n} \mathbb{E}[X_i]$$

Proof: By change of variable (*LOTUS*)

$$\mathbb{E}[XY] = \sum_{x,y} xy \Pr(X = x \cap Y = y) = \sum_{x,y} xy \Pr(X = x) \Pr(Y = y)$$

$$= \left(\sum_{x} x \Pr(X = x)\right)\left(\sum_{y} y \Pr(Y = y)\right) = \mathbb{E}[X]\mathbb{E}[Y]$$

Approximate Counting

Morris' algorithm +:
Maintain k independent copies of Morris' counters.

query: return $\hat{n} = \Sigma_i \hat{n}_i / k$

Chebyshev's Inequality

For random variable X, for any $t > 0$,

$$\Pr\left[\,|X - \mathbb{E}[X]| \geq t\,\right] \leq \frac{\mathbf{Var}[X]}{t^2}$$

- Morris' counter+ is unbiased by linearity of expectation: $\mathbb{E}[\hat{n}] = k \cdot \mathbb{E}[\hat{n}_i]/k = n$
- Space cost: $\Pr\left[\sum_i \hat{n}_i \geq 10kn\right] \leq 0.1$, encode with $\leq k \log\log n + O(k)$ bits w.p. 0.9
- $\mathbf{Var}[\hat{n}] = \mathbf{Var}[\hat{n}_1]/k.$ $\Pr[\,|\hat{n} - n| \geq \epsilon n] \leq \dfrac{\mathbf{Var}[\hat{n}_1]}{k\epsilon^2 n^2} \leq \dfrac{n^2/2}{k\epsilon^2 n^2} = \dfrac{1}{2k\epsilon^2} = \delta$
- Set $k = O(1/\epsilon^2\delta)$. Overall space cost $O\left(\dfrac{\log\log n}{\epsilon^2\delta}\right)$

Better algo?

Approximate Counting

Morris' algorithm +:
Maintain k independent copies of Morris' counters.

query: return **mean** $\hat{n} = \Sigma_i \hat{n}_i / k$

Morris' algorithm ++:
Maintain ℓ independent copies of Morris' counter+s with failure $1/3$.

query: return **median** of ℓ counters

- Morris' counter++ correct as long as more than half counter+s succeed
- One Morris' counter+ fails w.p. $\leq 1/3$.
- $\Pr\left[\sum_i^{\ell} Y_i \leq \ell/2\right] \leq \Pr\left[\sum_i Y_i - \ell\mu \leq -\ell/6\right]$ **Chernoff-Hoeffding bound!**

Chernoff-Hoeffding Bound

Chernoff-Hoeffding Bound:

For $X = \sum_{i=1}^{n} X_i$, where $X_1, \ldots, X_n \in \{0,1\}$ are ***independent*** (or ***negatively associated***),

for any $t > 0$:

$$\Pr\left[X \geq \mathbb{E}[X] + t\right] \leq \exp\left(-\frac{2t^2}{n}\right)$$

$$\Pr\left[X \leq \mathbb{E}[X] - t\right] \leq \exp\left(-\frac{2t^2}{n}\right)$$

Proved in *Foundations of Data Science.*
Proof is deferred to later sessions.

Approximate Counting

Morris' algorithm +:
Maintain k independent copies of Morris' counters.

query: return **mean** $\hat{n} = \Sigma_i \hat{n}_i / k$

Morris' algorithm ++ :
Maintain ℓ independent copies of Morris' counter+s with failure $1/3$.

query: return **median** of ℓ counters

- Morris' counter++ correct as long as more than half counter+s succeed
- One Morris' counter+ fails w.p. $\leq 1/3$.
- $\Pr\left[\sum_i^{\ell} Y_i \leq \ell/2\right] \leq \Pr\left[\sum_i Y_i - \ell\mu \leq -\ell/6\right] \leq \exp\left(-\frac{2(\ell/6)^2}{\ell}\right) = \underbrace{\exp\left(-\frac{\ell}{18}\right)}_{=:\,\delta}$
- Set $\ell = \lceil 18 \ln(1/\delta) \rceil$.

Approximate Counting

Morris' algorithm +:
Maintain k independent copies of Morris' counters.

query: return **mean** $\hat{n} = \Sigma_i \hat{n}_i / k$

Morris' algorithm ++:
Maintain ℓ independent copies of Morris' counter+s with failure $1/3$.

query: return **median** of ℓ counters

- Set $\ell = \lceil 18 \ln(1/\delta) \rceil$.
- $k\ell = O(\log(1/\delta)/\epsilon^2)$ counters in total. Recall $k = O(1/\epsilon^2 \delta')$ where $\delta' = 1/3$.
- Union bound: any counter reaches $X \geq \log(k\ell n/\delta)$, it ever increments w.p. $\leq n2^{-X} = \delta/k\ell$
- Union bound: none counter reaches $X \geq \log(k\ell n/\delta)$ w.p. δ
- Overall space cost: $k\ell \log\log(k\ell n/\delta) = O\left(\frac{\log(1/\delta)}{\epsilon^2} \cdot \log\log\left(\frac{n}{\epsilon\delta}\right)\right)$ bits w.p. $1-\delta$

Better algo?

Approximate Counting

Morris' algorithm:

increment: $X \leftarrow X + 1$ w.p. 2^{-X}
do nothing w.p. $1 - 2^{-X}$

query: return $\hat{n} = 2^X - 1$

Morris' algorithm++ :

Maintain ℓ independent copies of Morris' counter+s with failure $1/3$.

query: return **median** of ℓ counters

- General Morris' counter: increment w.p. $1/(1+\alpha)^X$, return $\hat{n} = ((1+\alpha)^X - 1)/\alpha$
- Intuition: higher α, higher accuracy, higher space cost.
- Let $\alpha = \theta(\epsilon^2\delta)$
- Space cost $O(\log\log n + \log(1/\epsilon) + \log(1/\delta))$ (Flajolet 1985)
- Space cost $O(\log\log n + \log(1/\epsilon) + \log\log(1/\delta))$ (Nelson & Yu 2020)

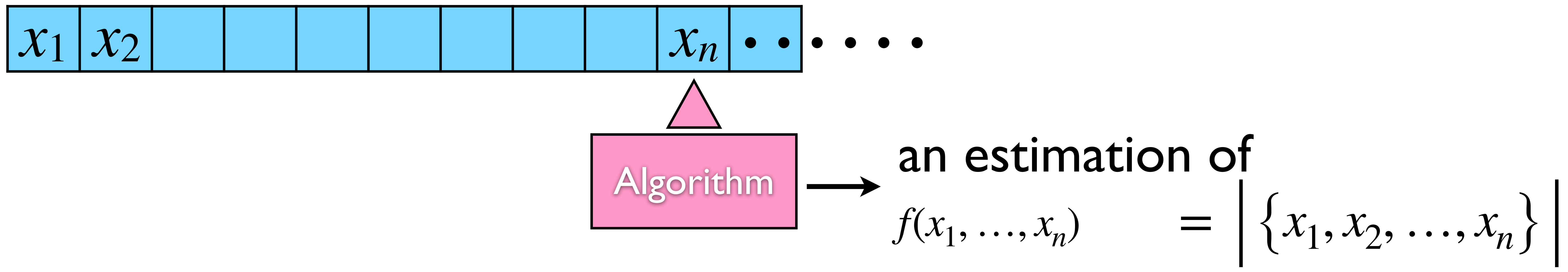
Distinct Elements
(0*th Frequency Moments*)

Count Distinct Elements

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left|\{x_1, x_2, \ldots, x_n\}\right|$

- Data stream **model**: input data item comes one at a time


x_1
x_2
x_n
Algorithm
an estimation of
$f(x_1, \ldots, x_n) = \left|\{x_1, x_2, \ldots, x_n\}\right|$

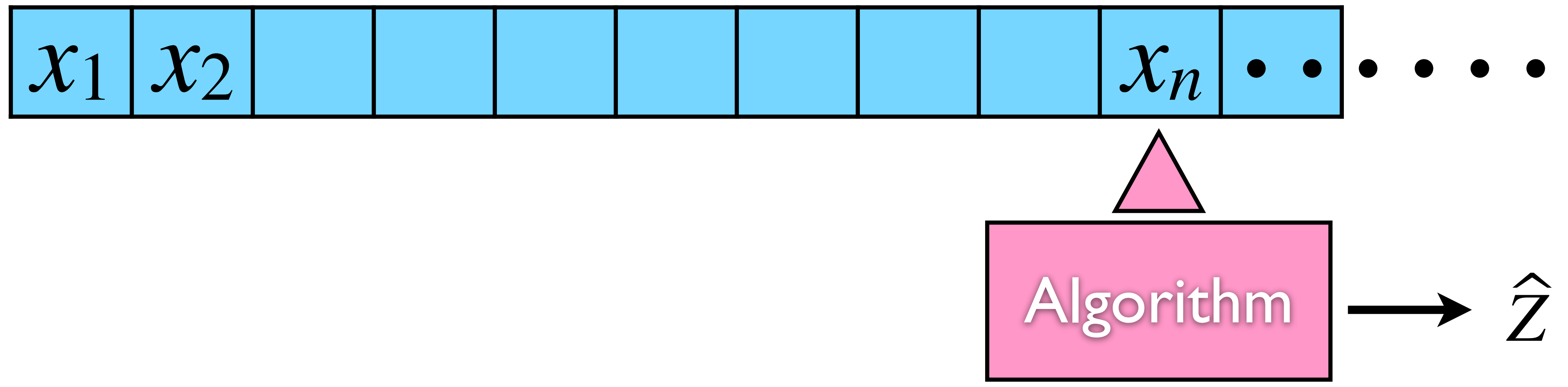
- Naïve algorithm: store all distinct data items using $\Omega(z \log N)$ bits
- **Sketch**: (lossy) representation of data using space $\ll z$
- **Lower bound** (Alon-Matias-Szegedy): any deterministic (exact or approx.) algorithm must use $\Omega(N)$ bits of space in the worst case

Count Distinct Elements

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- **Data stream model**: input data item comes one at a time



- **(ϵ, δ)-estimator**: randomized variable $\hat{Z}$

$$\Pr\left[(1-\epsilon)z \leq \hat{Z} \leq (1+\epsilon)z\right] \geq 1-\delta$$

Using only memory equivalent to 5 lines of printed text, you can estimate with a typical accuracy of 5% and in a single pass the total vocabulary of Shakespeare.

——Durand and Flajolet 2003

William Shakespeare

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

Simple Uniform Hash Assumption (SUHA):
A uniform function is available, whose preprocessing, representation and evaluation are considered to be easy.

- (*idealized*) uniform hash function $h : U \to [0,1]$
  - $x_i = x_j \longrightarrow$ the same hash value $h(x_i) = h(x_j) \in_r [0,1]$
- $\{h(x_1), \ldots, h(x_n)\}$: $z \times$ uniform and independent values in $[0,1]$
  - partition $[0,1]$ into $z+1$ subintervals (with *identically distributed* lengths)

$$\mathbb{E}\left[\min_{1 \le i \le n} h(x_i)\right] = \mathbb{E}[\text{length of a subinterval}] = \frac{1}{z+1} \quad \text{(by symmetry)}$$

- estimator: $\widehat{Z} = \dfrac{1}{\min_i h(x_i)} - 1$?   Variance is too large!

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- (*idealized*) uniform hash function $h : U \to [0,1]$

Min Sketch:

```
let $Y = \min_{1 \le i \le n} h(x_i)$;
return $\hat{Z} = \frac{1}{Y} - 1$;
```

- By symmetry:

$$\mathbb{E}[Y] = \frac{1}{z+1}$$

- Goal:

$$\Pr\left[\boxed{\hat{Z} < (1-\epsilon)z \text{ or } \hat{Z} > (1+\epsilon)z}\right] \le \delta$$

$\Downarrow$ assuming $\epsilon \le 1/2$

$$\left| Y - \mathbb{E}[Y] \right| > \frac{\epsilon/2}{z+1} \iff \left| Y - \frac{1}{z+1} \right| > \frac{\epsilon/2}{z+1}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left|\{x_1, x_2, \ldots, x_n\}\right|$

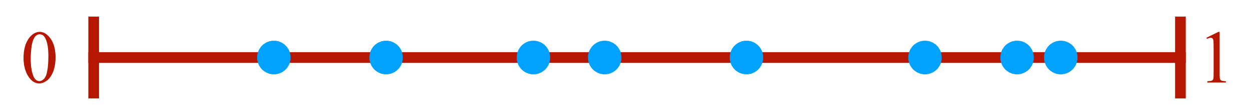
- (*idealized*) uniform hash function $h : U \to [0,1]$

Min Sketch:

```
let $Y = \min_{1 \le i \le n} h(x_i)$;
return $\widehat{Z} = \frac{1}{Y} - 1$;
```

- Uniform independent hash values:

$$H_1, \ldots, H_z \in [0,1]$$



- $Y = \min_{1 \le i \le z} H_i$

geometric probability: $\Pr[Y > y] = (1-y)^z \quad \Rightarrow \quad$ pdf: $p(y) = z(1-y)^{z-1}$

$$\mathbb{E}[Y^2] = \int_0^1 y^2 p(y)\,\mathrm{d}y = \int_0^1 y^2 z(1-y)^{z-1}\,\mathrm{d}y \quad = \frac{2}{(z+1)(z+2)}$$

$$\mathbf{Var}[Y] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2 \ = \frac{z}{(z+1)^2(z+2)} \le \frac{1}{(z+1)^2}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- (*idealized*) uniform hash function $h : U \to [0,1]$

Chebyshev's Inequality

For random variable X, for any $t > 0$,

$$\Pr\left[\,|X - \mathbb{E}[X]| \geq t\,\right] \leq \frac{\mathbf{Var}[X]}{t^2}$$

Min Sketch:

let $Y = \min_{1 \leq i \leq n} h(x_i)$;

return $\widehat{Z} = \frac{1}{Y} - 1$;

- By symmetry:

$$\mathbb{E}[Y] = \frac{1}{z+1}$$

- Goal:

$$\Pr\left[\boxed{\widehat{Z} < (1-\epsilon)z \text{ or } \widehat{Z} > (1+\epsilon)z}\right] \leq \delta$$

assuming $\epsilon \leq 1/2$ $\Downarrow$

$$\mathbf{Var}[Y] \leq \frac{1}{(z+1)^2} \xrightarrow{\text{(Chebyshev)}} \Pr\left[\left|Y - \mathbb{E}[Y]\right| > \frac{\epsilon/2}{z+1}\right] \leq \frac{4}{\epsilon^2}$$

The Mean Trick (for Variance Reduction)

- Variance and covariance:

$$\mathbf{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

$$\mathbf{Cov}(X, Y) = \mathbb{E}\left[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])\right]$$

- Useful properties:

$$\mathbf{Var}[X + a] = \mathbf{Var}[X]$$

$$\mathbf{Var}[aX] = a^2\mathbf{Var}[X]$$

$$\mathbb{E}\left[\frac{1}{k}\sum_{i=1}^{k} X_i\right] = \mathbb{E}[X_1]$$

$$\mathbf{Var}\left[\sum_i X_i\right] = \sum_i \mathbf{Var}[X_i] + \sum_{i \neq j} \mathbf{Cov}(X_i, X_j)$$

- For pairwise independent identically distributed X_i's:

$$\mathbf{Var}\left[\frac{1}{k}\sum_{i=1}^{k} X_i\right] = \frac{1}{k^2}\sum_{i=1}^{k} \mathbf{Var}[X_i] = \frac{1}{k}\mathbf{Var}[X_1]$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left|\{x_1, x_2, \ldots, x_n\}\right|$

- uniform & independent hash functions $h_1, \ldots, h_k : U \to [0,1]$

Min Sketch+:

for each $1 \le j \le k$, let $Y_j = \min_{1 \le i \le n} h_j(x_i)$;

return $\widehat{Z} = \frac{1}{\overline{Y}} - 1$ where $\overline{Y} = \frac{1}{k}\sum_{j=1}^{k} Y_j$;

- For every $1 \le j \le k$:

$$\mathbb{E}\left[Y_j\right] = \frac{1}{z+1} \xrightarrow{\text{linearity of expectation}} \mathbb{E}\left[\overline{Y}\right] = \frac{1}{z+1}$$

$$\mathbf{Var}[Y_j] \le \frac{1}{(z+1)^2} \xrightarrow{\text{independence}} \mathbf{Var}\left[\overline{Y}\right] \le \frac{1}{k(z+1)^2}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- uniform & independent hash functions $h_1, \ldots, h_k : U \to [0,1]$

Min Sketch+:

for each $1 \le j \le k$, let $Y_j = \min_{1 \le i \le n} h_j(x_i)$;

return $\widehat{Z} = \frac{1}{\overline{Y}} - 1$ where $\overline{Y} = \frac{1}{k} \sum_{j=1}^{k} Y_j$;

$$\mathbb{E}\left[\overline{Y}\right] = \frac{1}{z+1}$$

$$\mathbf{Var}\left[\overline{Y}\right] \le \frac{1}{k(z+1)^2}$$

- Goal: $\Pr\left[\widehat{Z} < (1-\epsilon)z \text{ or } \widehat{Z} > (1+\epsilon)z\right] \le \delta$

assuming $\epsilon \le 1/2$

$$\Pr\left[\left|\overline{Y} - \mathbb{E}\left[\overline{Y}\right]\right| > \frac{\epsilon/2}{z+1}\right] \le \frac{4}{k\epsilon^2} \le \delta$$

(Chebyshev)

Set $k = \left\lceil \frac{4}{\epsilon^2 \delta} \right\rceil$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left|\{x_1, x_2, \ldots, x_n\}\right|$

- uniform & independent hash functions $h_1, \ldots, h_k : U \to [0,1]$

Min Sketch+: set $k = \lceil 4/(\epsilon^2\delta) \rceil$

for each $1 \le j \le k$, let $Y_j = \min_{1 \le i \le n} h_j(x_i)$;

return $\widehat{Z} = \frac{1}{\overline{Y}} - 1$ where $\overline{Y} = \frac{1}{k}\sum_{j=1}^{k} Y_j$;

$$\Pr\left[(1-\epsilon)z \le \widehat{Z} \le (1+\epsilon)z\right] \ge 1-\delta$$

- **Space cost:** $k = O\left(\frac{1}{\epsilon^2\delta}\right)$ ***real numbers*** in $[0,1]$
- Storing k ***idealized*** hash functions.

Two-Point Sampling (2-Universal Hashing)

- Let $p > 1$ be a prime number and $[p] = \{0,1,\ldots,p-1\} = \mathbb{Z}_p$.
- Pick $\boldsymbol{a}, \boldsymbol{b} \in [p]$ *u.a.r.* and let $r_i = (\boldsymbol{a} \cdot i + \boldsymbol{b}) \bmod p$ for $i = 1,2,\ldots,p$
    - $r_1, \ldots, r_p \in [p]$ are <u>pairwise independent</u>
    - each r_i is <u>uniformly distributed</u> over $[p]$

- Linear congruential hashing $f : \mathrm{GF}(q) \to \mathrm{GF}(q)$ over finite field $\mathrm{GF}(q)$:
    - Pick $\boldsymbol{a}, \boldsymbol{b} \in \mathrm{GF}(q)$ u.a.r and let $f(x) = \boldsymbol{a} \cdot x + \boldsymbol{b}$ for $x \in \mathrm{GF}(q)$
        - $\{x \in \mathrm{GF}(q)\}$ are <u>pairwise independent</u>
        - each $f(x)$ is <u>uniformly distributed</u> over $\mathrm{GF}(q)$
        - $\mathrm{GF}(2^w)$ exists for any positive integer $w \in \mathbb{Z}^+$

Flajolet-Martin Algorithm

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

```
let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;
return $\widehat{Z} = 2^R$;
```

$$\Pr\left[\widehat{Z} < \frac{z}{C} \text{ or } \widehat{Z} > C \cdot z \right] \le \frac{3}{C}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\text{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \text{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

Let

$$Y_r = \sum_{\text{distinct } x \in \{x_1, \ldots, x_n\}} I\left[\text{zeros}\left(h(x)\right) \ge r\right]$$

(linearity of expectation)

$$\mathbb{E}[Y_r] = \sum_{\text{distinct } x \in \{x_1, \ldots, x_n\}} \Pr\left[\text{zeros}\left(h(x)\right) \ge r\right] = z2^{-r}$$

(pairwise independence)

$$\textbf{Var}[Y_r] = \sum_{\text{distinct } x \in \{x_1, \ldots, x_n\}} \textbf{Var}\left[I[\text{zeros}\left(h(x)\right) \ge r]\right] = z2^{-r}(1 - 2^{-r}) \le z2^{-r}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left|\{x_1, x_2, \ldots, x_n\}\right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

Let

$$Y_r = \sum_{\text{distinct } x \in \{x_1, \ldots, x_n\}} I\left[\mathrm{zeros}\left(h(x)\right) \ge r\right]$$

$$\mathbb{E}[Y_r] = z2^{-r} \qquad \mathbf{Var}[Y_r] \le z2^{-r}$$

(denote $r^* = \lceil \log_2 Cz \rceil$) $\quad \Pr\left[\widehat{Z} > Cz\right] \le \Pr[R \ge r^*]$

(observe $R = \max\{r : Y_r > 0\}$) $\quad \le \Pr[Y_{r^*} > 0] = \Pr[Y_{r^*} \ge 1]$

(Markov's inequality) $\quad \le \mathbb{E}[Y_{r^*}] = z/2^{r^*} \le 1/C$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left|\{x_1, x_2, \ldots, x_n\}\right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\text{zeros}(y) = \max\{i : 2^i | y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \text{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

Let

$$Y_r = \sum_{\text{distinct } x \in \{x_1, \ldots, x_n\}} I\left[\text{zeros}\left(h(x)\right) \ge r\right]$$

$$\mathbb{E}[Y_r] = z2^{-r} \qquad \mathbf{Var}[Y_r] \le z2^{-r}$$

(denote $r^{**} = \lceil \log_2(z/C) \rceil$) $\qquad \Pr\left[\widehat{Z} < z/C\right] \le \Pr[R < r^{**}]$

(observe $R = \max\{r : Y_r > 0\}$) $\qquad \le \Pr[Y_{r^{**}} = 0]$

(Chebyshev's inequality) $\qquad \le \mathbf{Var}[Y_{r^{**}}]/\mathbb{E}[Y_{r^{**}}]^2 \le 2^{r^{**}}/z$

$$\le 2/C$$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

```
let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;
return $\widehat{Z} = 2^R$;
```

$$\Pr\left[\widehat{Z} < \frac{z}{C} \text{ or } \widehat{Z} > C \cdot z\right] \le \frac{3}{C}$$

- **Space cost:** $O(\log \log N)$ bits for maintaining R
- $O(\log N)$ bits for storing 2-wise independent hash function

Bottom-k Algorithm

Input: a sequence $x_1, x_2, \ldots, x_n \in [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [N] \to [M] = \{1, \ldots, M\}$

Bottom-k Algorithm:

```
let $Y_1, \ldots, Y_k$ be the $k$ smallest hash values among
        $\{ h(x_1), h(x_2) \ldots, h(x_n) \}$;
return $\widehat{Z} = \frac{kM}{Y_k}$;
```

(Bar-Yossef, Jayram, Kumar, Sivakumar and Trevisan, 2002)

Input: a sequence $x_1, x_2, \ldots, x_n \in [N]$

Output: an estimation of $z = \left| \left\{ x_1, x_2, \ldots, x_n \right\} \right|$

- 2-wise independent hash function $h : [N] \to [M] = \{1, \ldots, M\}$

Bottom-k Algorithm:

```
let $Y_1, \ldots, Y_k$ be the $k$ smallest hash values among
        $\{ h(x_1), h(x_2) \ldots, h(x_n) \}$;
return $\widehat{Z} = \frac{kM}{Y_k}$;
```

- Goal: $$\Pr\left[\boxed{\widehat{Z} < (1-\epsilon)z \text{ or } \widehat{Z} > (1+\epsilon)z} \right] \le \delta$$

assuming $\epsilon \le 1$

$$\Downarrow$$

$$\left| Y_k - \frac{kM}{z} \right| > \frac{\epsilon}{2} \cdot \frac{kM}{z}$$

- uniform and 2-wise independent $X_1, \ldots, X_n \in [N^3]$
- let $Y_1, \ldots, Y_z$ be these elements in non-decreasing order

Let $V = \sum_{i=1}^{z} I\left[X_i \leq \left(1 - \frac{\epsilon}{2}\right)\frac{kM}{z}\right]$ $\quad W = \sum_{i=1}^{z} I\left[X_i \leq \left(1 + \frac{\epsilon}{2}\right)\frac{kM}{z}\right]$

$$\mathbb{E}[V] = \left(1 - \frac{\epsilon}{2}\right)k \qquad \mathbb{E}[W] = \left(1 + \frac{\epsilon}{2}\right)k$$

$$Y_k < \left(1 - \frac{\epsilon}{2}\right)\frac{k(M+1)}{z} \Longrightarrow V \geq k \qquad Y_k > \left(1 + \frac{\epsilon}{2}\right)\frac{k(M+1)}{z} \Longrightarrow W \leq k$$

(Chebyshev's inequality)

$$\Pr[V \geq k] \leq \frac{8}{k\epsilon^2} \qquad \Pr[W \leq k] \leq \frac{8}{k\epsilon^2}$$

- Goal: $\Pr\left[\left|Y_k - \frac{kM}{z}\right| > \frac{\epsilon}{2} \cdot \frac{kM}{z}\right] \leq \delta$ $\quad$ Set $k = \left\lceil \frac{16}{\epsilon^2 \delta} \right\rceil$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [N] \to [N^3]$

Bottom-k Algorithm: Set $k = \lceil 16/(\epsilon^2 \delta) \rceil$

let $Y_1, \ldots, Y_k$ be the k smallest hash values among

$\{ h(x_1), h(x_2) \ldots, h(x_n) \}$;

return $\widehat{Z} = \dfrac{kM}{Y_k}$;

$$\Pr\left[(1-\epsilon)z \le \widehat{Z} \le (1+\epsilon)z \right] \ge 1 - \delta$$

- **Space cost:** $O(k \log N) = O(\epsilon^{-2} \log N)$ bits when $\delta = \Omega(1)$

Counting in Practice

HyperLogLog: $M[m]$ (initialized to $-\infty$)

Upon each x_i: $M[h(x_i)] = \max\{M[h(x_i)], \rho(x_i)\}$

Query: compute $Z \triangleq 1 \Big/ \left(\sum_{j=1}^{m} 2^{-M[j]} \right)$

return $\alpha_m m^2 Z$, where

$$\alpha_m = \left(m \int_0^\infty \left(\log_2 \left(\frac{2+u}{1+u} \right) \right)^m du \right)^{-1}$$

- Easy to implement, efficient in computing, hard to analyze
- Fast merging: $M_{all}[j] = \max\{M_1[j], M_2[j]\}$

Frequency Moments

- **Data stream:** $x_1, x_2, \ldots, x_n \in U$
- for each $x \in U$, define *frequency* of x as $f_x = |\{i : x_i = x\}|$

k-th *frequency moments*: $F_k = \sum_{x \in U} f_x^k$

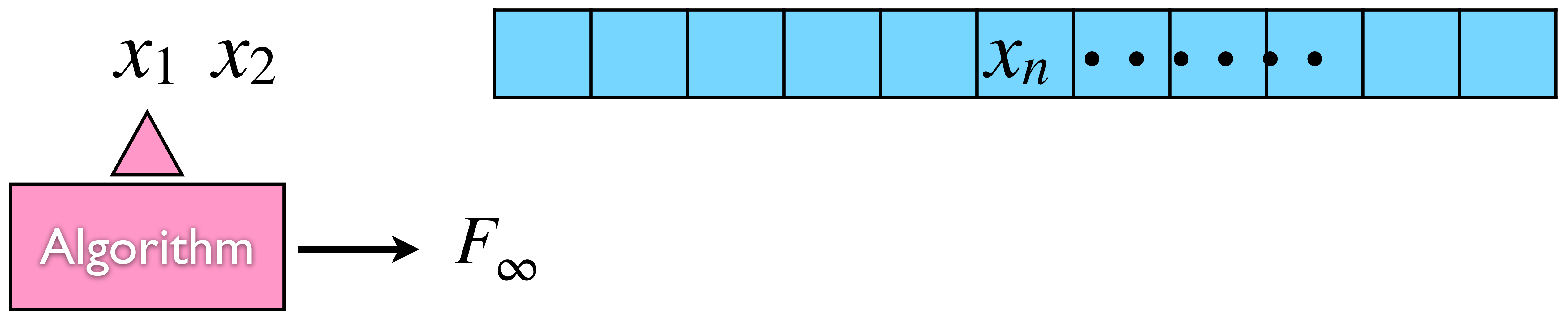
- **Space complexity** for (ϵ, δ)-estimation: constant ϵ, δ
  - for $k \le 2$: $\mathrm{polylog}(N)$ **[Alon-Matias-Szegedy '96]**
  - for $k > 2$: $\tilde{\Theta}(N^{1-2/k})$ **[Indyk-Woodruff '05]**
- **Count distinct elements:** F_0
  - optimal algorithm **[Kane-Nelson-Woodruff '10]**: $O(\epsilon^{-2} + \log N)$ bits

Heavy Hitter & Point Query (Frequency Estimation)

Frequency Estimation

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$
Query: Find the most frequent item

- <span>Data stream</span> **model**: input data item comes one at a time



- Cannot store the whole database with less space.
- Cannot apply even 2-approximation, even with randomness.
- Heavy hitters: items that appears $> \epsilon n$ times

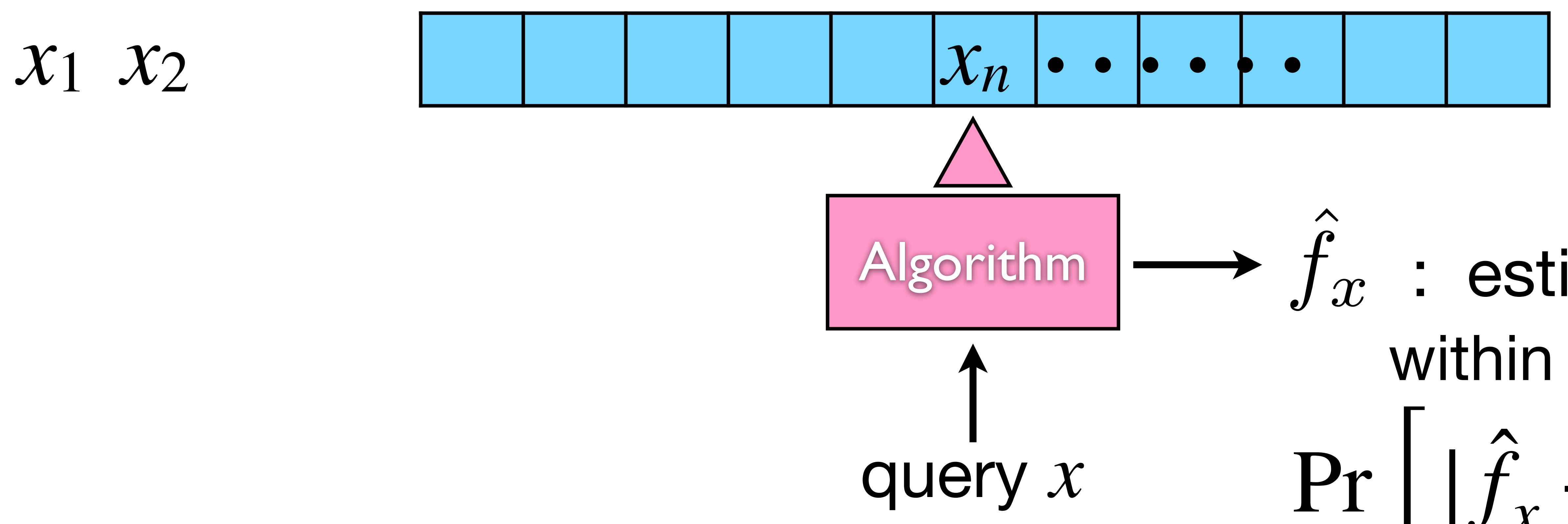
Frequency Estimation

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Point Query: an item $x \in U$

Estimate the *frequency* $f_x = |\{i : x_i = x\}|$ of x.

- **Data stream model**: input data item comes one at a time



$\hat{f}_x$: estimation of f_x within additive error

$$\Pr\left[\,|\hat{f}_x - f_x| \geq \epsilon n\right] \leq \delta$$

- Heavy hitters: items that appears $> \epsilon n$ times

Point Query

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Point Query: an item $x \in U$

Estimate the ***frequency*** $f_x = |\{i : x_i = x\}|$ of x.

- hash function $h : [N] \to [m]$

Bucket Sketch: $\mathrm{S}[m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{S}[h(x_i)] + +$;

Query x: return $\hat{f}_x = \mathrm{S}[h(x)]$

Observation: for all $x, f_x \leq \hat{f}_x$

$\leq ?$

Data: sequence $x_1, \ldots, x_n \in [N]$ **Query**: $x \in [N]$

frequency $f_x = |\{i : x_i = x\}|$ of x

Bucket Sketch: $\mathrm{S}[m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{S}[h(x_i)]++$;

Query x: return $\hat{f}_x = \mathrm{S}[h(x)]$

- $\hat{f}_x$ is good as long as m is large
  - Intuition: Scattered by h according to m
- **Claim:** $\mathbb{E}[\hat{f}_x] \leq f_x + n/m$.
- Proof: $\mathbb{E}[\hat{f}_x] = \mathbb{E}\left[\sum_y I(h(x) = h(y)) \cdot f_y\right] = \mathbb{E}\left[f_x + \sum_{y \neq x} I(h(x) = h(y)) \cdot f_y\right]$

$$= f_x + \sum_{y \neq x} f_y \cdot \underbrace{\Pr[h(x) = h(y)]}_{1/m \text{ for 2-universal } h} = f_x + (n - f_x)/m \leq f_x + n/m$$

Data: sequence $x_1, \ldots, x_n \in [N]$ **Query**: $x \in [N]$

frequency $f_x = |\{i : x_i = x\}|$ of x

Bucket Sketch: $\mathrm{S}[m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{S}[h(x_i)] + +$;

Query x: return $\hat{f}_x = \mathrm{S}[h(x)]$

Claim: $\mathbb{E}[\hat{f}_x] \le f_x + n/m$.

(Markov's inequality) $\Pr\left[\hat{f}_x - f_x \ge \epsilon n\right] \le \frac{1}{\epsilon m} \le \delta$

- $\hat{f}_x$ is good as long as m is large ($m \ge 1/\epsilon\delta$)

Improvement?

- What happens when $m \ll 1/\epsilon\delta$?
  - collision & false positivity
- Repeat to correct

Count-Min Sketch

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Point Query: an item $x \in U$

Estimate the ***frequency*** $f_x = |\{i : x_i = x\}|$ of x.

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)] + +$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min\limits_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

Intuition: $\mathrm{CMS}[j][h_j(x)] \ge f_x$, the **smaller**, the **better**!

Data: sequence $x_1, \ldots, x_n \in [N]$ **Query**: $x \in [N]$

frequency $f_x = |\{i : x_i = x\}|$ of x

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)] ++$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

$\forall x, \forall j$: $\mathrm{CMS}[j][h_j(x)] \ge f_x$

$$\mathbb{E}\left[\mathrm{CMS}[j][h_j(x)]\right] \le f_x + \frac{n}{m}$$

(Markov's inequality) $\Pr\left[\mathrm{CMS}[j][h_j(x)] - f_x \ge \epsilon n\right] \le \frac{1}{\epsilon m}$

$$\Pr\left[|\hat{f}_x - f_x| \ge \epsilon n\right] = \Pr\left[\forall 1 \le j \le k : \mathrm{CMS}[j][h_j(x)] - f_x \ge \epsilon n\right] \le \left(\frac{1}{\epsilon m}\right)^k$$

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Point Query: an item $x \in U$

Estimate the *frequency* $f_x = |\{i : x_i = x\}|$ of x.

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)]++$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

$$\Pr\left[|\hat{f}_x - f_x| \ge \epsilon n\right] \le \left(\frac{1}{\epsilon m}\right)^k \le \delta$$

- choose $m = \lceil \mathrm{e}/\epsilon \rceil$ and $k = \lceil \ln(1/\delta) \rceil$
  - **space cost:** $O\left(\frac{1}{\epsilon}\log(1/\delta)\log n\right)$ bits
  - $O(\log(1/\delta)\log N)$ bits for hash functions
  - **time cost for query:** $O(\log(1/\delta))$

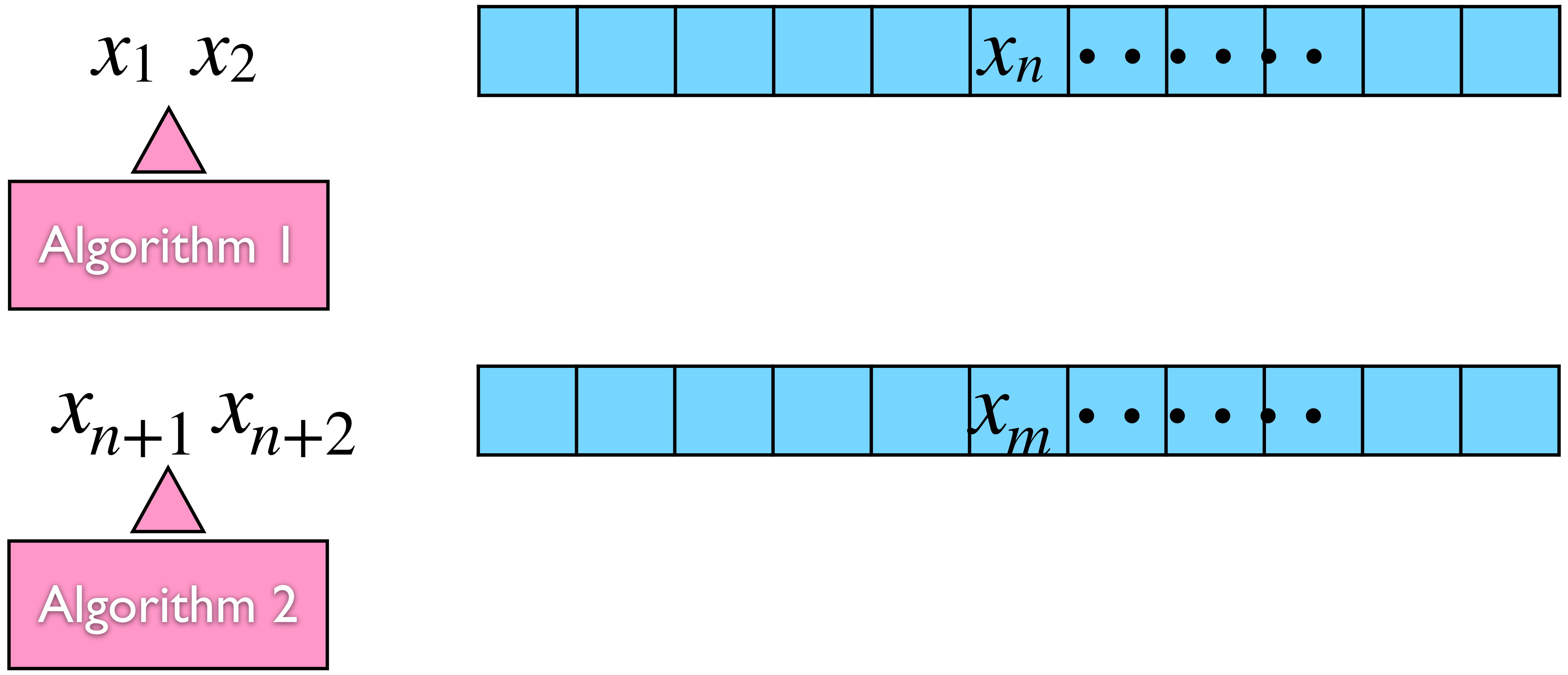
Linear Sketch

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)] + +$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

- Stream in parallel: multiple streams & multiple machines



Observation: $\mathrm{CMS}_{\mathrm{all}} = \mathrm{CMS}_1 + \mathrm{CMS}_2 + \ldots$

Heavy hitters with point query

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)]++$ for all $1 \leq j \leq k$;

Query x: return $\hat{f}_x = \min_{1 \leq j \leq k} \mathrm{CMS}[j][h_j(x)]$

- Heavy hitters: items that appears $> \epsilon n$ times

- Find heavy hitters with point queries: enumerate $x \in U$, output if $\hat{f}_x > \epsilon n$
  - We need $\delta = O(1/N)$
  - Time cost $O(kN)$

Large N? $n \ll N$?

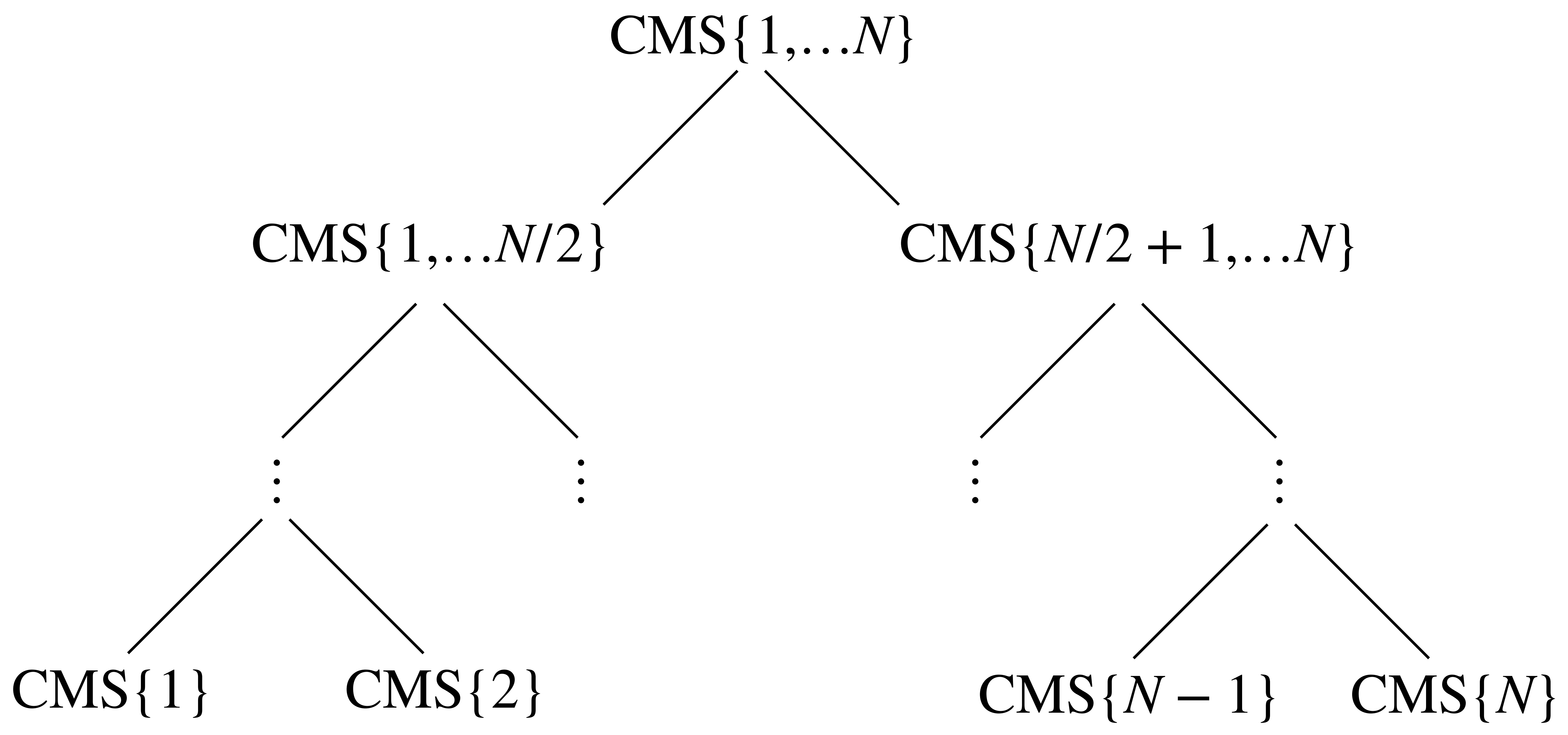
- Divide-and-conquer over N

Heavy hitters with point query by D'n'C

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)]++$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$



$\mathrm{CMS}\{1,\ldots N\}$
$\mathrm{CMS}\{1,\ldots N/2\}$
$\mathrm{CMS}\{N/2+1,\ldots N\}$
$\mathrm{CMS}\{1\}$
$\mathrm{CMS}\{2\}$
$\mathrm{CMS}\{N-1\}$
$\mathrm{CMS}\{N\}$

Tug-of-War (2nd Frequency Moments)

Second frequency moments

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$
Query: return the *2nd frequency moments*

Estimate *2nd frequency moments* $F_2 = \sum f_x^2$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)]++$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

Tug-of-War

- Does Count-Min sketch work for F_2?
- $f_1 = (1,1,\ldots,1)$ and $f_2 = (1,1,\ldots,1,\sqrt{n})$ with $\|f_1\|_2^2 = n$ and $\|f_2\|_2^2 = 2n-1$
  - Can't distinguish with $O(\sqrt{n})$ space because the 1s overwhelm the $\sqrt{n}$
- Idea: assign each item with *random sign*. 1s cancel each other, while $\sqrt{n}$ is kept

Tug-Of-War Algorithm

Count Sketch: z

Upon each x_i: $z \leftarrow z + \sigma(x_i)$

Query: return z^2

2-universal sign function $\sigma : [N] \to \{-1, +1\}$

- How correct is it? Unbiased: $\mathbb{E}[z^2] = F_2$
- Proof: $\mathbb{E}[z^2] = \mathbb{E}\left[\left(\sum_x \left(\sigma(x) f_x\right)^2\right)\right] = \mathbb{E}\left[\sum_{x,y} \sigma(x)\sigma(y) f_x f_y\right] = \sum_{x,y} \mathbb{E}\left[\sigma(x)\sigma(y)\right] f_x f_y.$

Observation: if $x = y$, $\mathbb{E}\left[\sigma(x)\sigma(y)\right] = 1$; o.w. $\mathbb{E}\left[\sigma(x)\sigma(y)\right] = 0.$

$$\Longrightarrow \mathbb{E}[z^2] = \sum_{x,y} \mathbb{E}\left[\sigma(x)\sigma(y)\right] f_x f_y = \sum_x f_x f_x = F_2$$

Tug-Of-War Algorithm

Count Sketch: z

Upon each x_i: $z \leftarrow z + \sigma(x_i)$

Query: return z^2

Chebyshev's Inequality

For random variable X, for any $t > 0$,

$$\Pr\left[\,|X - \mathbb{E}[X]| \geq t\,\right] \leq \frac{\mathbf{Var}[X]}{t^2}$$

- Bound the deviation with variance (again) with Chebyshev.
- Claim: $\mathbf{Var}(z^2) = O(F_2^2)$
- Proof: $\mathbf{Var}(z^2) = \mathbb{E}[z^4] - \left(\mathbb{E}[z^2]\right)^2$.

  Suffices to bound $\mathbb{E}[z^4] = \mathbb{E}[(\sum_x \sigma(x) f_x)^4] = \sum f_a f_b f_c f_d \cdot \mathbb{E}[\sigma(a)\sigma(b)\sigma(c)\sigma(d)]$.

  Observation: if the distinctness is 4-0-0-0 or 2-2-0-0, $\mathbb{E}[\sigma(a)\sigma(b)\sigma(c)\sigma(d)] = 1$; o.w. $= 0$.

Tug-Of-War Algorithm

Count Sketch: z

Upon each x_i: $z \leftarrow z + \sigma(x_i)$

Query: return z^2

Chebyshev's Inequality

For random variable X, for any $t > 0$,

$$\Pr\left[\,|X - \mathbb{E}[X]| \geq t\,\right] \leq \frac{\mathbf{Var}[X]}{t^2}$$

- Claim: $\mathbf{Var}(z^2) = O(F_2^2)$
- Proof: $\mathbf{Var}(z^2) = \mathbb{E}[z^4] - \left(\mathbb{E}[z^2]\right)^2 \leq \sum f_a f_b f_c f_d \cdot \mathbb{E}[\sigma(a)\sigma(b)\sigma(c)\sigma(d)]$

Observation: if the distinctness is 4-0-0-0 or 2-2-0-0, $\mathbb{E}[\sigma(a)\sigma(b)\sigma(c)\sigma(d)] = 1$; o.w. $= 0$.

$$\mathbf{Var}(z^2) \leq \sum_a f_a^4 + 3\sum_{a \neq b} f_a^2 f_b^2 \leq (\sum_a f_a^2)^2 + 3 \cdot (\sum_a f_a^2)^2 = 4F_2^2$$

Any idea?

- $\Pr\left[\left|z^2 - F_2\right| \geq \epsilon F_2\right] \leq \mathbf{Var}(z^2)/\epsilon^2 F_2^2 \leq 4/\epsilon^2$. Meaningless. $4/\epsilon^2 \leq 1/2 \iff \epsilon \geq \sqrt{8}$ **:(**

Tug-Of-War Algorithm+

Count Sketch: z

Upon each x_i: $z \leftarrow z + \sigma(x_i)$

Query: return z^2

Count Sketch+: $\mathrm{CS}[k]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CS}[j] \leftarrow \mathrm{CS}[j] + \sigma_j(x_i)$, for all $j \leq k$

Query: return $z^2 = \sum \mathrm{CS}[j]^2 \big/ k$

k independent 2-universal sign functions $\sigma_1, \ldots, \sigma_k : [N] \to \{-1, +1\}$

- Unbiased by the linearity of expectation: $\mathbb{E}[z^2] = F_2$.
- Lower variance: $\mathbf{Var}(z^2) \leq 4F_2^2 \big/ k$ by the independence.
- $\Pr\left[\left|z^2 - F_2\right| \geq \epsilon F_2\right] \leq \mathbf{Var}(z^2)/\epsilon^2 F_2^2 \leq 4/k\epsilon^2 \leq \delta$
- With space cost $k = 4/\epsilon^2\delta$, we have $\Pr\left[\left|z^2 - F_2\right| \geq \epsilon F_2\right] \leq \delta$

Tug-Of-War Algorithm++

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Estimate the *frequency* $f_x = |\{i : x_i = x\}|$ of x.

Count Sketch++: $\mathrm{CS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CS}[j][h_j(x_i)] \leftarrow \mathrm{CS}[j][h_j(x_i)] + \sigma_j(x_i), \ \forall j \le k$

Query x: return $\hat{f}_x =$ median among $\sigma_j(x)\mathrm{CS}[j][h_j(x)]$

- Look at one counter: $E_j := \sigma_j(x)\mathrm{CS}[j][h_j(x)] - f_x = \sum_{y \neq x} \sigma_j(x)\sigma_j(y) I\left[h_j(x) = h_j(y)\right] f_y$
- Correct as long as $> 1/2$ counters are correct: $|E_j| \le \epsilon F_2$
- $\Pr\left[\hat{f}_x = f_x \pm \epsilon F_2\right] \ge 1 - \delta$

Tug-Of-War Algorithm++

Count Sketch++: $\mathrm{CS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CS}[j][h_j(x_i)] \leftarrow \mathrm{CS}[j][h_j(x_i)] + \sigma_j(x_i)$, $\forall j \le k$

Query x: return $\hat{f}_x =$ median among $\sigma_j(x)\mathrm{CS}[j][h_j(x)]$

Chernoff-Hoeffding Bound:

$$\Pr\left[X \ge \mathbb{E}[X] + t\right] \le \exp\left(-\frac{2t^2}{n}\right)$$

$$\Pr\left[X \le \mathbb{E}[X] - t\right] \le \exp\left(-\frac{2t^2}{n}\right)$$

- Look at one counter: $E_j := \sigma_j(x)\mathrm{CS}[j][h_j(x)] - f_x = \sum_{y \ne x} \sigma_j(x)\sigma_j(y) I\left[h_j(x) = h_j(y)\right] f_y$
- Correct as long as $> 1/2$ counters are correct: $|E_j| \le \epsilon F_2$

Set $m = 9\epsilon^2$

- Claim: $\mathbb{E}[\,|E_j|\,] \le F_2/\sqrt{m}$. Markov's inequality: $\Pr\left[|E_j| \ge 3F_2/\sqrt{m}\right] \le 1/3$
- $\Pr\left[\sum_i^k Y_i \le k/2\right] \le \Pr\left[\sum_i Y_i - k\mu \le -k/6\right] \le \exp\left(-\frac{2(k/6)^2}{k}\right) = \exp\left(-\frac{k}{18}\right) \le \delta$
- Space cost: $km = O\left(\epsilon^{-2}\log(1/\delta)\right)$

Set $k = 18\log(1/\delta)$

Tug-Of-War Algorithm++

Count Sketch++: $\mathrm{CS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CS}[j][h_j(x_i)] \leftarrow \mathrm{CS}[j][h_j(x_i)] + \sigma_j(x_i), \ \forall j \leq k$

Query x: return $\hat{f}_x$ = median among $\sigma_j(x)\mathrm{CS}[j][h_j(x)]$

- Look at one counter: $E_j := \sigma_j(x)\mathrm{CS}[j][h_j(x)] - f_x = \sum_{y \neq x} \sigma_j(x)\sigma_j(y) I\left[h_j(x) = h_j(y)\right] f_y$
- Claim: $\mathbb{E}[\,|E_j|\,] \leq F_2/\sqrt{m}$.
- Proof: $\mathbb{E}[\,|E_j|\,]^2 \leq \mathbb{E}[E_j^2] = \mathbb{E}\left[\sum_{y,z \neq x} \sigma_j(y)\sigma_j(z) I\left[h_j(x) = h_j(y)\right] I\left[h_j(x) = h_j(z)\right] f_y f_z\right]$

Tug-Of-War Algorithm++

Count Sketch++: $\mathrm{CS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CS}[j][h_j(x_i)] \leftarrow \mathrm{CS}[j][h_j(x_i)] + \sigma_j(x_i),\ \forall j \leq k$

Query x: return $\hat{f}_x$ = median among $\sigma_j(x)\mathrm{CS}[j][h_j(x)]$

$$E_j := \sum_{y \neq x} \sigma_j(x)\sigma_j(y) I\left[h_j(x) = h_j(y)\right] f_y$$

- $\mathbb{E}[|E_j|]^2 \leq \mathbb{E}[E_j^2] = \mathbb{E}\left[\sum_{y,z \neq x} \sigma_j(y)\sigma_j(z) I\left[h_j(x) = h_j(y)\right] I\left[h_j(x) = h_j(z)\right] f_y f_z\right]$

$$= \mathbb{E}\left[\sum_{y \neq x} I\left[h_j(x) = h_j(y)\right] f_y^2 + \sum_{y,z \neq x, y \neq z} \sigma_j(y)\sigma_j(z) I\left[h_j(x) = h_j(y)\right] I\left[h_j(x) = h_j(z)\right] f_y f_z\right]$$

$$= \sum_{y \neq x} \mathbb{E}\left[I\left[h_j(x) = h_j(y)\right]\right] f_y^2 + \sum_{y,z \neq x, y \neq z} \mathbb{E}\left[\sigma_j(y)\sigma_j(z)\right] \mathbb{E}\left[I\left[h_j(x) = h_j(y)\right] I\left[h_j(x) = h_j(z)\right]\right] f_y f_z$$

Tug-Of-War Algorithm++

Count Sketch++: $\mathrm{CS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CS}[j][h_j(x_i)] \leftarrow \mathrm{CS}[j][h_j(x_i)] + \sigma_j(x_i)$, $\forall j \le k$

Query x: return $\hat{f}_x$ = median among $\sigma_j(x)\mathrm{CS}[j][h_j(x)]$

$$E_j := \sum_{y \neq x} \sigma_j(x)\sigma_j(y) I\left[h_j(x) = h_j(y)\right] f_y$$

- $\mathbb{E}[\,|E_j|\,]^2$

$$\le \sum_{y\neq x} \mathbb{E}\left[I\left[h_j(x) = h_j(y)\right]\right] f_y^2 + \sum_{y,z\neq x, y\neq z} \mathbb{E}\left[\sigma_j(y)\sigma_j(z)\right] \mathbb{E}\left[I\left[h_j(x) = h_j(y)\right] I\left[h_j(x) = h_j(z)\right]\right] f_y f_z$$

$$= \sum_{y\neq x} \Pr\left[h_j(x) = h_j(y)\right] f_y^2 + \sum_{y,z\neq x, y\neq z} \mathbb{E}\left[\sigma_j(y)\right] \mathbb{E}\left[\sigma_j(z)\right] \mathbb{E}\left[I\left[h_j(x) = h_j(y)\right] I\left[h_j(x) = h_j(z)\right]\right] f_y f_z$$

$$= \sum_{y\neq x} \Pr\left[h_j(x) = h_j(y)\right] f_y^2 \le F_2^2 / m$$

Filters

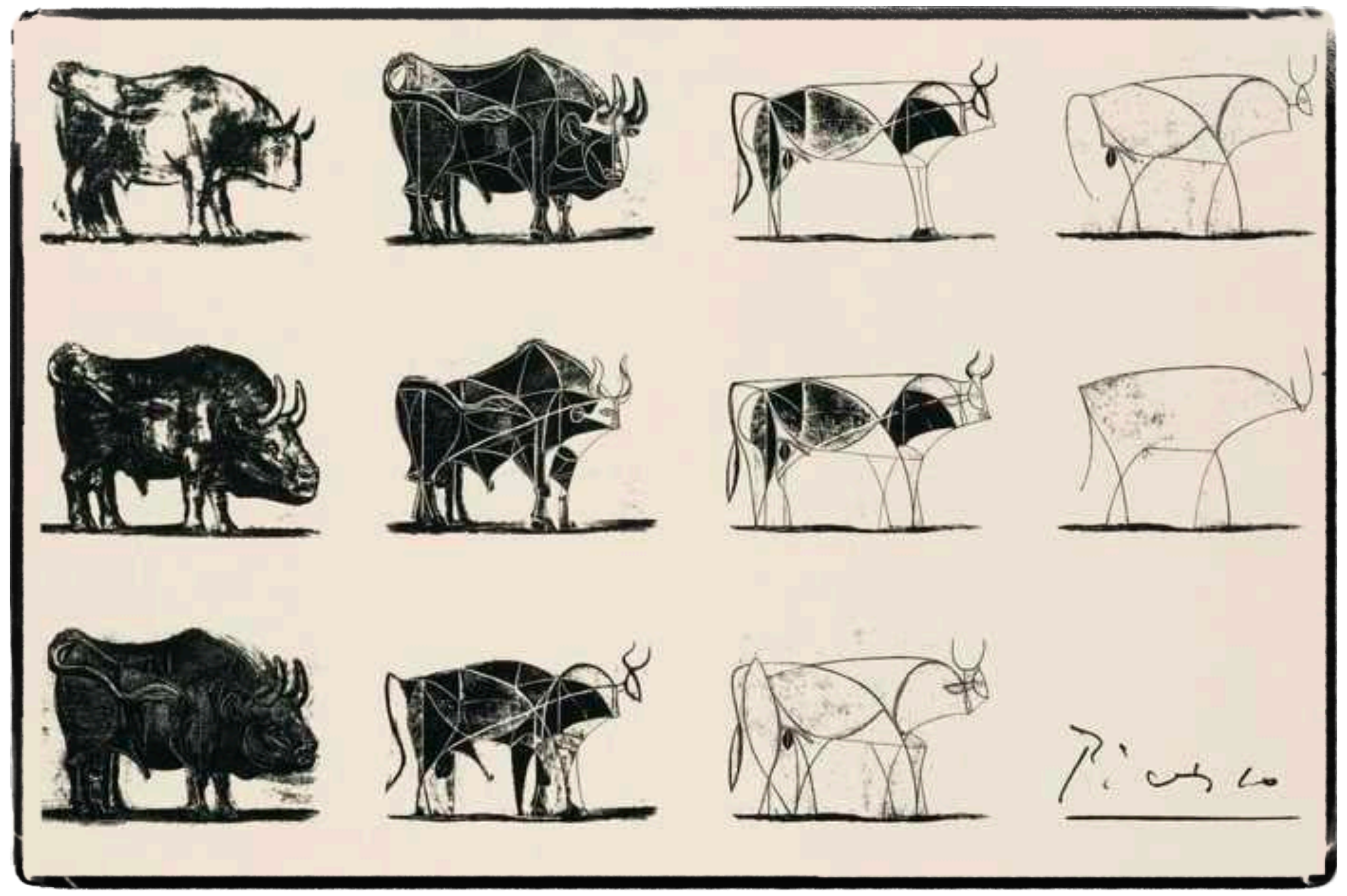
Data Structure for Set

Data: a set S of n items $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Determine whether $x \in S$.

- **Space cost:** size of data structure (in bits)
  - entropy of a set: $\log \binom{N}{n} = \Omega(n \log N)$ bits (when $N \gg n$)
- **Sketch:** lossy representation of S using < entropy space



Approximate Membership

Data: a set S of n items $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Answer whether $x \in S$ with false positivity.

- uniform hash function $h : U \to [m]$ (m to be fixed)

Bloom Filter: bit array $A \in \{0,1\}^m$

  A is initialized to all 0's;

  for each $x_i \in S$: set $A[h(x_i)] = 1$;

Query x: answer "yes" iff $A[h(x)] = 1$

- $x \in S$: always correct
- $x \notin S$: false positive $\Pr\left[A[h(x)] = 1\right] = 1 - (1 - 1/m)^n \approx 1 - \mathrm{e}^{-n/m}$

Bloom Filters (Bloom 1970)

Data: a set S of n items $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Answer whether $x \in S$ with false positivity.

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

(k and m to be fixed)

Bloom Filter+: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \le j \le k$;

Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \le j \le k$

Bloom Filters (Bloom 1970)

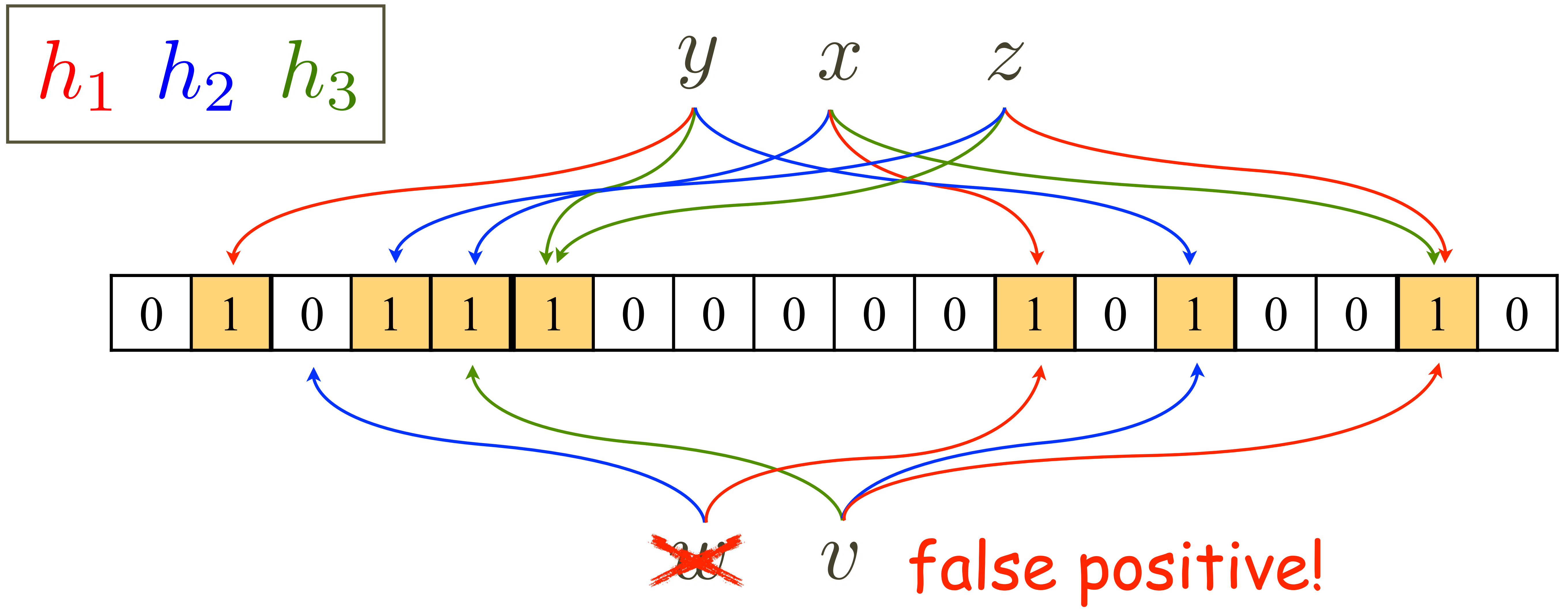
- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Bloom Filter+: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \le j \le k$;

Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \le j \le k$



Data: set $S \subseteq U$ of size n **Query**: $x \in U$

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Bloom Filter+: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \le j \le k$;

Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \le j \le k$

- $x \in S$: always correct
- $x \notin S$: false positive

choose $k = c \ln 2$

$m = cn$

$$\Pr\left[\forall 1 \le j \le k: \ A[h_j(x)] = 1\right]$$

heuristic

$$= \left(\Pr\left[A[h_j(x)] = 1\right]\right)^k = \left(1 - \Pr\left[A[h_j(x)] = 0\right]\right)^k$$

$$\le (1 - (1 - 1/m)^{kn})^k \approx \left(1 - \mathrm{e}^{-kn/m}\right)^k = 2^{-c \ln 2} \le (0.6185)^c$$

real prob:

$$\frac{1}{m^{k(n+1)}} \sum_{i=1}^{m} i^k i! \binom{m}{i} \left\{ {kn \atop i} \right\}$$

Data: set $S \subseteq U$ of size n **Query**: $x \in U$

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Bloom Filter+: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \le j \le k$;

Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \le j \le k$

- $x \in S$: always correct
- $x \notin S$: false positive

$$\Pr\left[\forall 1 \le j \le k : \ A[h_j(x)] = 1\right]$$

$$= \left(1 - \frac{\#0\text{'s}}{\#\text{slots}}\right)^k$$

$$= \sum_t \Pr[q_n = t](1 - t/m)^k$$

$$\mathbb{E}[\#0\text{'s}] = m\,(1 - 1/m)^{kn}$$

- $q_i \triangleq \mathbb{E}[\#0\text{'s}]$ after i insertions
- $\{q_i\}_{i \le n}$ is martingale with $|q_i - q_{i-1}| \le k, \forall i$

Martingale Tail Inequality

- Azuma's inequality: If a sequence $\{Y_n : n \geq 0\}$ is a martingale (with respect to some sequence $\{X_n : n \geq 0\}$), and for all $n \geq 1$,

$$\left| Y_n - Y_{n-1} \right| \leq c_n$$

then any $n \geq 1$ and $t > 0$:

$$\Pr\left(\left| Y_n - Y_0 \right| \geq t\right) \leq 2 \exp\left(-\frac{2t^2}{\sum_{i=1}^n c_i^2}\right)$$

Proof: Difference $D_i = Y_i - Y_{i-1}$ and Sum $S_n = \sum_{i=1}^{n} D_i = Y_n - Y_0$

New goal: $\Pr\left(\left| S_n \right| \geq t\right) \leq 2 \exp\left(-\frac{2t^2}{\sum_{i=1}^n c_i^2}\right)$

Data: set $S \subseteq U$ of size n **Query**: $x \in U$

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Bloom Filter+: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \leq j \leq k$;

Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \leq j \leq k$

Azuma's Inequality

For martingale with $\forall n, \left|Y_n - Y_{n-1}\right| \leq c_n$

$$\Pr\left(\left|Y_n - Y_0\right| \geq t\right) \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^n c_i^2}\right)$$

- $x \in S$: always correct
- $x \notin S$: false positive

$$\Pr\left[\forall 1 \leq j \leq k : \ A[h_j(x)] = 1\right]$$

$$= \left(1 - \frac{\text{#0's}}{\text{#slots}}\right)^k$$

$$= \sum_t \Pr[q_n = t](1 - t/m)^k \approx (1 - (1 - 1/m)^{kn})^k = 2^{-c\ln 2} \leq (0.6185)^c$$

$$\mathbb{E}[\text{#0's}] = m\,(1 - 1/m)^{kn}$$

- $q_i \triangleq \mathbb{E}[\text{#0's}]$ after i insertions
- $\{q_i\}_{i \leq n}$ is martingale with $|q_i - q_{i-1}| \leq k, \forall i$
- $\Pr[|q_n - \mathbb{E}[q_n]| \geq \lambda/m] \leq 2\exp(-2\lambda^2/kn)$

concentrate around $\lambda = O(\sqrt{kn})$

Bloom Filters (Bloom 1970)

Data: a set S of n items $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Answer whether $x \in S$ with false positivity.

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Bloom Filter+: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h(x_i)] = 1$;

Query x: answer "yes" iff $A[h(x)] = 1$

- choose $k = c \ln 2$ and $m = cn$
  - space cost: $m = cn$ bits, time cost: $k = c \ln 2$
  - false positive $\leq (0.6185)^c$

- space: $m = O(\log(1/\epsilon)n)$ bits
- time: $k = O(\log(1/\epsilon))$

$\epsilon = (0.6185)^c$

Counting Bloom Filters

Bloom Filter+: bit array $A \in \{0,1\}^m$

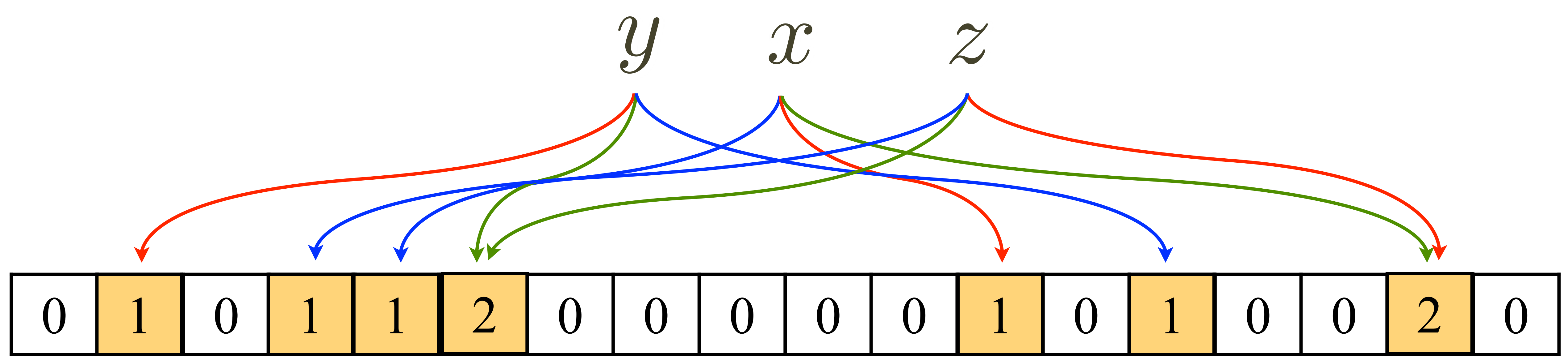
A is initialized to all 0's;

Insert x: set $A[h(x_i)] = 1$;

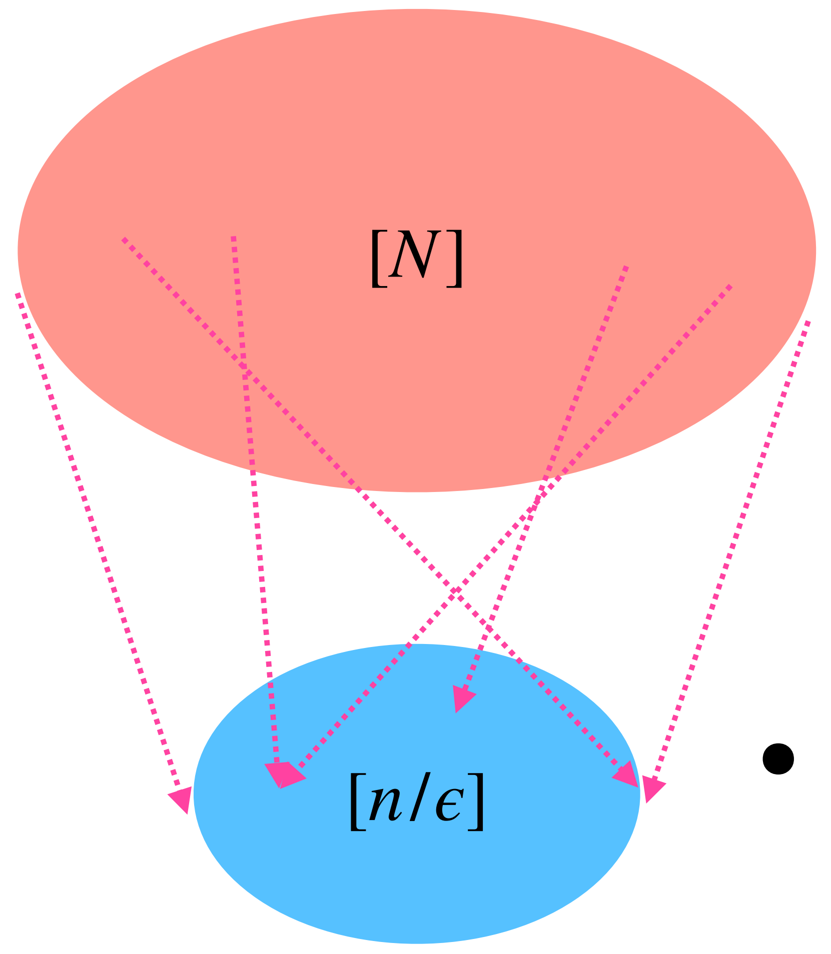
Query x: answer "yes" iff $A[h(x)] = 1$

deletion?

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$



Approximate by Exact



Data: a set S of n items $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Answer whether $x \in S$ with false positivity.

- Assume an exact membership data structure
- uniform & 2-universal hash function $h : U \to [n/\epsilon]$

Sketching: initialize exact set S over $[n/\epsilon]$

Insert x: insert $h(x)$ to S

Query x: answer "yes" iff $h(x) \in S$

deletion?

- false positive: ϵ
- time cost: $O(1)$
- space cost: $O(n)$ words $\approx O(n \log(1/\epsilon))$ bits

- Bloom filter
  - space: $m = O(\log(1/\epsilon)n)$ bits
  - time: $k = O(\log(1/\epsilon))$