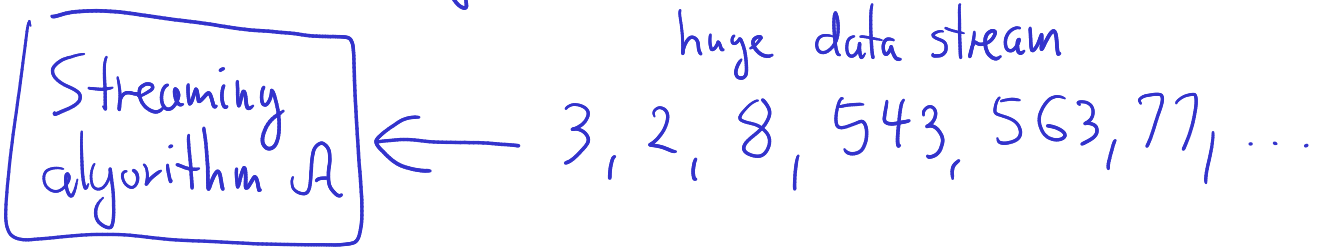
DS-563 / CS-543 2024-02-06

Today

- Estimating the number of distinct elements (F_0)
- Practical algorithms for F_0
- Space complexity of streaming moments

Model: Streaming algorithms



X - universe of elements of the stream

for $x \in X$, $f(x) = \#$ occurrences of x

Goal:

Estimate number of distinct elements

$$F_0 = |\{x \in X : f(x) \neq 0\}|$$

More precisely: compute $(1+\varepsilon)$-multiplicative approximation

$$(1-\varepsilon) F_0 \leq \hat{F}_0 \leq (1+\varepsilon) F_0$$

$$\Leftrightarrow |\hat{F}_0 - F_0| \leq \varepsilon F_0$$

Algorithm (insertions only)

h = "fully random" hash function from X to N s.t.

$$\Pr(h(x) = i) = 2^{-(i+1)}$$

$h(x) =$	0	1	2	...
$\Pr(\quad) =$	1/2	1/4	1/8	...

Initialization:

$z \leftarrow 0$

$A \leftarrow \phi$

Processing:

For each element x in the stream:

if $h(x) \geq z$:

$A \leftarrow A \cup \{(x, h(x))\}$

while $|A| \geq C/\varepsilon^2$

$z \leftarrow z+1$

remove from A all pairs (y, g) s.t. $g < z$

large constant ($C > 576$)

Estimate:

Output $|A| \cdot 2^z$

Analysis:

Random variables:

Z - final value of z

$$X_{i,x} = \begin{cases} 1 & \text{if } h(x) \geq i \\ 0 & \text{o.w.} \end{cases} \quad (\text{for } x \in X, i \in \mathbb{N})$$

$$Y_i = \sum_{x : f(x) > 0} X_{i,x} \quad (\text{for } i \in \mathbb{N})$$

Output of the algorithm: $Y_Z \cdot 2^Z$

Incorrect output:

$$|Y_Z \cdot 2^Z - F_0| > \varepsilon F_0$$

$$\Downarrow$$

$$\left| Y_Z - \frac{F_0}{2^Z} \right| > \frac{\varepsilon F_0}{2^Z}$$

Observation: if $Z = 0$, $|A| = Y_0 = F_0$

The algorithm outputs exact value

$$\Pr(\text{incorrect output}) = \sum_{i=1}^{\infty} \Pr\left(\left|Y_i - \frac{F_0}{2^i}\right| > \frac{\varepsilon F_0}{2^i} \wedge Z = i\right)$$

$$= \otimes$$

Select integer s s.t.

$$\frac{12}{\varepsilon^2} \leq \frac{F_0}{2^s} < \frac{24}{\varepsilon^2}$$

Observation: if $s \leq 1$, $F_0 < \frac{48}{\varepsilon^2} \ll \frac{C}{\varepsilon^2}$

Hence $Z = 0$ and the algorithm outputs the exact value

So assume $s > 1$ for the rest of the proof

$$\otimes \leq \underbrace{\sum_{i=1}^{s-1} \Pr\left(\left|Y_i - \frac{F_0}{2^i}\right| > \frac{\varepsilon F_0}{2^i}\right)}_{\Delta} + \underbrace{\sum_{i=s}^{\infty} \Pr(Z = i)}_{\square}$$

$$\square = \Pr(Z \geq s) = \Pr(Y_{s-1} \geq C/\varepsilon^2)$$

$$\leq \frac{\mathbb{E}[Y_{s-1}]}{C/\varepsilon^2} = \frac{F_0/2^{s-1}}{C/\varepsilon^2} = \frac{2\varepsilon^2}{C} \cdot \frac{F_0}{2^s}$$

↑ Markov's

$$\leq \frac{2\varepsilon^2}{C} \cdot \frac{24}{\varepsilon^2} = \frac{48}{C} < \frac{1}{12}$$

↑ Selection of s

↑ large C

$\Delta \leq ?$

$$\mathbb{E}[Y_i] = \frac{F_0}{2^i}$$

$$\mathrm{Var}[Y_i] = \sum_{x: f(x)>0} \mathrm{Var}[X_{i,x}] \leq \sum_{x: f(x)>0} \mathbb{E}[X_{i,x}^2]$$

$$= \sum_{x: f(x)>0} \mathbb{E}[X_{i,x}] = \mathbb{E}[Y_i] = \frac{F_0}{2^i}$$

$$\Delta = \sum_{i=1}^{s-1} \Pr\left(|Y_i - \mathbb{E}[Y_i]| > \frac{\varepsilon F_0}{2^i} \right)$$

$$\leq \sum_{i=1}^{s-1} \Pr\left(|Y_i - \mathbb{E}[Y_i]| > \varepsilon \sqrt{\frac{F_0}{2^i}} \cdot \sqrt{\mathrm{Var}[Y_i]} \right)$$

$$\leq \sum_{i=1}^{s-1} \frac{2^i}{\varepsilon^2 F_0} = \frac{1}{\varepsilon^2 F_0} \sum_{i=1}^{s-1} 2^i \leq \frac{2^s}{\varepsilon^2 F_0} \leq \frac{1}{\varepsilon^2} \cdot \frac{\varepsilon^2}{12} = \frac{1}{12}$$

↑ Chebyshev's inequality

↑ selection of s

$$\circledast \leq \Delta + \square \leq \frac{1}{12} + \frac{1}{12} = \frac{1}{6}$$

Conclusion: the algorithm outputs a $(1+\varepsilon)$-multiplicative approximation w.p. at least $5/6$

Space usage:

$O(1/\varepsilon^2)$ elements of X + $O(1/\varepsilon^2)$ integers

can be reduced by hashing into a small range (2-wise independence is sufficient)

at most $O(\log\log m)$ bits with high probability for further space savings (m = stream length)

Optimal: $O(1/\varepsilon^2 + \log n)$ bits for $X = [n]$

Hash function h?

- 2-wise independence suffices, needed for $\mathrm{Var}[Y_i]$
- Non-uniform distribution on some range? See Homework 3?

Practical F_0 algorithms? See:

- HyperLogLog
- HyperLogLog++

Space complexity of streaming F_p

$X = [n]$, stream length $= O(\mathrm{poly}(n))$

$p \in [0,2]$: $O\left(\frac{1}{\varepsilon^2}\log n\right)$

$p > 2$: $n^{1-2/p} \cdot \mathrm{poly}\left(\frac{\log n}{\varepsilon}\right)$