DS-563 / CS-543 Lecture 1 / 2023-01-19

Website: https://onak.pl/ds563
cs543

Course content: algorithms for big data
- data projections (including streaming algorithms, nearest neighbor search)
- representative subsets
- sampling from probability distribution
- querying & sampling subsets of data sets
- distributed computation

Online tools:
- Piazza (prefered way to contact us!)
- Gradescope

Course requirements:
- active participation: 5%
- two theoretical homeworks: 25%
- three programming assignments: 25%
- final project proposal: 5%
- final project: 40%

Prerequisites:

- programming
- basic algorithms
- math: linear algebra / ~~algorithms~~ / probability / calculus

See self-assessment questionnaire

Don't know something ⇒ ask, opportunity to learn

Electronic device ~~policy~~ policy

tL;dr no devices ~~except~~ except for taking notes

Assigment 0: upload a recording of your name to gradescope (wav/mp3/ogg/...)

Today:

- Count Min Sketch
- after seeing lots of items, provide estimates of frequency for any item

Setting:

- multiset of items from some universe X
- arrive in arbitrary ~~time~~ order

Goal: design data structure that allows for

- adding a new item $x \in X$
- querying: what fraction is $y \in X$?

Example: online store,
"what fraction of queries is 'BU mascot'?"

Straightforward solution:

explicitly store mapping $x \in X \to$ # occurrences of x

Lots of space!

Will use less by allowing:

- small additive approximation, say, $\pm 0.01\%$
- incorrect answer w.p. $\delta \in (0,1)$

COMMON THEME IN DS563!

First attempt: bucketing

Suppose random hash function $h: X \to [k]$

$= \{1, \ldots, k\}$

Store array $A[1 \ldots k]$ of integers

Initially $A[i] = 0$ for all i

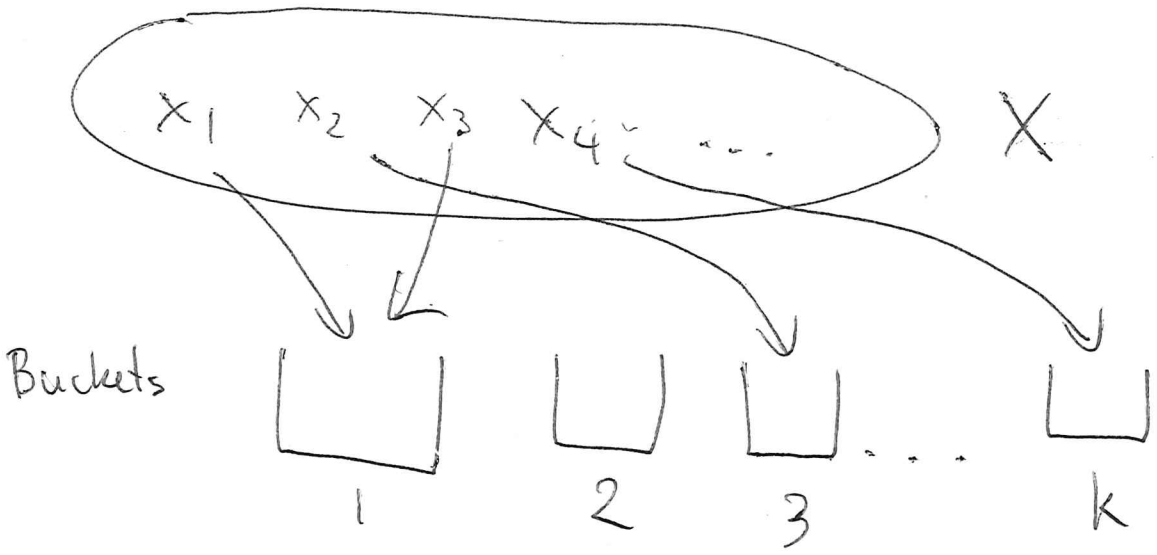
Adding new element x:

$A[h(x)] \leftarrow A[h(x)] + 1$

Estimate $g(y)$ for $y \in X$:

return $A[h(y)]$

$\sum_i A[i] = S$ = total number of items



How good is this!?

- can overestimate by a lot!!!
- never underestimate: $g(y) \geq \frac{f(y)}{5}$

Analysis:

$f(y)$ = real number of occurrences of y

$$g(y) = \frac{1}{5} \sum_{\substack{x \in X \\ h(x)=h(y)}} f(x) = \frac{1}{5}\left(f(y) + \sum_{\substack{x \in X \\ x \neq y}} C_{x,y} f(x)\right)$$

collision $C_{x,y} = \begin{cases} 0 & h(x) \neq h(y) \\ 1 & h(x) = h(y) \end{cases}$

↑ random variable

h fully random $\Rightarrow$ $\mathbb{E}[C_{x,y}] = \frac{1}{k}$ for $x \neq y$

$$g(y) = \frac{f(y)}{S} + \boxed{\frac{1}{S}\sum_{\substack{x \in X \\ x \neq y}} C_{x,y} f(x)}$$

$$\text{error} = \text{error}$$

$\text{error} \geq 0$

$$\mathbb{E}[\text{error}] = \frac{\sum_{\substack{x \in X \\ x \neq y}} f(x)\,\mathbb{E}[C_{x,y}]}{S} = \frac{1}{k} \cdot \frac{\sum_{\substack{x \in X \\ x \neq y}} f(x)}{S}$$

$$\leq \frac{1}{k} \frac{\sum_{x \in X} f(x)}{S} = \frac{1}{k}$$

($\sum_{x \in X} f(x) = S$)

Markov's inequality

X – non-negative random variable

$a > 0$

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

$$\Pr\left[\text{error} \geq \frac{2}{k}\right] \leq \frac{1/k}{2/k} = \frac{1}{2}$$

Set $k = \lceil 2/\varepsilon \rceil$ to get additive ε approximation w.p. $\frac{1}{2}$

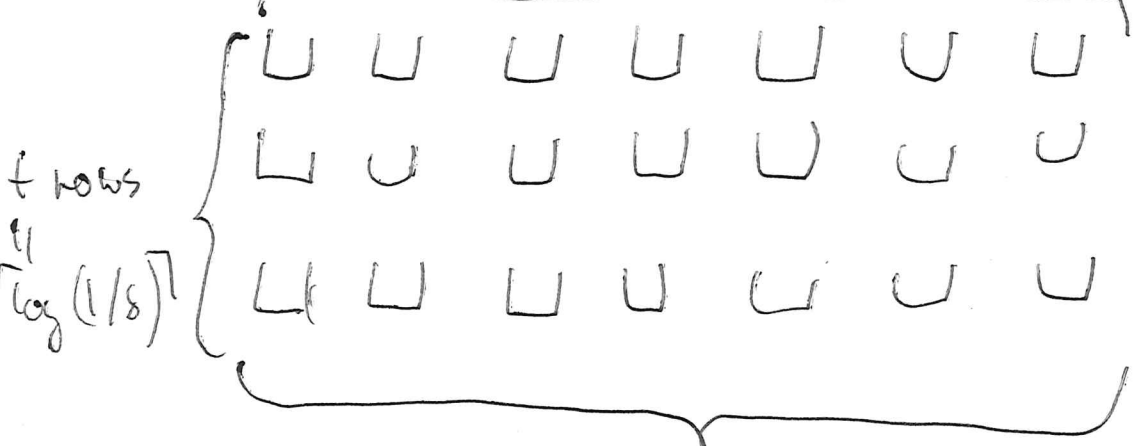
How to make probability of error at most $\delta \in (0, \frac{1}{2})$?

- Run $t = \lceil \log(1/\delta) \rceil$ independent copies
- Query y: return the <u>minimum</u> of all estimates

$$\Pr[\text{all wrong}] \leq \left(\frac{1}{2}\right)^t \leq 1/\delta$$

$\Pr[\text{all wrong}]$ = overestimate by more than ε

This is CountMin Sketch



$k = \lceil 2/\varepsilon \rceil$ buckets each

Total space usage: $O\left(\frac{1}{\varepsilon} \log(1/\delta)\right)$

What is missing?

How do we store random hash functions?

- We can't (lots of space!)
- Pairwise independence suffices:
  - For $x \neq y$: $\mathbb{E}[C_{x,y}] = \Pr[h(x) = h(y)] \leq \frac{1}{k}$
  - $\mathbb{E}[C_{x,y}] \leq \frac{O(1)}{k}$ ← fixed constant

    suffices as well, just increase k by a constant factor
  - example will be covered in the discussion section

Nice properties of CountMin Sketch

- can handle deletions (subtract from specific counters)
- can be computed separately for subsets and easily combined (need to use the same hash function!)

Application: different data centers handling different parts of the data set

---

CountMin Sketch is an example of <u>linear sketch</u>