DS-563/CS-543 2025-01-23

Reminders:

- Course webpage: https://onak.pl/ds563 (or .../cs543)
- Make sure you are on Piazza/Gradescope
- Optional HW0: what name do you want to go by or how to pronounce it, audio file (mp3, wav, ...)

Today:

- Problem: observe an evolving set of items, with items coming and leaving, and provide frequency estimates on request
- Solution: Count-Min Sketch

Setting:

- multiset of items from some universe X
  - initially empty
- items arrive and depart in arbitrary order

Goal: design a data structure that supports the following operations:

- Insert(x): insert $x \in X$ into the multiset
- Delete(x): remove $x \in X$ from the multiset (we assume that x is present in the multiset)
- Query(x): return the number of copies of $x \in X$ in the multiset

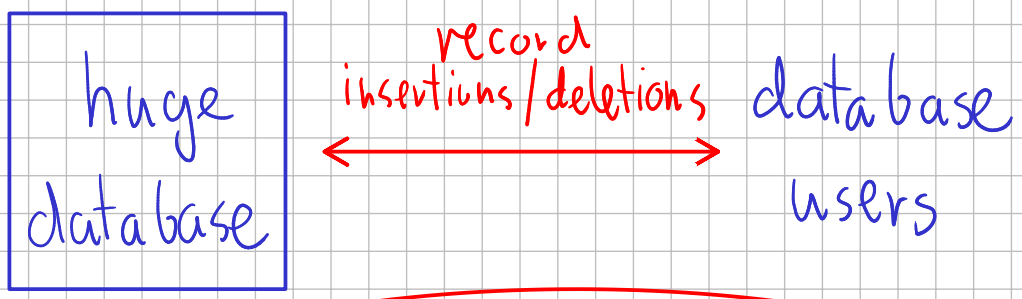
---

Motivation:

Example 1: search engine / online store / ...

What is the number of queries "BU mascot"?

Example 2: tracking statistics inside a large database



quick frequency estimates without reaching into the database

Straightforward solution:

Keep a dictionary/map from X to counts.

(Note: No need to store the counts for $x \in X$ that are not present. This limits the number of counts stored to the number of different elements, which can be much smaller than $|X|$.)

Problem: Potentially lots of space if there are many different elements present.

We will use less by allowing:

- approximation: return something close to the real value
- randomization: with small probability, the returned value may be outside of the desired range

approximation + randomization ← common themes in this class

First attempt: bucketing

Assume a "fully random" hash function

$$h: X \to [k]$$

$$\| \quad \{1, 2, \ldots, k\}$$

fully random $\equiv$ each $h(x)$ distributed uniformly and independently of other values of h

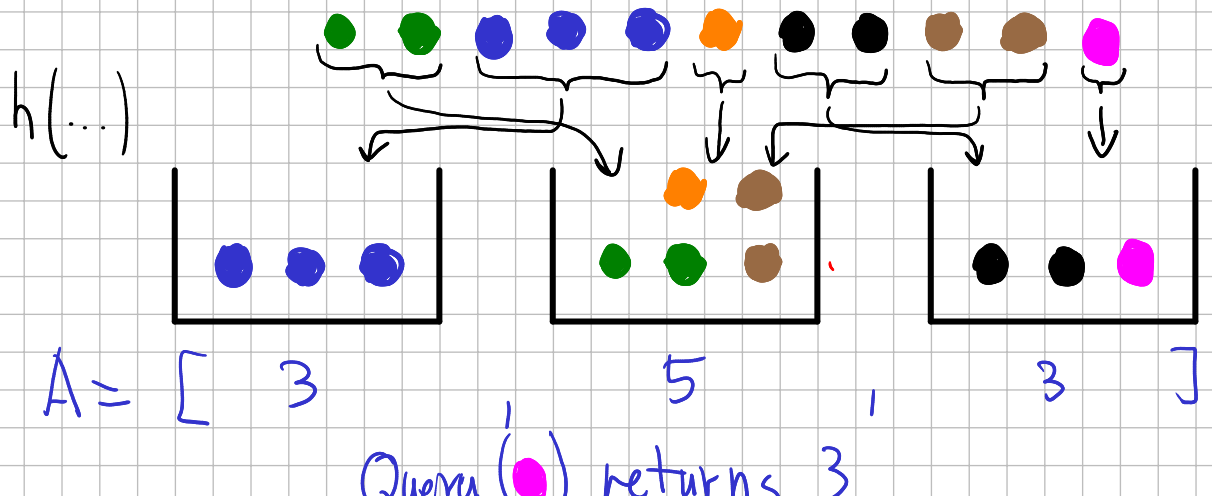
Our solution:

- store array $A[1 \ldots k]$ of integers/counters
- initially, $A[i] = 0$ for all $i \in [k]$
- operations:

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

  Remove (x): $A[h(x)] \leftarrow A[h(x)] - 1$

  Query (x): return $A[h(x)]$



How good are the estimates?

- Can underestimate? No.
- Can overestimate? Yes. By a lot.

Analysis of overestimation:

Definitions:

1. "collision" $C_{x,y} \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } h(x) = h(y) \\ 0 & \text{if } h(x) \neq h(y) \end{cases}$

   x, y: elements of X

   x and y "collide" if they are mapped to the same bucket

2. $f(x) \stackrel{\text{def}}{=}$ exact number of copies of x

3. $g(x) \stackrel{\text{def}}{=}$ the value returned by Query (x)

Analysis:

$$g(x) = \sum_{\substack{y \in X \\ \text{s.t. } h(y)=h(x)}} f(y) = f(x) + \underbrace{\sum_{\substack{y \in X \\ y \neq x}} C_{x,y} f(x)}_{= \text{overestimate} \geq 0}$$

To be continued