CS6931 Database Seminar

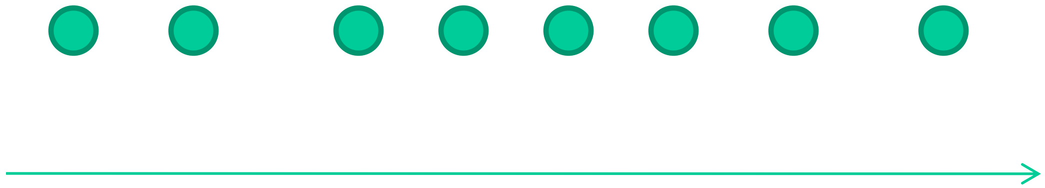
Lecture 4: Streaming Model

Problem One: Missing Card

- I take one from a deck of 52 cards, and pass the rest to you. Suppose you only have a (very basic) calculator and bad memory, how can you find out which card is missing with just one pass over the 51 cards?

- What if there are two missing cards?

A data stream algorithm …



- Makes one pass over the input data
- Uses a small amount of memory (much smaller than the input data)
- Computes something

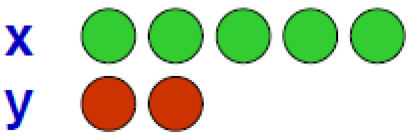
Why do we need streaming algorithms

- Often get to see the data once
- Don't want to store the entire data
- Data stored on disk, sequential scans are much faster

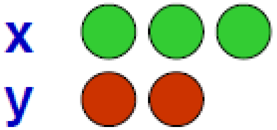
- Data stream algorithms have been a very active research area for the past 15 years
- Problems considered today
  - Missing card
  - Majority
  - Heavy hitters
  - Self-join size

Streaming Model

- We model data streams as sequences of simple tuples
- Complexity arises from massive length of streams
- Arrivals only streams:
  - Example: (x, 3), (y, 2), (x, 2) encodes the arrival of 3 copies of item x, 2 copies of y, then 2 copies of x.



  - Could represent eg. packets on a network; power usage
- Arrivals and departures:
  - Example: (x, 3), (y,2), (x, -2) encodes final state of (x, 1), (y, 2).



  - Can represent fluctuating quantities, or measure differences between two distributions

Technique One: Tricks

Problem two: Majority

- Given a sequence of items, find the majority if there is one

- A A B C D B A A B B A A A A A A C C C D A B A A A
- Answer: A

- Trivial if we have O(n) memory
- Can you do it with O(1) memory and two passes?
  - First pass: find the possible candidate
  - Second pass: compute its frequency and verify that it is > n/2
- How about one pass?
  - Unfortunately, no

Problem three: Heavy hitters

- Problem: find all items with counts > φn, for some 0< φ<1
- Relaxation:
  - If an item has count > φ n, it must be reported, together with its estimated count with (absolute) error < εn
  - If an item has count < ($\varphi - \varepsilon$) n, it cannot be reported
  - For items in between, don't care
- In fact, we will estimate all counts with at most εn error
- Applications
  - Frequent IP addresses
  - Data mining

Technique Two: Counter-Based Algorithms

The algorithm [Metwally, Agrawal, Abbadi, 2006]

- Input parameters: number of counters m, stream S
- Algorithm:

A A B C D B A A B B A A A A A A C C C D A B

```
for each element e in S {
  if e is monitored {
    find counter of e, counter_i;
    counter_i++;
  } else {
    find the element with least frequency, e_m, denote its frequency min;
    replace e_m with e;
    assign counter for e with min+1;
  }
}
```

Properties of the Algorithm

- Actual count of a monitored item $\leq$ counter
- Actual count of a monitored item $\geq$ counter – min
- Actual count of an item not monitored $\leq$ min
  - Proof by induction
- The sum of the counters maintained is *n*
  - Why?
- So min <= n/m
- If we set $m = 1/\varepsilon$, it's sufficient to solve the heavy hitter problem
  - Why?
  - So the heavy hitter problem can be solved in $O(1/\varepsilon)$ space

How about deletions?

- Any deterministic algorithm needs O(1/ε^2) space
  - Why?
  - In fact, Las Vegas randomization doesn't help
- Will design a randomized algorithm that works with *high probability*
  - For any item x, we can estimate its actual count within error εn with probability 1-δ for any small constant δ

Technique Three: Hashing

The Count-Min sketch [Cormode, Muthukrishnan, 2003]

A two-dimensional array counts with width w and depth

Given parameters (ε, δ), set $w = \left\lceil \frac{2}{\varepsilon} \right\rceil$ and $d = \left\lceil \log \frac{1}{\delta} \right\rceil$.

d hash functions are chosen randomly from a 2-universal family (pair-wise independent)

For example, we can choose a prime number $p > u$, and random a_j, b_j, $j=1,\ldots,d$. Define:

$$h_j(x) = (a_j\, x + b_j \bmod p) \bmod w$$

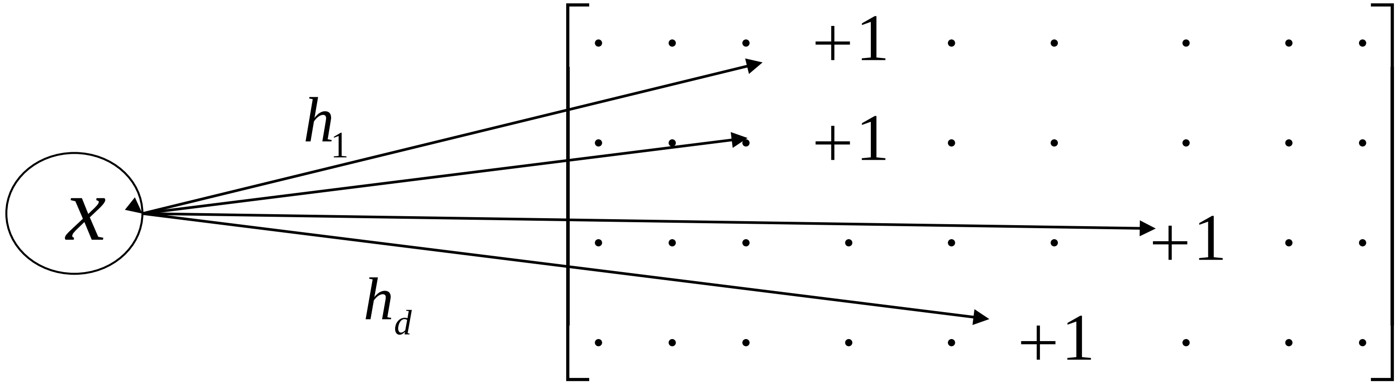
Property: for any $x \neq y$, $\Pr[h_j(x)=h_j(y)] \leq 1/w$

Updating the sketch

Update procedure :

When item x arrives, set $\forall 1 \le j \le d$

$$count[j, h_j(x)] \leftarrow count[j, h_j(x)] + 1$$



When item x is deleted, do the same except changing +1 to -1

Estimating the count of *x*

$$Q(x) \Rightarrow \hat{a}_x = \min_j count[j, h_j(x)]$$

actual count   estimated count

Theorem 1 $a_x \leq \hat{a}_x$

$$\Pr[\hat{a}_x > a_x + \varepsilon n] \leq \delta$$

Proof

We introduce indicator variables

$$I_{x,y,j} = \begin{cases} 1 & \text{if } (x \neq y) \wedge (h_j(x) = h_j(y)) \\ 0 & \text{otherwise} \end{cases}$$

$$E(I_{x,y,j}) = \Pr[h_j(x) = h_j(y)] \leq \frac{1}{w} = \frac{\varepsilon}{2}$$

Define the variable $\quad I_{x,j} = \sum_y I_{x,y,j} a_y$

By construction,

$$count[j, h_j(x)] = a_x + I_{x,j} \Longrightarrow \min count[j, h_j(i)] \geq a_i$$

For the other direction, observe that

$$E(I_{x,j}) = E\left(\sum_y I_{x,y,j} a_y\right) = \sum_y a_y E(I_{x,y,j}) \leq n\varepsilon / 2$$

$$\Pr[\hat{a}_x > a_x + \varepsilon n] = \Pr[\forall j, count[j, h_j(x)] > a_x + \varepsilon n]$$

$$= \Pr[\forall j, a_x + I_{x,j} > a_x + \varepsilon n]$$

$$= \Pr[\forall j, I_{x,j} > 2E(I_{x,j})] < 2^{-d} \leq \delta$$

Markov

$$\Pr[X \geq t] \leq \frac{E(X)}{t} \quad \forall t > 0$$

So, the Count-Min Sketch has size $O\left(\frac{1}{\varepsilon} \log \frac{1}{\delta}\right)$ ■

Technique Four: Tail Bounds

Estimating Self-Join Size

- Given a sequence of items:

  A A B C D B A A B B A A A A A A C C C D A B A A A
  1 1 2 3 4 2 1 1 2 2 1 1 1 1 1 1 3 3 3 4 1 2 1 1 1

- Let x_i be the frequency of item I

- The self-join size is $F_2 = \sum_i x_i^2$

Solution: The AMS sketch [Alon-Matias-Szegedy'96]

- Let $h(i)$ be a 4-wise independent hash function such that

$$\Pr[h(i) = 1] = \Pr[h(i) = -1] = 1/2$$

- The algorithm maintains

$$Z = \sum_i h(i) x_i \qquad \text{(how to maintain?)}$$

- Algorithm returns $Y = Z^2$

- Claim: Y approximates F_2 “well”

Analysis

- The expectation of $Z^2 = (\Sigma_i\, h(i)\, x_i\,)^2$ is equal to $E[Z^2] = E[\Sigma_{i,j}\, h(i)x_i h(j)x_j] = \Sigma_{i,j}\, x_i\, x_j\, E[h(i)h(j)]$

- We have
  - For $i \neq j$, $E[h(i)h(j)] = 0$
  - For $i = j$, $E[h(i)h(j)] = 1$

- Therefore

$$E[Z^2] = \Sigma_i\, x_i^2 = F_2$$

(unbiased estimator)

Need to bound variance

- $\mathrm{Var}[Y] = E[Z^4] - E^2[Z^2]$

$$= \Sigma_i x_i^4 + 6\,\Sigma_{i<j} x_i^2\, x_j^2 - (\Sigma_i x_i^2)^2$$
$$= \Sigma_i x_i^4 + 6\,\Sigma_{i<j} x_i^2\, x_j^2 - \Sigma_i x_i^4 - 2\,\Sigma_{i<j} x_i^2\, x_j^2$$
$$= 4\,\Sigma_{i<j} x_i^2\, x_j^2$$
$$\leq 2\,(\Sigma_i x_i^2)^2$$
$$= 2\,E^2[Y]$$

  $\sigma = O(E[Y])$

- Now use Chebyshev inequality:

$$\Pr[\ |E[Y]-Y| \geq c\sigma\ ] \leq 1/c^2$$

  a constant approximation with a constant probability

- How to boost?

Reduce error: Taking average

- Run k independent instances, yielding $Y_1, \ldots, Y_k$ and return $B = (Y_1 + \ldots + Y_k) / k$

- $E[B] = (E[Y_1] + \ldots + E[Y_k]) / k = E[Y]$
- $Var[B] = (Var[Y_1] + \ldots + Var[Y_k]) / k^2 = Var[Y] / k$
  $\sigma = E[Y] / k^{1/2}$

- We choose $k = O(1/\varepsilon^2)$, then $\sigma = 2\, \varepsilon\, E[Y]$

- Chebyshev gives a $(1+ \varepsilon)$-approximation with constant probability, say 3/4

Boost probability: Taking median

- Run $m = O(\log(1/\delta))$ independent copies of the previous algorithm: $B_1, \ldots, B_m$
- Take the median
- For the median to be outside of the range

$$((1-\varepsilon)\, E[Y], (1+\varepsilon)\, E[Y]),$$

  at least half of $B_1, \ldots, B_m$ have to be wrong.
- But in expectation, at most ¼ of them should be wrong.
- By the Chernoff inequality, this happens with probability $2^{\Theta(m)} = \delta$
- The sketch has total size $O(1/\varepsilon^2 \log(1/\delta))$

Chernoff inequality

- Let $X_1, \ldots, X_n$ be independent Bernoulli random variables, each having probability $p > 1/2$. Then the probability of simultaneous occurrence of more than $n/2$ of the events $\{X_k = 1\}$ has an exact value P, where

$$P = \sum_{i=\lfloor \frac{n}{2} \rfloor + 1}^{n} \binom{n}{i} p^i (1-p)^{n-i}.$$

The Chernoff bound shows that P has the following lower bound:

$$P \geq 1 - e^{-2n\left(p-\frac{1}{2}\right)^2}.$$