IIT KHARAGPUR

IIT GANDHINAGAR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

Scalable Data Science

Lecture 9: Frequent Elements

Anirban Dasgupta

Computer Science and Engineering

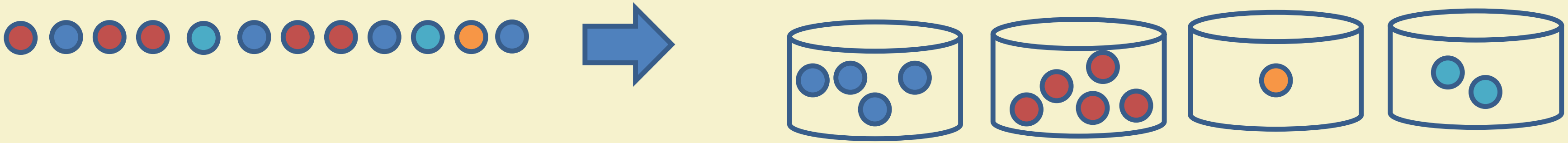
IIT GANDHINAGAR

Streaming model revisited

- Data is seen as incoming sequence
  - can be just element-ids, or ids +frequency updates
- Arrival only streams
- Arrival + departure
  - Negative updates to frequencies possible
  - Can represent fluctuating quantities, e.g.

Frequency Estimation

- Given the input stream, answer queries about item frequencies at the end
  - Useful in many practical applications e.g. finding most popular pages from website logs, detecting DoD attacks, database optimization



- Also used as subroutine in many problems
  - Entropy estimation, itemset mining etc

[Slides courtesy of Graham Cormode]

Frequency estimation

Q1. Can we create a data structure, sketch, sublinear in the data size to answer all frequency queries accurately?

Frequency estimation in one pass

Q1. Can we create a data structure, sketch, sublinear in the data size to answer all frequency queries accurately?

- No

Q2. Can we create a sketch to estimate frequencies of the “most frequent” elements exactly?

Frequency estimation in one pass

Q1. Can we create a data structure, sketch, sublinear in the data size to answer all frequency queries exactly?

- No

Q2. Can we create a sketch to answer frequencies of the “most frequent” elements exactly?

- No

Q3. Sketch to estimate frequencies of “most frequent” elements approximately?

Frequency estimation in one pass

Q1. Can we create a data structure, sketch, sublinear in the data size to answer all frequency queries exactly?

- No

Q2. Can we create a sketch to answer frequencies of the “most frequent” elements exactly?

- No

Q3. Sketch to estimate frequencies of “most frequent” elements approximately?

- YES!

Approximate Heavy Hitters

- Given an update stream of length m, find out all elements that occur "frequently"
  - e.g. at least 1% of the time
  - cannot be done in sublinear space, one pass
- Find out elements that occur at least ϕm times, and none that appears $< (\phi - \epsilon)m$ times
  - Error ϵ
  - Related question: estimate each frequency with error $\pm\epsilon m$

Starting with a puzzle

[J. Algorithms, 1981] Suppose we have a list of N numbers, representing votes of N processors on result of some computation. We wish to decide if there is a majority vote and what that vote is.

- By J.S. Moore
- Did not talk about streaming solution, but proposed solution is
- Strict majority: >N/2

Majority Algorithm

- Arrivals only model
- Start with a counter set to zero
- For each item
  - if counter = 0, pick new item and increment counter
  - else if new item is same as item in hand, increment counter
  - else decrement counter

Majority Algorithm

- Start with a counter set to zero
- For each item
  - if counter = 0, pick new item and increment counter
  - else if new item is same as item in hand, increment counter
  - else decrement counter
- If there is a majority item, it is in hand at the end
- Proof: Since majority occurs > N/2 times, not all occurrences can be cancelled out

Frequent [Misra-Gries]

- Keep k counters and items in hand

Initialize:

- Set all counters to 0

Process(x)

- if x is same as any item in hand, increment its counter
- else if number of items $< k$, store x with counter $= 1$
- else drop x and decrement all counters

Query(q)

- If q is in hand return its counter, else 0

Frequent

- f_x be the true frequency of element x
- At the end, some set of elements is stored with counter values
- If $query\ y$ in hand, $\widehat{f_y} =$ counter value, else $\widehat{f_y} = 0$

Example

Theoretical Bound

Claim: No element with frequency $> m/k$ is missed at the end

Theoretical Bound

Claim: No element with frequency $> m/k$ is missed at the end

Intuition: Each decrement (including drop) is charged with k arrivals. Therefore, will have some copy of an item with frequency $> m/k$

Stronger Claim

Choose $k = \frac{1}{\epsilon}$. For every item x, with frequency f_x the algo can return an estimate $\widehat{f_x}$ such that

$$f_x - \epsilon m \leq \widehat{f_x} \leq f_x$$

Stronger Claim

Choose $k = \frac{1}{\epsilon}$. For every item x, with frequency f_x the algo can return an estimate $\widehat{f_x}$ such that

$$f_x - \epsilon m \leq \widehat{f_x} \leq f_x$$

Same intuition, whenever we drop a copy of item x, we also drop $k - 1$ copies of other items

Summary

- Simple deterministic algorithm to estimate heavy hitters
  – Works only in the arrival model
- Proposed in 1982, rediscovered multiple times with modifications
- Also basis of matrix low rank approximation
- Our next lecture will discuss other algorithms

References:

- Primary references for this lecture
  - Lecture slides by Graham Cormode http://dmac.rutgers.edu/Workshops/WGUnifyingTheory/Slides/cormode.pdf
  - Lecture notes by Amit Chakrabarti: http://www.cs.dartmouth.edu/~ac/Teach/data-streams-lecnotes.pdf
  - Sketch techniques for approximate query processing, Graham Cormode. http://dimacs.rutgers.edu/~graham/pubs/papers/sk.pdf

Thank You!!