DSC 190: Algorithms for Data Science

Arya Mazumdar

University of California San Diego

Winter 2021
Lecture 13
Data Streaming
Counting distinct items

- \( \text{stream } a_1, a_2, \ldots, a_n \rightarrow \text{Algorithm (low space)} \rightarrow \text{some output} \)

- \( a_1, a_2, \ldots, a_n \) a sequence of \( n \) elements
- Each \( a_i \) chosen from a set \( \Sigma \) of size \( m \)
- Say, \( \Sigma \equiv \{1, 2, 3, \ldots, m\} \)
- Find the number of distinct items in the stream
- Easy to do in \( O(m) \) space (store a \( m \) bit vector that records the elements appearing)
- Easy to do in \( O(n \log m) \) space (store each of the distinct elements seen)
- \( m \) and \( n \) are very large
Count-distinct: Intuition

- Let $S$ be the set of distinct items in the sequence.
- We want to estimate $|S|$.
- Suppose (this is just a supposition) $S$ were randomly chosen from \{1, 2, \ldots, m\}.
- Let $\text{min}$ be the minimum element in the stream (in $S$).
- What is the expected value of $\text{min}$?

$$|S| = \frac{m + 1}{\mathbb{E}[\text{min}]} - 1$$

$$|S| \approx \frac{m}{\text{min}}$$
Count-distinct: Intuition

\[ |S| = \frac{m + 1}{\mathbb{E}[\text{min}]} - 1 \]

If we keep an estimate of minimum of the stream, then we can calculate \(|S|\).

We can maintain min of the stream with only \(\log m\) memory.

Con: This method works for a random \(S\).

\[ m = 10^{12} \]
\[ n = 10^6 \]
\[ \log m = 12. \]
Count-distinct: Algorithm

- Instead of tracking the minimum of values of the stream ...
- We hash the values, with a random hash \( h : \Sigma \rightarrow \{0, 1, \ldots, m - 1\} \) and ...
- Track the minimum of the hashed values

Con: We have to store the hash table. This will take \( m \log m \) space
Count-distinct: pairwise independent hashes

Derandomize the hashing:

- We randomly and uniformly select two integers $a$ and $b$ from $\{0, 1, \ldots, m - 1\}$
- We store only these two integers (storage $2 \log m$ bits)
- Define $h : \Sigma \rightarrow \{0, 1, \ldots, m - 1\}$ as
  
  $$h(x) = ax + b \mod m$$
Count-distinct: pairwise independent hashes

Properties of the hash:

- Uniformity: For an element $x$ in the stream
  \[ \Pr(h(x) = s) = \Pr(ax + b \mod m = s) = \Pr(b = s - ax \mod m) = \frac{1}{m} \]

- Pairwise independence: For any two distinct elements $x, y$ in the stream
  \[ \Pr(h(x) = s, h(y) = t) = \frac{\Pr(h(x) = s) \Pr(h(y) = t)}{\Pr([a, b]^T = [x, 1]^{-1} [s, t]^T \mod m)} \]
  \[ = \frac{1}{m^2} \quad \left[ \begin{array}{c} x \\ y \end{array} \right] \left[ \begin{array}{c} a \\ b \end{array} \right] = \left[ \begin{array}{c} s \\ t \end{array} \right] \mod m \quad \frac{ax + b = s}{\mod m} \quad ay + b = t \quad \mod m \]
This is called the Flajolet-Martin algorithm
Randomly select and store two integers \( a \) and \( b \) from 
\( \{0, 1, \ldots, m - 1\} \)

\[
h(x) = ax + b \mod m
\]

For each element \( x \) in the stream
- Compute \( h(x) \)
- Track the minimum of the hash values (log \( m \) space)
- Declare \( |\hat{S}| = \frac{m}{\min} \)
Count-distinct: Analysis

Claim: With probability at least \( \frac{2}{3} \) we have

\[
\left| S \right| \leq \frac{m}{\min} \leq 6 \left| S \right|
\]

\[
\Pr(\frac{m}{\min} > 6\left| S \right|) = \Pr(\min \leq \frac{m}{6\left| S \right|}) = \Pr(\exists \text{ an element } x \text{ for which } h(x) < \frac{m}{6\left| S \right|}) \leq \left| S \right| \Pr(h(x) < \frac{m}{6\left| S \right|}) \leq \frac{\left| S \right|}{\frac{m}{6\left| S \right|}} = \frac{1}{6}
\]

estimate

6 - approximation

\[
\frac{m}{\min} > 6\left| S \right|
\]

\[
\frac{m}{\min} \leq \frac{1\left| S \right|}{6}
\]

\[
< 33\%
\]

Error 1:

Error 2:
Count-distinct: Analysis

Claim: With probability at least $\frac{2}{3}$ we have $\frac{|S|}{6} \leq \frac{m}{\text{min}} \leq 6|S|

- Let the distinct elements be $x_1, x_2, \ldots, x_{|S|}$

$$
\Pr\left( \frac{m}{\text{min}} < \frac{|S|}{6} \right) = \Pr\left( \min > \frac{6m}{|S|} \right) = \Pr(\forall k = 1, 2, \ldots, |S|, h(x_k) > \frac{6m}{|S|})
$$

- Define the indicator random variables $\chi_1, \chi_2, \ldots, \chi_{|S|}$ where

$$
\chi_k = \begin{cases} 
0, & h(x_k) > \frac{6m}{|S|} \\
1, & \text{o.w.}
\end{cases}
$$
Count-distinct: Analysis

Claim: With probability at least $\frac{2}{3}$ we have $\frac{|S|}{6} \leq \frac{m}{\min} \leq 6|S|

- Define the indicator random variables $\chi_1, \chi_2, \ldots, \chi_{|S|}$ where

$$\chi_k = \begin{cases} 0, & h(x_k) > \frac{6m}{|S|} \\ 1, & \text{o.w.} \end{cases}$$

- $\mathbb{E} \chi_k = \Pr(h(x_k) \leq \frac{6m}{|S|}) = \frac{6}{|S|}$
- $\var \chi_k = \frac{6}{|S|} \left(1 - \frac{6}{|S|}\right) < \frac{6}{|S|}$
- Let $X = \sum_k \chi_k$
- $\mathbb{E} X = |S| \cdot \frac{6}{|S|} = 6$

$$\Pr(X = 0) \quad \mathbb{E} X = 6$$

$$\mathbb{E} X = \sum_k \mathbb{E} \chi_k$$

$$= \sum_k \Pr(h(x_k) \leq \frac{6m}{|S|})$$

$$= \sum_k \frac{6 \mu_k \cdot \frac{1}{|S|}}{\mu_k}$$

$$= 18 \cdot \frac{6}{18} = 6$$
Count-distinct: Analysis

\[ \text{Var} X = \mathbb{E}(X - \mathbb{E}X)^2 = \mathbb{E}\left(\sum_k (X_k - \mathbb{E}X_k)^2\right) \]

\[ = \sum_k (X_k - \mathbb{E}X_k)^2 + 2 \sum_k \sum_l \mathbb{E}(X_k - \mathbb{E}X_k)(X_l - \mathbb{E}X_l) \]

\[ = \sum_k \text{Var} X_k + 2 \sum_k \sum_l \mathbb{E}(X_k - \mathbb{E}X_k)\mathbb{E}(X_l - \mathbb{E}X_l) \]

\[ \leq \frac{6}{|S|} |S| = 6 \]

\[ \Pr \{X = 0\} \quad \text{if} \quad \mathbb{E}X = 6 \]

\[ \text{Var} X \leq 6 \]
Count-distinct: Analysis

\[
\Pr\left( \frac{m}{\min} < \frac{|S|}{6} \right) = \Pr\left( \forall k = 1, 2, \ldots, |S|, h(x_k) > \frac{6m}{|S|} \right) = \Pr(X = 0)
\]

\[
\leq \Pr(|X - \mathbb{E}X| \leq 6) \leq \frac{\text{Var} X}{36} \leq \frac{6}{36} = \frac{1}{6}
\]

With probability at least \( \frac{2}{3} \) we have \( \frac{|S|}{6} \leq \frac{m}{\min} \leq 6|S| \)
The Heavy Hitters

- $A = a_1, a_2, \ldots, a_n$ a sequence of $n$ elements
- Each $a_i$ chosen from a set $\Sigma$ of size $m$
- Given a parameter $k$:
  - Find all elements that appear at least $\frac{n}{k}$ times
The Heavy Hitters: Applications

- Computing popular products. $A$ could be all of the page views of products on amazon.com yesterday. The heavy hitters correspond to frequently viewed items.

- Computing frequent search queries. For example, $A$ could be all of the searches on Google yesterday. The heavy hitters are then searches made most often.

- Identifying heavy TCP flows. Here, $A$ is a list of data packets passing through a network switch, each annotated with a source-destination pair of IP addresses. The heavy hitters are then the flows that are sending the most traffic. This is useful for, among other things, to identify denial-of-service attacks.

- Identifying volatile stocks. Here, $A$ is a list of stock trades.

Special Case: Majority

- An array $A = a_1, a_2, \ldots, a_n$ a sequence of $n$ elements with the promise that it has a majority element - a value that is repeated strictly more than $n/2$ times.
- Find that element
- Find that element in linear time
- Find the Majority element in linear time in a single left to right pass in “small” space
Majority: Algorithm

- Set count = 1, current = $a_1$
- For $i = 2, 3, \ldots$
  1. If count == 0 set current = $a_i$ and count = 1
  2. Else If $a_i$ == current then count = count + 1
     Else count = count -1
- Return current

If there is an majority element, that will be returned. Why?
Heavy Hitters: Approximate Version

\[ \frac{n}{10} \text{ times} : \quad \text{list} = \frac{n}{2d} \text{ times} \]

- \( A = a_1, a_2, \ldots, a_n \) a sequence of \( n \) elements
- Given a parameter \( k \):
  - Find all elements that appear at least \( \frac{n}{k} - \epsilon n \) times

Why can we not set \( \epsilon = 0 \)?

The space requirement is growing with \( \frac{1}{\epsilon} \)

If we take \( \epsilon = \frac{1}{2k} \), space usage is \( \tilde{O}(k) \), all elements with frequency \( \frac{n}{k} \) is in the list and the elements in the list have frequency at least

\[
\frac{n}{k} - \frac{n}{2k} = \frac{n}{2k}
\]
Bigger problem: Estimating frequencies

\( a \ b \ b \ c \ d \ a \cdot b \ a \)

\( f_a = 3 \quad f_b = 3 \quad f_c = 1 \quad f_d = 1 \)

- \( A = a_1, a_2, \ldots, a_n \) a sequence of \( n \) elements
- Each \( a_i \) chosen from a set \( \Sigma \) of size \( m \)
- Frequency of an item \( f_j = |\{i : a_i = j, i = 1, 2, \ldots, n\}| \)

- Point Query: For \( j \in \Sigma \), find \( f_j \)
- Heavy Hitters: Find all elements \( j \) such that \( f_j \geq \phi n \) for a given \( \phi \in [0, 1] \)

Solution: Count-Min Sketch
Count-Min Sketch

\[ \frac{n}{\varepsilon} - \frac{\varepsilon n}{\delta} \]

- Select an \( \varepsilon > 0 \) and \( \delta > 0 \): \( \varepsilon \) denotes the error-parameter, and \( \delta \) denotes our confidence.
- Select \( d = \ln \frac{1}{\delta} \) hash functions \( h_1, h_2, \ldots, h_d \) independently and randomly from a pair-wise independent hash family. Each \( h_i : \{1, 2, \ldots, m\} \rightarrow \{1, 2, \ldots, w\} \) where \( w = \frac{e}{\varepsilon} \).
- Initialize a table \( T \) of dimension \( d \times w \) all with 0.
- Update: At time \( t \), when \( a_t \) arrives, do the following.
  \[ T(1, h_1(a_t)) = T(1, h_1(a_t)) + 1 \]
  \[ T(2, h_2(a_t)) = T(2, h_2(a_t)) + 1 \]
  .
  .
  .
  \[ T(d, h_d(a_t)) = T(d, h_d(a_t)) + 1 \]

http://research.neustar.biz/tag/count-min-sketch/

\[ O\left(\frac{1}{\varepsilon} \log \frac{1}{\delta}\right) \]

\[ d \times w = \frac{e}{\varepsilon} \log \frac{1}{\delta} \]
Count-Min Sketch: Point Query

- **Problem** For $i \in [m]$, estimate $f_i$
- **Output** An estimate $\hat{f}_i$ such that $f_i \leq \hat{f}_i \leq f_i + \epsilon n$
- **Algorithm** Construct Count-Min sketch. Return

$$\min_{l=1,\ldots,d} T(l, h_l(i))$$

$h_1(i) \quad h_2(i) \quad h_3(i) \quad \ldots \quad h_d(i)$