Streaming Algorithm: Filtering & Counting Distinct Elements

CompSci 590.02
Instructor: AshwinMachanavajjhala
Streaming Databases

Continuous/Standing Queries: Every time a new data item enters the system, (conceptually) re-evaluate the answer to the query.

Can't hope to process a query on the entire data, but only on a small working set.
Examples of Streaming Data

• Internet & Web traffic
  – Search/browsing history of users: Want to predict which ads/content to show the user based on their history.
    Can’t look at the entire history at runtime

• Continuous Monitoring
  – 6 million surveillance cameras in London
  – Video feeds from these cameras must be processed in real time

• Weather monitoring

• ...

Lecture 6 : 590.02 Spring 13
Processing Streams

• Summarization
  – Maintain a small size sketch (or summary) of the stream
  – Answering queries using the sketch
  – E.g., random sample
  – later in the course – AMS, count min sketch, etc
  – Types of queries: # distinct elements, most frequent elements in the stream, aggregates like sum, min, max, etc.

• Window Queries
  – Queries over a recent k size window of the stream
  – Types of queries: alert if there is a burst of traffic in the last 1 minute, denial of service identification, alert if stock price > 100, etc.
Streaming Algorithms

• Sampling
  – We have already seen this.

• Filtering
  – “… does the incoming email address appear in a set of white listed addresses…”

• Counting Distinct Elements
  – “… how many unique users visit cnn.com…”

• Heavy Hitters
  – “… news articles contributing to >1% of all traffic…”

• Online Aggregation
  – “… Based on seeing 50% of the data the answer is in [25,35]…”
Streaming Algorithms

- **Sampling**
  - We have already seen this.

- **Filtering**
  - “… does the incoming email address appear in a set of white listed addresses …”

- **Counting Distinct Elements**
  - “… how many unique users visit cnn.com …”

- **Heavy Hitters**
  - “… news articles contributing to >1% of all traffic …”

- **Online Aggregation**
  - “… Based on seeing 50% of the data the answer is in [25,35] …”
FILTERING
Problem

- A set $S$ containing $m$ values
  - A whitelist of a billion non-spam email addresses

- Memory with $n$ bits.
  - Say 1 GB memory

- Goal: Construct a data structure that can efficiently check whether a new element is in $S$
  - Returns TRUE with probability 1, when element is in $S$
  - Returns FALSE with high probability $(1-\varepsilon)$, when element is not in $S$
Bloom Filter

• Consider a set of hash functions \( \{h_1, h_2, \ldots, h_k\}, h_i : S \to [1, n] \)

Initialization:
• Set all \( n \) bits in the memory to 0.

Insert a new element ‘a’:
• Compute \( h_1(a), h_2(a), \ldots, h_k(a) \). Set the corresponding bits to 1.

Check whether an element ‘a’ is in S:
• Compute \( h_1(a), h_2(a), \ldots, h_k(a) \).
  If all the bits are 1, return TRUE.
  Else, return FALSE
Analysis

If \( a \) is in \( S \):

- If \( h_1(a), h_2(a), \ldots, h_k(a) \) are all set to 1.
- Therefore, Bloom filter returns TRUE with probability 1.

If \( a \) not in \( S \):

- Bloom filter returns TRUE if each \( h_i(a) \) is 1 due to some other element

\[
\Pr[\text{bit } j \text{ is 1 after } m \text{ insertions}] = 1 - \Pr[\text{bit } j \text{ is 0 after } m \text{ insertions}]
\]
\[
= 1 - \Pr[\text{bit } j \text{ was not set by } k \times m \text{ hash functions}]
\]
\[
= 1 - (1 - 1/n)^{km}
\]

\[
\Pr[\text{Bloom filter returns TRUE}] = \{1 - (1 - 1/n)^{km}\}^k \approx (1 - e^{-km/n})^k
\]
Example

- Suppose there are $m = 10^9$ emails in the white list.
- Suppose memory size of 1 GB ($8 \times 10^9$ bits)

$k = 1$
- $\Pr[\text{Bloom filter returns TRUE | a not in S}] = 1 - e^{-m/n}$
  \[= 1 - e^{-1/8} = 0.1175\]

$k = 2$
- $\Pr[\text{Bloom filter returns TRUE | a not in S}] = (1 - e^{-2m/n})^2$
  \[= (1 - e^{-1/4})^2 \approx 0.0493\]
Example

- Suppose there are $m = 10^9$ emails in the white list.
- Suppose memory size of 1 GB ($8 \times 10^9$ bits)

Exercise:
What is the optimal number of hash functions given $m=|S|$ and $n$. 
Summary of Bloom Filters

• Given a large set of elements $S$, efficiently check whether a new element is in the set.

• Bloom filters use hash functions to check membership
  – If $a$ is in $S$, return TRUE with probability 1
  – If $a$ is not in $S$, return FALSE with high probability
  – False positive error depends on $|S|$, number of bits in the memory and number of hash functions
COUNTING DISTINCT ELEMENTS
Distinct Elements

INPUT:
• A stream $S$ of elements from a domain $D$
  – A stream of logins to a website
  – A stream of URLs browsed by a user
• Memory with $n$ bits

OUTPUT
• An estimate of the number of distinct elements in the stream
  – Number of distinct users logging in to the website
  – Number of distinct URLs browsed by the user
FM-sketch

• Consider a hash function \( h: D \rightarrow \{0,1\}^L \) which uniformly hashes elements in the stream to \( L \) bit values.

• IDEA: The more distinct elements in \( S \), the more distinct hash values are observed.

• Define: \( \text{Tail}_0(h(x)) = \text{number of trailing consecutive 0's} \)
  - \( \text{Tail}_0(101001) = 0 \)
  - \( \text{Tail}_0(101010) = 1 \)
  - \( \text{Tail}_0(001100) = 2 \)
  - \( \text{Tail}_0(101000) = 3 \)
  - \( \text{Tail}_0(000000) = 6 = L \)
FM-sketch

**Algorithm**

- For all $x \in S$,
  - Compute $k(x) = \text{Tail}_0(h(x))$
- Let $K = \max_{x \in S} k(x)$
- Return $F' = 2^K$
Analysis

Lemma: \( \Pr[ \text{Tail}_0(h(x)) \geq j ] = 2^{-j} \)

Proof:

• \( \text{Tail}_0(h(x)) \geq j \) implies at least the last \( j \) bits are 0

• Since elements are hashed to \( L \)-bit string uniformly at random, the probability is \( (\frac{1}{2})^j = 2^{-j} \)
Analysis

• Let $F$ be the true count of distinct elements, and let $c>2$ be some integer.

• Let $k_1$ be the largest $k$ such that $2^k < cF$

• Let $k_2$ be the smallest $k$ such that $2^k > F/c$

• If $K$ (returned by FM-sketch) is between $k_2$ and $k_1$, then
  \[ \frac{F}{c} \leq F' \leq cF \]
Analysis

• Let $z_x(k) = 1$ if $\text{Tail}_0(h(x)) \geq k$
  = 0 otherwise
• $E[z_x(k)] = 2^{-k}$, $\text{Var}(z_x(k)) = 2^{-k}(1 - 2^{-k})$

• Let $X(k) = \sum_{x \in S} z_x(k)$

• We are done if we show with high probability that
  $X(k_1) = 0$ and $X(k_2) \neq 0$
Analysis

Lemma: \( \Pr[X(k_1) \geq 1] \leq 1/c \)

Proof: \( \Pr[X(k_1) \geq 1] \leq E(X(k_1)) \) \[ \text{Markov Inequality} \]
\[ = F \cdot 2^{-k_1} \leq 1/c \]

Lemma: \( \Pr[X(k_2) = 0] \leq 1/c \)

Proof: \( \Pr[X(k_2) = 0] = \Pr[X(k2) - E(X(k2)) = E(X(k2))] \)
\[ \leq \Pr[|X(k2) - E(X(k2))| \geq E(X(k2))] \]
\[ \leq \frac{\text{Var}(X(k2))}{E(X(k2))^2} \] \[ \text{Chebyshev Ineq.} \]
\[ \leq 2^{k_2}/F \leq 1/c \]

Theorem: If FM-sketch returns \( F' \), then for all \( c > 2, \)
\( F/c \leq F' \leq cF \) with probability \( 1 - 2/c \)
Boosting the success probability

• Construct \( s \) independent FM-sketches (\( F'_1, F'_2, \ldots, F'_s \))
• Return the median \( F'_{\text{med}} \)

Q: For any \( \delta \), what is the value of \( s \) s.t. \( P[F/c \leq F'_{\text{med}} \leq cF] > 1 - \delta \)?
Analysis

• Let $c > 4$, and $x_i = 0$ if $F/c \leq F'_i \leq cF$, and 1 otherwise

• $\rho = E[x_i]$
  
  $\rho = 1 - \Pr[F/c \leq F'_i \leq cF] \leq 2/c < \frac{1}{2}$

• Let $X = \sum_i x_i$, $E(X) = \rho$

Lemma: If $X < s/2$, then $F/c \leq F'_{\text{med}} \leq cF$ \hspace{1cm} (Exercise)

We are done if we show that $\Pr[X \geq s/2]$ is small.
Analysis

\[ \Pr[ X \geq s/2 ] = \Pr[ X - E(X) = s/2 - E(X) ] \]
\[ \leq \Pr[ |X - E(X)| \geq s/2 - sp ] \]
\[ = \Pr[ |X - E(X)| \geq (1/2\rho - 1) sp ] \]
\[ \leq 2\exp( -(1/2\rho - 1)^2 sp/3 ) \quad \text{Chernoff bounds} \]

Thus, to bound this probability by \( \delta \), we need \( s \) to be:

\[ s \geq \frac{3\rho}{(1/2 - \rho)^2} \ln\left(\frac{2}{\delta}\right) \]
Boosting the success probability

In practice,

• Construct sk independent FM sketches
• Divide the sketches into s groups of k each
• Compute the mean estimate in each group
• Return the median of the means.
Summary

• Counting the number of distinct elements exactly takes $O(N)$ space and $\Omega(N)$ time, where $N$ is the number of distinct elements.

• FM-sketch estimates the number of distinct elements in $O(\log N)$ space and $\Theta(N)$ time.

• FM-sketch: maximum number of trailing 0s in any hash value.

• Can get good estimates with high probability by computing the median of many independent FM-sketches.