Persistent Data Sketching

Zhewei Wei
Renmin University of China

Ge Luo
The Hong Kong University of Science and Technology

Ke Yi
The Hong Kong University of Science and Technology

Xiaoyong Du
Renmin University of China

Ji-Rong Wen
Renmin University of China
Streaming Algorithms

- A data stream is a (massive) sequence of data.
Streaming Algorithms

- A data stream is a (massive) sequence of data
  - Single Pass: Each record is examined at most once

Data Stream → Stream Processing Engine → (Approximate) Answer query → Summary in Memory
Streaming Algorithms

- A data stream is a (massive) sequence of data
  - Single Pass: Each record is examined at most once
  - Small Space: Log or polylog in data stream size

Summary in Memory

Data Stream

Stream Processing Engine

(Approximate) Answer query
Streaming Algorithms

- A data stream is a (massive) sequence of data
  - **Single Pass:** Each record is examined at most once
  - **Small Space:** Log or polylog in data stream size
  - **Small time:** Low per-record processing time ($O(1)$ to polylog $N$)

Diagram:

- Data Stream
- **Summary in Memory**
- Stream Processing Engine
- (Approximate) Answer query
Sketches

- Sub-linear space
  - Fast update and query time

- Answer queries approximately

- Linear transformation of the data frequencies
Sketches

• Count-Min Sketch [Cormode and Muthukrishnan 2005]
  - Point queries, heavy hitters (frequent items)

• AMS Sketch [Alon et. al. 1999]
  - Frequency moments

• Count Sketch [Charikar et. al. 2002]
  - Join size queries, self join size queries [Rusu and Dobra 2007]

• ...

Sketches

- Sub-linear space
  - Fast update and query time

- Answer queries approximately

- Linear transformation of the data frequencies
Sketches

• Sub-linear space
  - Fast update and query time

• Answer queries approximately

• Linear transformation of the data frequencies

• Ephemereral
  - Answer queries on current version of data stream
Query Back in Time

- The ability to query on historical data is necessary for analyzing trends & change pattern of data
Persistent Database/ Data Structure

• Answer queries on the past version of the database
Persistent Database/Data Structure

- Answer queries on the past version of the database
Persistent Database/Data Structure

• Answer queries on the past version of the database


• Microsoft Immortal DB [Lomet et. al. 2005], SNAP [Shrira and Xu 2005], Ganymed [Plattner et. al. 2006], Skippy [Shaull et. al. 2008] and LIVE[Sarma et. al. 2010]
Persistent Database/Data Structure

• Answer queries on the past version of the database


• Microsoft Immortal DB [Lomet et. al. 2005], SNAP [Shrira and Xu 2005], Ganymed [Plattner et. al. 2006], Skippy [Shaull et. al. 2008] and LIVE[Sarma et. al. 2010]

• Space linear in # of updates
  - Large storage
  - Storage on disk (not in streaming setting)
Persistent Database
Query on historical data
Linear space

Sketch
Query on current data
Sub-linear space
Persistent Database
Query on historical data
Linear space

Sketch
Query on current data
Sub-linear space

Persistent Sketch
Query on historical data
Sub-linear space
Persistent Sketch

- Historical window query
Persistent Sketch

- Historical window query

Start time $s$  
End time $t$
Persistent Sketch

- Historical window query

- Given a time interval \((s, t]\), return a sketch for substream \(f(s, t)\)

- What is the top-k/frequency moment/join size of the stream between \(s\) and \(t\)?
High Level Ideas
&
Our Results
Count-Min Sketch
[Cormode and Muthukrishnan 2005]

- Given an error parameter $\varepsilon$

- Choose a hash function $h: [n] \rightarrow [2/\varepsilon]$ and build a hash table of size $2/\varepsilon$
Count-Min Sketch
[Cormode and Muthukrishnan 2005]

- Given an error parameter $\varepsilon$

- Choose a hash function $h: [n] \rightarrow [2/\varepsilon]$ and build a hash table of size $2/\varepsilon$
Count-Min Sketch
[Cormode and Muthukrishnan 2005]

- Given an error parameter $\varepsilon$

- Choose a hash function $h: [n] \rightarrow [2/\varepsilon]$ and build a hash table of size $2/\varepsilon$
Count-Min Sketch
[Cormode and Muthukrishnan 2005]

- Given an error parameter $\varepsilon$

- Choose a hash function $h: [n] \rightarrow [2/\varepsilon]$ and build a hash table of size $2/\varepsilon$

$$C[h(i)] = C[h(i)] + 1$$
Linear Transformation

\[ h(i) \begin{bmatrix} 0, 1, 0, \ldots, 0, \ldots, 0, 0, \\ \vdots \\ 0, 0, 0, \ldots, 1, \ldots, 0, 0, \\ \vdots \\ 0, 0, 0, \ldots, 0, \ldots, 1, 0, \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_i \\ \vdots \\ f_N \end{bmatrix} = C[h(i)] \]
Linear Transformation

Stream

Start time $s$  End time $t$
Linear Transformation

Start time $s$  
End time $t$  
Stream

$C_s$  
$C_t$
Linear Transformation

Let $C_t - C_s$ represent the linear transformation. The diagram illustrates the transformation from start time $s$ to end time $t$. The stream from $C_s$ to $C_t$ is shown, indicating the change over time.
Linear Transformation

\[ C_t - C_s \]

- Linear Space
Linear Transformations

- Linear Space
- Sketch is already an approximation
Baseline Solution

Ephemeral sketch:

\[
C[i]
\]
Baseline Solution

Ephemeral sketch:

Historical Lists:

\[ C[i] - C[i, t_1] \approx \Delta \]

\[ C[i, t_2] - C[i, t_1] \approx \Delta \]

\[ C[i] \text{ at time } t_1 \]

\[ C[i] \text{ at time } t_2 \]

\[ C[i] \text{ at time } t_3 \]
Baseline Solution

Ephemeral sketch:

Historical Lists:

Query time $t$

$$C[i] \approx \Delta$$
Baseline Solution

Ephemeral sketch:

Historical Lists:

Query time $t$
Baseline Solution

- Historical window point/heavy hitters query:
  - What is frequency of “/images/space.gif” between day 34 and day 37
  - What are the mostly requested URLs between day 34 and day 37
Baseline Solution

- Historical window **point/heavy hitters** query:
  - What is frequency of “/images/space.gif” between day 34 and day 37
  - What are the mostly requested URLs between day 34 and day 37

- Error: $\varepsilon \| f(s,t) \|_1$ (ephemeral error) + $\Delta$ (persistent error)
Baseline Solution

• Historical window **point/heavy hitters** query:
  - What is frequency of “/images/space.gif” between day 34 and day 37
  - What are the most requested URLs between day 34 and day 37

• Error: $\varepsilon \| f(s, t) \|_1$ (ephemeral error) + $\Delta$ (persistent error)
Baseline Solution

- Historical window **point/heavy hitters** query:
  - What is frequency of “/images/space.gif” between day 34 and day 37
  - What are the most requested URLs between day 34 and day 37

- Error: \( \varepsilon \|f(s,t)\|_1 \) (ephemeral error) + \( \Delta \) (persistent error)

- Space: proportional to \( (1/\varepsilon + m/\Delta) \)
Baseline Solution

- Historical window point/heavy hitters query:
  - What is frequency of “/images/space.gif” between day 34 and day 37
  - What are the mostly requested URLs between day 34 and day 37

- Error: $\epsilon \| f(s,t) \|_1$ (ephemeral error) + $\Delta$ (persistent error)

- Space: proportional to $(1/\epsilon + m/\Delta)$

- Cannot handle (self) join size queries
Piece-wise Linear Approximation

• Counter changes by at most 1 at each timestamp

• Each counter is a discrete function according to timestamps
PLA-based Persistent Sketch

Ephemeral sketch:

PLA generator:
PLA-based Persistent Sketch

Ephemeral sketch:

PLA generator:

Query time $t$
PLA-based Persistent Sketch

- Historical window point/heavy hitters query:
  - What is frequency of “/images/space.gif” between day 34 and day 37
  - What are the mostly requested URLs between day 34 and day 37

- Error: $\varepsilon \|f(s,t)\|_1$ (ephemeral error) + $\Delta$ (persistent error)
PLA-based Persistent Sketch

• Historical window point/heavy hitters query:
  - What is frequency of “/images/space.gif” between day 34 and day 37
  - What are the mostly requested URLs between day 34 and day 37

• Error: $\varepsilon \| f(s,t) \|_1$ (ephemeral error) + $\Delta$ (persistent error)

• Space: proportional to $(1/\varepsilon + m/\Delta^2)$ in random stream model
Estimating Join Size

- Estimating (self) join size in an ephemeral sketch: $\sum_i C[i]^2$
Estimating Join Size

- Estimating (self) join size in an ephemeral sketch: \( \sum_i C[i]^2 \)
- Estimating (self) join size in a persistent sketch:
Estimating Join Size

• Estimating (self) join size in an ephemeral sketch:
  \[ \sum_i C[i]^2 \]

• Estimating (self) join size in a persistent sketch:
  \[ \sum_i (C[i] + \text{error of } \Delta)^2 \]
Estimating Join Size

• Estimating (self) join size in an ephemeral sketch:
  \[ \sum_i C[i]^2 \]

• Estimating (self) join size in a persistent sketch:
  \[ \sum_i (C[i] + \text{error of } \Delta)^2 \]

• Bias will amplify error significantly
Estimating Join Size

• Estimating (self) join size in an ephemeral sketch:
  \[ \sum_i C[i]^2 \]

• Estimating (self) join size in a persistent sketch:
  \[ \sum_i (C[i] + \text{error of } \Delta)^2 \]

• Bias will amplify error significantly

• Need unbiased estimator of the counter
Sampling Based Persistent Sketch

Ephemeral sketch:

Historical Lists:

$C[i]$ at time $t_1$

$C[i]$ at time $t_2$

$C[i]$ at time $t_3$
Sampling Based Persistent Sketch

Ephemeral sketch:

Historical Lists:

Sample with probability $1/\Delta$

$C[i]$ at time $t_1$

$C[i]$ at time $t_2$

$C[i]$ at time $t_3$

$C[i]$ at time $t_1$
Sampling Based Persistent Sketch

Ephemeral sketch:

Historical Lists:

Query time $t$

$C[i, t_1]$ at time $t_1$
Sample with probability $1/\Delta$

$C[i]$ at time $t_2$

$C[i, t_3]$ at time $t_3$

$C[i, t_3] + \Delta - 1$

$0$
Sampling based AMS Sketch

- Unbiased Estimator

- Historical window join size query:
  - What is the join size of stream 1 and stream 2 between day 34 and day 37

- Error: \( \varepsilon \sqrt{\left( \| f_{s,t} \|_2^2 + \left( \frac{\Delta f}{\varepsilon} \right)^2 \right) \left( \| g_{s,t} \|_2^2 + \left( \frac{\Delta g}{\varepsilon} \right)^2 \right)} \)

- Space: proportional to \( \frac{1}{\varepsilon} + \frac{m}{\Delta} \)
Experimental Study

- 7,000,000 requests from the 1998 World Cup web site access log
- Built sketches on two attributes

**Requested URL**

**IP address of the request**
Experimental Study

Query range (0.2N, 0.6N)

Requested URL

IP address of the request

Point Query

Self Join Size Query

<table>
<thead>
<tr>
<th>Sketch size (log scale)</th>
<th>Absolute error (log scale)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PWC_AMS</td>
<td>PLA</td>
</tr>
<tr>
<td>PWC_CountMin</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sketch size (log scale)</th>
<th>Absolute error (log scale)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PWC_AMS</td>
<td>PLA</td>
</tr>
<tr>
<td>PWC_CountMin</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sketch size (log scale)</th>
<th>Relative error (log scale)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample</td>
<td>PWC_AMS</td>
</tr>
<tr>
<td>PWC_CountMin</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sketch size (log scale)</th>
<th>Relative error (log scale)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample</td>
<td>PWC_AMS</td>
</tr>
<tr>
<td>PWC_CountMin</td>
<td></td>
</tr>
</tbody>
</table>

Point Query range (0.2N, 0.6N)
Conclusion

- Persistent sketch
  - Query on historical data
  - Sub-linear space
- Support point/heavy hitters/join size queries
- Provable error and space bound
- Performs well in practice
Thanks!