Data Sketches

Lecturer: Jiaheng Lu
Autumn 2016
Outline

• Massive data stream
• Bloom filter (this lecture)
• Count-min (this lecture)
• Count-sketch (next lecture)
• FM-sketch (next lecture)
• Data is *continuously growing* faster than our ability to store or index it

```
  records processed one at a time
```

---

**Massive Data Streams**

- Data is *continuously growing* faster than our ability to store or index it.
Massive Data Streams applications

- There are 3 Billion Telephone Calls in US each day, 30 Billion emails daily, 1 Billion SMS

- Scientific data: NASA's observation satellites generate billions of readings per day

- IP Network Traffic: up to 1 Billion packets per hour
Data stream

- **Data stream**: a sequence $A = <a_1, a_2, ..., a_m>$, where the elements of the sequence are drawn from the universe $\{1, 2, ..., n\}$
Sketches for data streams

• Sketch can “see” all the data even if it can’t “remember” it all

• For example, one hash function can be used to answer set membership checking problem
Set Membership checking

- x: Element
- S: Set of elements
- Input: x, S
- Output:
  - True (if x in S)
  - False (if x not in S)
Example of set membership checking

- Does 1 appear in the following set? Yes
- Does 9 appear in the following set? Yes
- Does 10 appear in the following set? No
Hashing for membership checking

Hash function: $x \mod 5$

| 9 | 1 | 4 | 1 | 2 | 1 | 4 | 7 | 6 | 9 |

Bit array: set to 1 if the array position (starting from 0) is equal to “$x \mod 5$”
Hashing for membership checking

Hash function: \( x \mod 5 \)

Bit array: set to 1 if the array position (starting from 0) is equal to “x mod 5”
Hashing for membership checking

Hash function: \( x \mod 5 \)

```
9 1 4 1 2 1 4 7 6 9
```

Bit array: set to 1 if the array position (starting from 0) is equal to “\( x \mod 5 \)”
Hashing for membership checking

Hash function: $x \mod 5$

Bit array: set to 1 if the array position (starting from 0) is equal to “$x \mod 5$”
Hashing for membership checking

Hash function: $x \mod 5$

Does 1 appear in the set? Yes
Does 9 appear in the set? Yes
Does 10 appear in the set? No
Hashing for membership checking

Hash function 1: $x \mod 5$

| 9 | 1 | 4 | 1 | 2 | 1 | 4 | 7 | 6 | 9 |

• Does 11 appear in the set? Yes

It is a false positive!
Bloom Filters

- Bloom filters can reduce the probability of false positive.
- Proposed by Burton Howard Bloom in 1970.
- His idea is to use more than one hash function.
Example of a bloom filter

Hash function 1: \( x \mod 5 \)
Hash function 2: \( (x \mod 8) \mod 5 \)
Example of a bloom filter

Hash function 1: \( x \mod 5 \)
Hash function 2: \((x \mod 8) \mod 5\)
Example of a bloom filter

| 9 | 1 | 4 | 1 | 2 | 1 | 4 | 7 | 6 | 9 |

Hash function 1: $x \mod 5$
Hash function 2: $(x \mod 8) \mod 5$

Final state:  0 1 1 0 1
Example of a bloom filter

| 9 | 1 | 4 | 1 | 2 | 1 | 4 | 7 | 6 | 9 |

Hash function 1: \( x \mod 5 \)
Hash function 2: \((x \mod 8) \mod 5\)

- Does 11 appear in the following set? No

Two hash functions avoid false positive!
Example of a bloom filter

Hash function 1: $x \mod 5$

Hash function 2: $(x \mod 8) \mod 5$

Does 12 appear in the following set? Yes

Unfortunately, it still cannot fully avoid false positive!
Bloom Filters Applications

- Bloom Filters widely used in “big data” applications
  - Many problems require storing a large set of items

- Bloom Filters are still an active research area
  - Often appear in networking conferences
  - Also known as “signature files” in databases
Summary of Bloom Filters

• Given a large set of elements $S$, efficiently check whether a new element is in the set.
  • Bloom filters use multiple 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
Main properties of a sketch

- Queries supported
- Sketch size
- Update speed
- Query time
- Sketch initialization
Main properties of a sketch e.g. Bloom filter

- Queries supported: membership checking
- Sketch size: the length of the bit array
- Update speed: $k$ hash functions
- Query time: $k$ hash functions
- Sketch initialization: the length of the bit array
• Watch a video on bloom filter
Outline

• Massive data stream
• Bloom filter (this lecture)
• Count-min (this lecture)
• Count-sketch (next lecture)
• FM-sketch (next lecture)
Count-Min Sketch

- Problem: Estimating the frequency of each item.

| 9 | 1 | 4 | 1 | 2 | 1 | 4 | 9 |

- 9 appears twice, 1 appears three times, 4 appears twice,
- 2 appears once.
Count-Min Sketch

- Count Min sketch encodes item counts
  - Some similarities to Bloom filters
- Model input data as a matrix
  - Create a small summary as an array of $w \times d$ in size
  - Use $d$ hash function to map vector entries to $[1..w]$
Count-Min Sketch

\[ f_i \]

\[ C_1[h_1(i)] \]

\[ C_2[h_2(i)] \]

\[ \ldots \]

\[ C_d[h_d(i)] \]
Example of count-min

<table>
<thead>
<tr>
<th></th>
<th>9</th>
<th>1</th>
<th>4</th>
<th>1</th>
<th>2</th>
<th>1</th>
<th>4</th>
<th>9</th>
</tr>
</thead>
</table>

Initial

\[
\begin{array}{cccccccc}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{array}
\]

\[
h_1(x) = x \ mod \ 3 \\
h_2(x) = (x \ mod \ 4) \ mod \ 3 \\
h_3(x) = (2*x) \ mod \ 3
\]
Example of count-min

| 9 | 1 | 4 | 1 | 2 | 1 | 4 | 9 |

Initial

| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |

\[
h_1(x) = x \mod 3 \]
\[
h_2(x) = (x \mod 4) \mod 3 \]
\[
h_3(x) = (2*x) \mod 3 \]
Example of count-min

\[
\begin{align*}
\text{h1}(x) &= x \mod 3 \\
\text{h2}(x) &= (x \mod 4) \mod 3 \\
\text{h3}(x) &= (2x) \mod 3
\end{align*}
\]
Example of count-min

Initial

\[
\begin{array}{cccccccc}
9 & 1 & 4 & 1 & 2 & 1 & 4 & 9 \\
\end{array}
\]

\[
h_1(x) = x \mod 3 \\
h_2(x) = (x \mod 4) \mod 3 \\
h_3(x) = (2 \times x) \mod 3
\]
Example of count-min

Initial

<table>
<thead>
<tr>
<th>1</th>
<th>3</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

\[
h_1(x) = x \mod 3 \\
h_2(x) = (x \mod 4) \mod 3 \\
h_3(x) = (2\times x) \mod 3
\]
Example of count-min

\begin{array}{cccccccc}
9 & 1 & 4 & 1 & 2 & 1 & 4 & 9 \\
\end{array}

Final

\begin{array}{ccc}
2 & 5 & 1 \\
2 & 5 & 1 \\
2 & 1 & 5 \\
\end{array}

h_1(x) = x \mod 3 \\
h_2(x) = (x \mod 4) \mod 3 \\
h_3(x) = (2x) \mod 3
Example of count-min

\[
\begin{array}{cccccccc}
9 & 1 & 4 & 1 & 2 & 1 & 4 & 9 \\
\end{array}
\]

Final

\[
\begin{array}{ccc}
2 & 5 & 1 \\
2 & 5 & 1 \\
2 & 1 & 5 \\
\end{array}
\]

Frequency of 9?

\[
\min(2,5,2) = 2
\]

\[
h_1(x) = x \mod 3
\]

\[
h_2(x) = (x \mod 4) \mod 3
\]

\[
h_3(x) = (2x) \mod 3
\]
Example of count-min

Final

<table>
<thead>
<tr>
<th>2</th>
<th>5</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

Frequency of 9?

\[ \text{Min}(2,5,2) = 2 \]

\[ h_1(x) = x \mod 3 \]
\[ h_2(x) = (x \mod 4) \mod 3 \]
\[ h_3(x) = (2x) \mod 3 \]
Example of count-min

Final

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Frequency of 1?

\[ \text{Min}(5,5,5) = 5 \]

It is an over-estimation!
Example of count-min

\[
\begin{array}{cccccccc}
9 & 1 & 4 & 1 & 2 & 1 & 4 & 9 \\
\end{array}
\]

Final

\[
\begin{array}{ccc}
2 & 5 & 1 \\
2 & 5 & 1 \\
2 & 1 & 5 \\
\end{array}
\]

Frequency of 4?

\[
\text{Min}(5, 2, 5) = 2
\]

\[
\begin{align*}
h_1(x) &= x \mod 3 \\
h_2(x) &= (x \mod 4) \mod 3 \\
h_3(x) &= (2 \times x) \mod 3
\end{align*}
\]
Example of count-min

| 9 | 1 | 4 | 1 | 2 | 1 | 4 | 9 |

Final

| 2 | 5 | 1 |
| 2 | 5 | 1 |
| 2 | 1 | 5 |

Frequency of 2?

\[ \text{Min}(1,1,5) = 1 \]

\[
\begin{align*}
  h_1(x) &= x \mod 3 \\
  h_2(x) &= (x \mod 4) \mod 3 \\
  h_3(x) &= (2x) \mod 3 
\end{align*}
\]
Count-Min Sketch Error Analysis

- Focusing on first row:
  - $x[i]$ is added to $CM[1, h_1(i)]$
  - but $x[j], j \neq i$ is added to $CM[1, h_1(i)]$ with prob. $1/w$
  - The expected error is $x[j]/w$
  - The total expected error is $\frac{\sum_{j \neq i} x[j]}{w} \leq \frac{||x||_1}{w}$
  - By Markov inequality, $\Pr[\text{error} > \frac{e \cdot ||x||_1}{w}] < \frac{1}{e}$
  - By taking the minimum of $d$ rows, this prob. is $\left(\frac{1}{e}\right)^d$
Count-Min Sketch Error Analysis

- Focusing on first row:
  - \( x[i] \) is added to \( CM[1, h_1(i)] \)
  - but \( x[j], j \neq i \) is added to \( CM[1, h_1(i)] \) with prob. \( 1/w \)
  - The expected error is \( x[j]/w \)
  - The total expected error is \( \frac{\sum_{j \neq i} x[j]}{w} \leq \frac{||x||_1}{w} \)

- By Markov inequality, \( \Pr[\text{error} > \frac{e \cdot ||x||_1}{w}] < \frac{1}{e} \)

- By taking the minimum of \( d \) rows, this prob. is \( \left(\frac{1}{e}\right)^d \)
**Review: Markov’s Inequality**

- [Thm] If $X \geq 0$, then
  
  $$\Pr[X \geq a] \leq \frac{E[X]}{a}.$$ 

  In other words, if $E[X] = \mu$, then
  
  $$\Pr[X \geq k\mu] \leq \frac{1}{k}.$$
Count-Min Sketch Error Analysis

• Theorem: Give an $\varepsilon \|x\|_1$ error with prob $1 - \delta$, the count-min sketch needs to have size $\frac{e}{\varepsilon} \times \ln \frac{1}{\delta}$

• Proof: By Markov inequality, $\Pr[\text{error} \leq \frac{e \cdot \|x\|_1}{w}] \geq 1 - \left(\frac{1}{e}\right)^d$

• Then $\left(\frac{1}{e}\right)^d = \delta$, then $d = \ln \frac{1}{\delta}$, $\frac{e \cdot \|x\|_1}{w} = \varepsilon \|x\|_1$, $\frac{e}{w} = \varepsilon$,

• $w = \frac{e}{\varepsilon}$. Therefore, the size is $w \times d = \frac{e}{\varepsilon} \times \ln \frac{1}{\delta}$
• Watch a video on Count-Min
  • https://www.youtube.com/watch?v=bEmBh1HtYrw&t=79s

• Bloom filters

• Probabilistic data structure
  • https://www.youtube.com/watch?v=F7EhDBfsTA8&t=1572s