Sublinear Algorithms for Big Data

Part 4: Random Topics

Qin Zhang
Topic 3: Random sampling in distributed data streams

(based on a paper with Cormode, Muthukrishnan and Yi, PODS’10, JACM’12)
Distributed streaming

- Motivated by database/networking applications
  - Adaptive filters [Olston, Jiang, Widom, SIGMOD’03]
  - A generic geometric approach [Scharfman et al. SIGMOD’06]
  - Prediction models [Cormode, Garofalakis, Muthukrishnan, Rastogi, SIGMOD’05]

- Environment monitoring
- Network monitoring
- Cloud computing
- Sensor networks
Reservoir sampling [Waterman ’??; Vitter ’85]

- Maintain a (uniform) sample (w/o replacement) of size $s$ from a stream of $n$ items
  - Every subset of size $s$ has equal probability to be the sample
Reservoir sampling [Waterman ´??; Vitter ´85]

- Maintain a (uniform) sample (w/o replacement) of size $s$ from a stream of $n$ items
  - Every subset of size $s$ has equal probability to be the sample
- When the $i$-th item arrives
  - With probability $s/i$, use it to replace an item in the current sample chosen uniformly at random
  - With probability $1 - s/i$, throw it away
When $k = 1$, reservoir sampling has cost $\Theta(s \log n)$

When $k \geq 2$, reservoir sampling has cost $O(n)$ because it’s costly to track $i$
Reservoir sampling from distributed streams

- When \( k = 1 \), reservoir sampling has cost \( \Theta(s \log n) \)
- When \( k \geq 2 \), reservoir sampling has cost \( O(n) \) because it’s costly to track \( i \)

Tracking \( i \) approximately?
Sampling won’t be uniform
When $k = 1$, reservoir sampling has cost $\Theta(s \log n)$.

When $k \geq 2$, reservoir sampling has cost $O(n)$ because it's costly to track $i$.

Key observation:
We don’t have to know the size of the population in order to sample!
Basic idea: binary Bernoulli sampling
Basic idea: binary Bernoulli sampling
Basic idea: binary Bernoulli sampling

Conditioned upon a row having $\geq s$ active items, we can draw a sample from the active items.
Basic idea: binary Bernoulli sampling

Conditioned upon a row having $\geq s$ active items, we can draw a sample from the active items

The coordinator could maintain a Bernoulli sample of size between $s$ and $O(s)$
Random sampling – Algorithm
[with Cormode, Muthu & Yi, PODS '10 JACM '11]

- Initialize $i = 0$
- In epoch $i$:
  - Sites send in every item w.pr. $2^{-i}$
Random sampling – Algorithm

[with Cormode, Muthu & Yi, PODS '10 JACM '11]

- Initialize $i = 0$
- In epoch $i$:
  - Sites send in every item w.pr. $2^{-i}$
  - Coordinator maintains a lower sample and an upper sample: each received item goes to either with equal prob.

  (Each item is included in lower sample w.pr. $2^{-(i+1)}$)
Random sampling – Algorithm
[with Cormode, Muthu & Yi, PODS ’10 JACM ’11]

- Initialize $i = 0$

- In epoch $i$:
  - Sites send in every item w.pr. $2^{-i}$
  - Coordinator maintains a lower sample and an upper sample: each received item goes to either with equal prob.
    (Each item is included in lower sample w.pr. $2^{-(i+1)}$)
  - When the lower sample reaches size $s$, the coordinator broadcasts to $k$ sites advance to epoch $i \leftarrow i + 1$

Discards the upper sample
Randomly splits the lower sample into a new lower and an upper sample
Random sampling – Algorithm
[with Cormode, Muthu & Yi, PODS ’10 JACM ’11]

- Initialize $i = 0$

- In epoch $i$:
  - Sites send in every item w.pr. $2^{-i}$
  - Coordinator maintains a lower sample and an upper sample: each received item goes to either with equal prob.
    (Each item is included in lower sample w.pr. $2^{-(i+1)}$)
  - When the lower sample reaches size $s$, the coordinator broadcasts to $k$ sites advance to epoch $i ← i + 1$
    Discards the upper sample
    Randomly splits the lower sample into a new lower and an upper sample

**Correctness:** (1): In epoch $i$, each item is maintained in $C$ w. pr. $2^{-i}$
Random sampling – Algorithm
[with Cormode, Muthu & Yi, PODS ’10 JACM ’11]

- Initialize $i = 0$
- In epoch $i$:
  - Sites send in every item w.pr. $2^{-i}$
  - Coordinator maintains a lower sample and an upper sample: each received item goes to either with equal prob.
    (Each item is included in lower sample w.pr. $2^{-(i+1)}$)
  - When the lower sample reaches size $s$, the coordinator broadcasts to $k$ sites advance to epoch $i ← i + 1$
    Discards the upper sample
    Randomly splits the lower sample into a new lower and an upper sample

Correctness:
1: In epoch $i$, each item is maintained in $C$ w. pr. $2^{-i}$
2: Always $\geq s$ items are maintained in $C$
A running example

Maintain $s = 3$ samples

Epoch 0 ($p = 1$)

sites

$S_1$  $S_2$  $S_3$  $S_4$

upper

lower

coordinator

$C$
A running example

Maintain $s = 3$ samples
Epoch 0 ($p = 1$)

sites

$S_1$ $S_2$ $S_3$ $S_4$

coordinator

$C$

upper

lower

1
A running example

Maintain $s = 3$ samples
Epoch 0 ($p = 1$)

Coordinating sites $S_1, S_2, S_3, S_4$
A running example

Maintain $s = 3$ samples

Epoch 0 ($p = 1$)
A running example

Maintain $s = 3$ samples
Epoch 0 ($p = 1$)
A running example

Maintain $s = 3$ samples

Epoch 0 ($p = 1$)
A running example

Maintain $s = 3$ samples

Epoch 0 ($p = 1$)

Coordinators

Sites

Upper

Lower

$2$  $3$

$1$  $4$
A running example

Maintain $s = 3$ samples

Epoch 0 ($p = 1$)

upper

lower

2 3

1 4 5
A running example

Maintain \( s = 3 \) samples

Epoch 0 \((p = 1)\)

Upper sites

\[ S_1 \quad 5 \]

\[ S_2 \quad 2 \]

\[ S_3 \quad 3 \]

\[ S_4 \quad 4 \]

Now \(|\text{lower sample}| = 3\)

- discard upper sample
- split lower sample
- advance to Epoch 1

Lower sites

\[ 1 \quad 4 \quad 5 \]
A running example

Maintain $s = 3$ samples
Epoch 0 ($p = 1$)

Now $|\text{lower sample}| = 3$
- discard upper sample
- split lower sample
- advance to Epoch 1
A running example (cont.)

Maintain $s = 3$ samples

Epoch 1 ($p = 1/2$)
A running example (cont.)

Maintain \( s = 3 \) samples

Epoch 1 \((p = 1/2)\)

```
upper
4
lower
1 5
```

```
5
2
3 4
6 (discard)
```
A running example (cont.)

Maintain $s = 3$ samples

Epoch 1 ($p = 1/2$)

1. **Maintain** $s = 3$ samples.

2. **Epoch 1 ($p = 1/2$)**

   - **Coordinator**: $C$
   - **Sites**: $S_1, S_2, S_3, S_4$

   - **Values**:
     - $S_1$: 5
     - $S_2$: 2
     - $S_3$: 3, 4 (discard)
     - $S_4$: 1, 7

   - **Upper**:
     - 4
     - 7

   - **Lower**:
     - 1
     - 5
A running example (cont.)

Maintain $s = 3$ samples

Epoch 1 ($p = 1/2$)

<table>
<thead>
<tr>
<th>upper</th>
<th>lower</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

sites

<table>
<thead>
<tr>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>

(coordinator)

6 (discard)
A running example (cont.)

Maintain $s = 3$ samples

Epoch 1 ($p = 1/2$)

coordinator

<table>
<thead>
<tr>
<th>upper</th>
<th>4</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>lower</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

sites

$S_1$

5

8

$S_2$

2

$S_3$

3

4

9 (discard)

$S_4$

1

7

6 (discard)
A running example (cont.)

Maintain $s = 3$ samples

Epoch 1 ($p = 1/2$)

sites

`S_1`

5
8

`S_2`

2
6 (discard)
10

`S_3`

3
4
9 (discard)

 coordinators

`C`

upper
4
7
8

lower
1
5

10
A running example (cont.)

Maintain $s = 3$ samples
Epoch 1 ($p = 1/2$)

upper
1 5 10
lower

<table>
<thead>
<tr>
<th></th>
<th>upper</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

sites

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>$S_2$</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$S_3$</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>$S_4$</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

(coordinator)

6 (discard)
9 (discard)
A running example (cont.)

Maintain $s = 3$ samples
Epoch 1 ($p = 1/2$)

Again $|\text{lower sample}| = 3$
- discard upper sample
- split lower sample
- advance to Epoch 2
A running example (cont.)

Maintain $s = 3$ samples

Epoch 1 ($p = 1/2$)

Again $|\text{lower sample}| = 3$

- discard upper sample
- split lower sample
- advance to Epoch 2
A running example (cont.)

Maintain $s = 3$ samples

Epoch 2 ($p = 1/4$)

More items will be discarded locally

```
coordinator

sites

$S_1$

5
8

$S_2$

2
6 (discard)
10

$S_3$

3
4
9 (discard)

$S_4$

1
7
```

upper

<table>
<thead>
<tr>
<th>1</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>
A running example (cont.)

Maintain $s = 3$ samples

Epoch 2 ($p = 1/4$)

More items will be discarded locally

Intuition: maintain a sample prob. at each site $p \approx s/n$ ($n$: total # items) without knowing $n$. 

<table>
<thead>
<tr>
<th>upper</th>
<th>1</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>lower</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

coordinator

sites

$s_1$

$S_1$

5

8

$S_2$

2

6 (discard)

$S_3$

3

4

9 (discard)

$S_4$

1

7
Random sampling – Analysis

- Initialize \( i = 0 \)
- In epoch \( i \):
  - Sites send in every item w.pr. \( 2^{-i} \)
  - Coordinator maintains a lower sample and an upper sample: each received item goes to either with equal prob.
    (Each item is included in lower sample w.pr. \( 2^{-(i+1)} \))
  - When the lower sample reaches size \( s \), the coordinator broadcasts to \( k \) sites advance to epoch \( i \leftarrow i + 1 \)
    Discards the upper sample
    Splits the lower sample into a new lower sample and an upper sample

Analysis: Messages sent per epoch \( O(k + s) \) \( \times \) # epochs \( O(\log n) = O((k + s) \log n) \)
Random sampling – Analysis and experiments

- Can be
  - improved to $\Theta\left( \frac{k \log k}{s} n + s \log n \right)$ and
  - extended to sliding window cases.
Random sampling – Analysis and experiments

- Can be 1. improved to $\Theta(k \log_{k/s} n + s \log n)$ and
  2. extended to sliding window cases.

- Experiments on the real data set from 1998 world cup logs.

- Basic case
  - cost VS sample size
  - $n = 7000000$, $k = 128$

- Time-based sliding
  - cost VS sample size
  - $n = 320000$, $k = 128$

- Time-based sliding
  - cost VS window size
  - $s = 128$, $k = 128$
Random sampling – Analysis and experiments

- Can be 1. improved to $\Theta(k \log_{k/s} n + s \log n)$ and
  2. extended to sliding window cases.

- Experiments on the real data set from 1998 world cup logs.

  ![Graph showing cost vs sample size](image)

  - total # items $n = 7,000,000$
  - # items sent $\approx 4,000$
  - size of sample $s = 128$
  - # sites $k = 128$
Sampling from a (time-based) sliding window

- **Expired windows**
- **Frozen window**
- **Current window**

Diagram showing a sliding window with different states over time.
Sampling from a (time-based) sliding window

Sample for sliding window =
(1) a subsample of the (unexpired) sample of frozen window +
(2) a subsample of the sample of current window by ISWoR
Sampling from a (time-based) sliding window

- Sample for sliding window =
  - (1) a subsample of the (unexpired) sample of frozen window +
  - (2) a subsample of the sample of current window

- (1), (2) may be sampled by different rates.
  But as long as both have sizes $\geq \min\{s, \# \text{ live items}\}$, fine.
Sample for sliding window =
(1) a subsample of the (unexpired) sample of frozen window +
(2) a subsample of the sample of current window

(1), (2) may be sampled by different rates.
But as long as both have sizes $\geq \min\{s, \#\ live\ items\}$, fine.

The key issue: how to guarantee “both have sizes $\geq s$”? as items in the frozen window are expiring ...
Sampling from a (time-based) sliding window

Sample for sliding window =
(1) a subsample of the (unexpired) sample of frozen window +
(2) a subsample of the sample of current window by ISWoR

(1), (2) may be sampled by different rates. But as long as both have sizes \( \geq \min\{s, \# \text{ live items}\} \), fine.

The key issue: how to guarantee “both have sizes \( \geq s \)”?
As items in the frozen window are expiring ...

Solution: In the frozen window, find a good sample rate such that the sample size \( \geq s \).
Dealing with the frozen window

Keep all the levels? Need $O(w)$ communication
Dealing with the frozen window

Expired windows

Frozen window

Current window

Keep all the levels? Need \( O(w) \) communication

Keep most recent sampled items in a level until \( s \) of them are also sampled at the next level. Total size: \( O(s \log w) \)
Dealing with the frozen window

Keep all the levels? Need $O(w)$ communication

Keep most recent sampled items in a level until $s$ of them are also sampled at the next level. Total size: $O(s \log w)$

**Guaranteed:** There is a blue window with $\geq s$ sampled items that covers the unexpired portion of the frozen window.
Dealing with the frozen window: The algorithm

Each site builds its own level-sampling structure for the current window until it freezes.

- **Needs** $O(s \log w)$ space and $O(1)$ time per item
Dealing with the frozen window: The algorithm

- Each site builds its own level-sampling structure for the current window until it freezes.
  - Needs $O(s \log w)$ space and $O(1)$ time per item.
- When the current window freezes:
  - For each level, do a $k$-way merge to build the level of the global structure at the coordinator. Total communication $O((k + s) \log w)$.