Lp Sampling from Streams

joint work with
Gábor Tardos (Alfréd Rényi Institute of Mathematics)
Hossein Jowhari (MADALGO)

July 25, 2012

Mert Sağlam
### Lp Sampling from Update Streams

- The input is an *update stream*.
- We have an $n$ dimensional vector $x$, initially zero.
- The input is updates to the coordinates of $x$.
- When the stream is exhausted, an $\varepsilon$ relative error sampler outputs a coordinate $J$ s.t.
- An *augmented sampler* also returns an $\varepsilon$ appx. to $x_J$.

**Table:**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>-2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

$(2,5) \ (6,-2) \ (5,4) \ (2,-3) \ (8,-2)$

$$
\Pr[J = i] = (1 \pm \varepsilon) \frac{|X_i|^p}{\|x\|_p^p} \pm n^{-c}
$$

Here, $\|x\|_p^p = \sum_{i=1}^n |x_i|^p$.

---

**Equation:**

$$
\|x\|_p^p = \sum_{i=1}^n |x_i|^p
$$
Lp Sampling from Update Streams

- In SODA 2010 Monemizadeh and Woodruff introduced Lp sampling.
  - They gave $\text{poly}(1/\varepsilon, \log n)$ space $\varepsilon$ error Lp samplers for $p$ in $[0,2]$.
- In FOCS 2011 Andoni, Krauthgamer and Onak improved space usage to $O(\varepsilon^{-p} \log n)$ bits for $p$ in $[1,2]$.
- We give an Lp samplers with $O(\varepsilon^{-p} \log^2 n)$ bits of space for $p$ in $[1,2]$.
  - Our sampler works for $p$ in $[0,1]$ too, taking $O(\varepsilon^{-1} \log^2 n)$ space. For $p=0$ space usage is $O(\log^2 n)$.
  - We show that any one pass Lp sampler requires $\Omega(\log^2 n)$ bits.
  - Any one pass augmented sampler requires $\Omega(\varepsilon^{-p} \log n)$ space.
Our Lp Sampler for p=1

- The bare-bones algorithm
- For $i=1,...,n$ pick $r_i$ uniformly at random from real interval $[0,1]$
- Set $z_i = x_i / r_i$. 
- Find $i$ with $|z_i|$ maximal.
- If $|z_i| > \varepsilon^{-1} \|x\|_1$, output $J=i$, otherwise output FAIL.

<table>
<thead>
<tr>
<th>x</th>
<th>0</th>
<th>2</th>
<th>0</th>
<th>0</th>
<th>4</th>
<th>-2</th>
<th>0</th>
<th>-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>0.3</td>
<td>0.2</td>
<td>0.4</td>
<td>0.9</td>
<td>0.2</td>
<td>0.4</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>z</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>-5</td>
<td>0</td>
<td>-20</td>
</tr>
</tbody>
</table>

What is the probability that we output coordinate $i$?
Our Lp Sampler for p=1

Claim 1: Pr[J = i] ≤ ε|x_i|/∥x∥_1

- We output a coordinate only if |z_i| > ε^{-1}∥x∥_1.
- This happens only when |x_i|/r_i > ε^{-1}∥x∥_1.

Claim 2: Pr[J=i] ≥ (ε-ε^2)|x_i|/∥x∥_1

- Conditioned on |z_i| > ε^{-1}∥x∥_1, probability that |z_j| > ε^{-1}∥x∥_1 is ≤ ε|x_j|/∥x∥_1 by Claim 1.
- Union bound over all j, ∃j has probability ε.
Our Lp Sampler for $p=1$

- By Claim 2, $\Pr[J=i] \geq (\epsilon - \epsilon^2) \frac{|x_i|}{\|x\|_1}$
- Summing over all $j$, we see that the procedure outputs a coordinate with probability $(\epsilon - \epsilon^2)$
- Hence if we repeat in parallel $O(\epsilon^{-1} \log(1/\delta))$ times, and return the first non failing output, we get a coordinate with $(1-\delta)$ probability.

<table>
<thead>
<tr>
<th>$x$</th>
<th>0</th>
<th>2</th>
<th>0</th>
<th>0</th>
<th>4</th>
<th>-2</th>
<th>0</th>
<th>-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r$</td>
<td>0.3</td>
<td>0.2</td>
<td>0.4</td>
<td>0.9</td>
<td>0.2</td>
<td>0.4</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>$z$</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>-5</td>
<td>0</td>
<td>-20</td>
</tr>
</tbody>
</table>

But how do we find max coordinate of $z$ in small space?

We don't..
Our Lp Sampler for p=1

- Take $O(\log n)$ random binary strings $m^1, \ldots, m^\log n$ each of length $n$
- Take $O(\log n)$ $n$ dimensional random $\pm 1$ vectors $k^1, \ldots$
- Calculate $z^* \cdot m^l \cdot k^l$ for $l=1, \ldots, \log n$. Here $\cdot$ is coordinate-wise multiplication.
- Estimate $z_i$ by the median of $z_i \cdot m^l_i \cdot k^l_i$ for all $l$.

\begin{align*}
    x &= \begin{bmatrix} 0 & 2 & 0 & 0 & 4 & -2 & 0 & -2 \end{bmatrix} \\
    \\
    r &= \begin{bmatrix} 0.3 & 0.2 & 0.4 & 0.9 & 0.2 & 0.4 & 0.2 & 0.1 \end{bmatrix} \\
    \\
    z &= \begin{bmatrix} 0 & 10 & 0 & 0 & 20 & -5 & 0 & -20 \end{bmatrix}
\end{align*}

- It is known that $z_i^* = z_i \pm 3\|z\|_2$ with all but $n^{-c}$ probability.
Our Lp Sampler for p=1

- Approximating $z_i$ by $z^*_i$ changes our analysis only if
  \[ \epsilon^{-1} \|x\|_1 - \|z\|_2 \leq |z_i| \leq \epsilon^{-1} \|x\|_1 + \|z\|_2 \]

- Conditioned on $\|z\|_2 < 10 \|x\|_1$, $z_i$ is in this interval only with probability $2\epsilon^2 |x_i|/\|x\|_1$

- Condition $\|z\|_2 < 10 \|x\|_1$ happens with good probability and can be detected if does not happen via standard norm estimation algorithms.
Finding Duplicates

- Given an array of length $n+1$ where each item is in $[1..n]$ find an item that appears at least twice.
- By pigeonhole principle a duplicate exists.
- There is a $O(1)$ words RAM algorithm due to Floyd that runs in linear time.
- In the streaming model, a folklore $p$ pass deterministic algorithm with $O(n^{1/p} \log^{1-1/p})$ space.
Finding Duplicates

- Muthukrishnan asks whether there exists a constant pass polylog space algorithm.
- In 2007, Tarui shows that any deterministic $p$ pass algorithm needs $\Omega(n^{1/(2p-1)})$ space.
- In SODA'09 Gopalan and Radhakrishnan give a one pass $O(\log^3 n)$ space randomized algorithm.
- We give a $O(\log^2 n)$ space one pass randomized algorithm.
- We show that any one pass algorithm takes $\Omega(\log^2 n)$ space.
Finding Duplicates Upper Bound

- Run the $\frac{1}{2}$ relative error sampler on a vector $x$.
- Subtract 1 from each coordinate of $x$.
- For each item $i$ increment $x_i$ by one.
- For each item $i$ that appears multiple times, $x_i > 0$.
- We have $n$ decrements and $n+1$ increments.

\[
\begin{array}{cccccccc}
A & 5 & 1 & 2 & 7 & 2 & 4 & 3 & 6 \\
\end{array}
\]

- Hence a perfect L1 sample returns a positive coordinate with more than $\frac{1}{2}$ probability.
- $\frac{1}{2}$ relative error sampler returns positive coordinate with constant probability.
- We run $O(\log(1/\delta))$ instances of the L1 sampler and return the first positive coordinate.
Lower Bounds Map

Conjecture

$\Omega\left(\log^2 n \log (1/\delta)\right)$

Universal Relation

Augmented Indexing

FCE

$\Omega\left(\log^2 n\right)$

$\Omega\left(\varepsilon^{-p} \log n\right)$

$\Omega\left(\phi^{-1} \log^2 n\right)$

$\Omega\left(s \log n\right)$

Odd Frequency Items

Lp Sampling

Heavy Hitters

Finding Duplicates
Augmented Indexing Problem

- Alice is given a length $n$ string $x$ over the alphabet $[m]$.
- Alice sends a single message to Bob.
- Bob is given $i \in [n]$ and $x_j$ for $j < i$.
- Bob's goal is to output $x_i$.

We show that in any one round protocol with $(1-\delta)$ success probability, Alice sends a message of size $\Omega(n \log m)$ whenever $(1-\delta) > 1/m^{1-\varepsilon}$.
Lower Bounds Map

Conjecture

Upper Bound

Universal Relation

Augmented Indexing

FCE

Odd Frequency Items

Lp Sampling

Heavy Hitters

Finding Duplicates

$\Omega(\log^2 n \log(1/\delta))$

$\Omega(\log^2 n)$

$\Omega(\varepsilon^{-p} \log n)$

$\Omega(\phi^{-1} \log^2 n)$

$\Omega(s \log n)$
Universal Relation

- Alice and Bob are given a binary string each.
- Call these strings $x$ and $y$.
- Players exchange messages and the last player outputs a coordinate $i$ such that $x_i \neq y_i$. 

```
1 1
0 0
0 1
0 0
1 1
0 1
1 1
1 1
```
Universal Relation

- Suppose Alice get a length $s$ string $z$ over $[2^t]$.
- Bob gets $i \in [s]$ and $z_j$ for $j < i$.
- The players construct vectors $u$ and $v$ as follows.
  - Let $e_i$ be the $2^t$ dimensional vector 0 everywhere except coordinate $i$ and is 1 in coordinate $i$.
  - For $j=1,\ldots,s$ Alice appends $2^{s-j}$ copies of $e_{z_j}$. This is $u$.
  - For $j=1,\ldots,i-1$ Bob appends $2^{s-j}$ copies of $e_{z_j}$. Bob appends zeros to reach length $|u|$. This is $v$.
  - They randomly shuffle the positions in $u$ and $v$.
  - A mismatch reveals $x_i$ with $\frac{1}{2}$ probability.
Universal Relation

- Setting $s = t = O(\log n)$ guarantees that $|u| = |v| < n$
- By the augmented indexing lower bound we have $\Omega(st) = \Omega(\log^2 n)$ lower bound.
Lower Bounds Map

Conjecture $\Omega\left(\log^2 n \log(1/\delta)\right)$

Universal Relation $\Omega(\log^2 n)$

Augmented Indexing $\Omega(\varepsilon^{-p} \log n)$ $\Omega(\phi^{-1} \log^2 n)$

FCE $\Omega(s \log n)$

Odd Frequency Items

Lp Sampling

Heavy Hitters

Finding Duplicates
Lp Sampling Lower Bound

- Alice and Bob are given binary strings $u$ and $v$.
- Suppose there is a one pass Lp sampler with $S$ space.
- We give a one round universal relation protocol that communicates $S$ bits.
- Let $x$ be the vector the sampling algorithm implicitly keeps.
- Alice generates updates so that $x = u$.
- Bob generates updates so that $x = u - v$.
- We see that $x_i$ is positive iff $u_i \neq v_i$.
- Any Lp sampler returns a positive coordinate with constant probability. Hence an $\Omega(\log^2 n)$ lower bound holds.
Thank You!
Questions?