CS 15-859: Algorithms for Big Data Fall 2019

Lecture 8 (part 1) — Oct.24.2019

Prof. David Woodruff *Scribe: Shuli Jiang*

1 Heavy Hitter in a Stream

We would like to find "large" entries in a sequence of inputs in the turnstile streaming model. Question: How do we define "large"? We introduce two guarantees: l-1 vs. l-2.

1.1 Heavy Hitter Guarantees

l-1 guarantee

Output a set containing all items j for which $|x_j| \geq \phi|x|_1$. We allow some slacks (false positives) such that $(\phi - \epsilon)|x|_1 \leq |x_j| \leq |x|_1$ in the output set. But the set should not contain any j with $|x_j| \leq (\phi - \epsilon)|x|_1$.

l-2 guarantee

Output a set containing all items j for which $|x_j|^2 \geq \phi|x|_2^2$. We allow some slacks (false positives) such that $(\phi - \epsilon)|x|_2^2 \leq |x_j|_2^2 \leq |x|_2^2$ in the output set. But the set should not contain any j with $|x|_j^2 \leq (\phi - \epsilon)|x|_2^2$

l-1 vs. l-2 guarantee

Claim: l-2 guarantee can be much stronger than the l-1 guarantee.

Suppose $x = (\sqrt{n}, 1, 1, \ldots, 1)$

Notice that item 1 is an l-2 heavy hitter for constant ϕ, ϵ, but not an l-1 heavy hitter.

By definition, if $|x_j| \geq \phi|x|_1$ since $|x|_1 \geq |x|_2$, we have $x_j^2 \geq \phi^2|x|_1^2 \geq \phi^2|x|_2^2$. This means a l-1 heavy hitter implies a l-2 heavy hitter: a ϕ heavy hitter for l-1 is also a ϕ^2 heavy hitter for l-2.

1.2 Intuition

We would like to develop some intuition for a randomized algorithm to achieve l-2 guarantee.

Consider the following example:

Suppose you are promised at the end of the stream $x_i = n$, and $x_j \in \{0, 1\}$ for $j \in \{1, 2, \ldots, n\}$ with $j \neq i$.

Question: How to find the identity i with low memory (i.e. poly(log n) memory) streaming algorithm?

For each $j \in \{1, 2, 3, \ldots, \log n\}$, we divide the inputs into two categories, A_j and B_j. Let $A_j \subset [n]$ be the set of indices with j-th bit in their binary representation equal to 0, and B_j be the set with j-th bit equal to 1.

We maintain two counters, a_j and b_j such that $a_j = \sum_{i \in A_j} x_i$ and $b_j = \sum_{i \in B_j} x_j$ for each $j \in \{1, 2, \ldots, \log n\}$. Now we know if $a_j \geq n$, then $b_j \leq n - 1$ or vise versa. So the larger sum tells us the j-th bit in identity i. Therefore, we can read off the identity of item i.

The total memory used in this algorithm is $O(\log^2 n)$, because we maintain 2 counters, each counter needs $\log n$ bits and we repeat for all $\log n$ j's.

Now consider another example:

Suppose you are promised at the end of the stream, $x_i = 100\sqrt{n \log n}$ and $x_j \in \{0, 1\}$ for $j \in \{1, 2, \ldots, n\}$ with $j \neq i$.

Question: How to find the identity i with low memory streaming algorithm?

Again, for each $j \in \{1, 2, 3, \ldots, \log n\}$ we divide the input into two categories, A_j and B_j. Let $A_j \subset [n]$ be the set of indices with j-th bit in their binary representation equal to 0, and B_j be the set with j-th bit equal to 1.

We still maintain two counters but we scale each item by an independent random sign σ this time, so that the expectation of each counter will be 0, and the variance will be $O(\sqrt{n})$. Compute $a_j = \sum_{i \in A_j} \sigma_i \cdot x_i$ and $b_j = \sum_{i \in B_j} \sigma_i \cdot x_i$ for each $j \in \{1, 2, \ldots, \log n\}$.

For $\sigma_1, \sigma_2, \ldots, \sigma_n$, applying additive Chernoff bound we have $Pr[|\sum \sigma_i| > \sqrt{n \log n}] \leq \frac{1}{n^k}$, where k is a constant. This means the magnitude of "noise" in a count is at most $\sqrt{n \log n}$ with high probability. This implies with high probability the bucket containing the heavy item will have magnitude $\geq 100\sqrt{n \log n} - \sqrt{n \log n}$, while the other bucket will have magnitude $\leq \sqrt{n \log n}$. Thus we can read off the identity i.

Now our goal is to remove the two assumptions:

1. $x_i = 100\sqrt{n \log n}$
2. $x_j \in \{0, 1\}$ for $j \in \{1, 2, \ldots, n\}$ with $j \neq i$

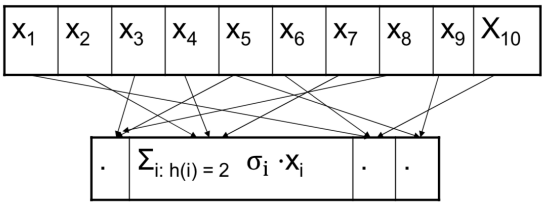
1.3 Applying CountSketch to achieve l-2 guarantee

From the previous examples, one might gain the intuition that random signs might help achieve the l-2 guarantee.

We present an algorithm to achieve l-2 guarantee using CountSketch as follow:

Assign each coordinate i a random sign $\sigma_i \in \{-1, 1\}$.

Randomly partition coordinates into B buckets, maintain $c_j = \sum_{i:h(i)=j} x_i \sigma_i$ in the j-th bucket, illustrated by the following picture.



Estimate x_i as $\sigma_i c_{h(i)}$.

Proof of correctness:

The estimate is unbiased: $E[\sigma_i c_{h(i)}] = E[\sigma_i \sum_{i':h(i)=h(i')} \sigma_{i'} x_{i'}] = x_i$ since if $i' \neq i$, $E[\sigma_i x_i] = 0$.

Suppose we independently repeat this hashing scheme O($\log n$) times. (i.e. maintain $\log n$ buckets of size B) Then output the median of the estimates across the $\log n$ repetitions.

The "Noise" in a bucket is $\sigma_i \sum_{i' \neq i, h(i')=h(i)} \sigma_{i'} x_{i'}$.

$E[\text{noise}] = 0$.

$Var[\text{noise}] = E[(\sigma_i \sum_{i' \neq i, h(i')=h(i)} \sigma_{i'} x_{i'})^2]$

$= E[\sigma_i^2 (\sum_{i' \neq i, h(i')=h(i)} \sigma_{i'} x_{i'})^2] = E[(\sum_{i' \neq i, h(i')=h(i)} \sigma_{i'} x_{i'})^2]$

$= E[\sum_{i' \neq i, i'' \neq i} \mathbf{I}(h(i') = h(i)) \mathbf{I}(h(i'') = h(i)) \sigma_{i'} x_{i'} \sigma_{i''} x_{i''}]$

$= E[\sum_{i' \neq i, i'' \neq i} \mathbf{I}(h(i') = h(i)) \mathbf{I}(h(i'') = h(i)) \sigma_{i'} \sigma_{i''}] x_{i'} x_{i''}$

where $\mathbf{I}$ is an indicator.

If $i' \neq i''$, $E = 0$.

If $i' = i''$, $E = \frac{1}{B} |x_{i'}|^2 \leq \frac{|x|_2^2}{B}$.

Therefore, $Var[\text{noise}] \leq \frac{|x|_2^2}{B}$

By Chebyshev's inequality, with constant probability, the noise in a bucket is O($\frac{|x|_2}{\sqrt{B}}$) in magnitude. This means in one repetition, with high probability, we have an estimate of $x_i \pm O(\frac{|x|_2}{\sqrt{B}})$

Now we are taking the median of $\log n$ repetitions. What is the probability that the median estimate fails to be $x_i \pm O(\frac{|x|_2}{\sqrt{B}})$?

Let $z_j = 1$ be an indicator indicating whether j-th repetition satisfies the estimate $x_i \pm O(\frac{|x|_2}{\sqrt{B}})$.

We already know that for each repetition, $E[z_j] = Pr[z_j = 1] \geq k$ where k represents a high probability (e.g. $\frac{9}{10}$). Now let $z = \sum_{j=1}^{R} z_j$, where R is the number of repetitions. $z < \frac{R}{2}$ means the median estimate fails. By Chernoff Bound, $Pr[z < \frac{R}{2}] \leq e^{-\Theta(R)} = \frac{1}{n}$ if we set $R = \Theta(\log n)$.

Therefore, since the log n repetitions are independent, this ensures that our estimate $\sigma_i c_{h(i)}$ will equal $x_i \pm$ O($\frac{|x|_2}{\sqrt{B}}$) with probability 1 - 1/poly(n). Note that poly(n) depends on the number of repetitions.

Hence, we approximate **every** x_i simultaneously up to additive error O($\frac{|x|_2}{\sqrt{B}}$).

1.4 Tail Guarantee

CountSketch approximates every x_i simultaneously up to additive error $O(\frac{|x|_2}{\sqrt{B}})$.

But what if x_1 is a super large poly(n), and $x_2 = n$ and $x_3 = \cdots = x_n = 1$? What if we want B to be small? We will get a pretty bad approximation to x_2 because the value depends on x_1.

Tail Guarantee: CountSketch approximates every x_i simultaneously up to additive error $O(\frac{|x-\frac{B}{4}|_2}{\sqrt{B}})$, where $x - \frac{B}{4}$ is x after zero-ing out its top $\frac{B}{4}$ coordinates in magnitude.r

Proof:

What is the probability that one of the top $\frac{B}{4}$ coordinates of x will go to the same bucket as x_i? By union bound, it is $\frac{1}{B} \times \frac{B}{4} = \frac{1}{4}$. This means with probability at least $\frac{3}{4}$ in each repetition the top $\frac{B}{4}$ coordinates of x in magnitude do not land in the same hash bucket as x_i.

Suppose we pick only hash functions that will not the top $\frac{B}{4}$ entries to the same bucket as x_i. The estimate: $\hat{x_i} = \sigma_i c_{h(i)} = x_i + \sum_{i' \neq i, h(i')=h(i), i' \text{not in top B/4 entries}} \sigma_i \sigma_{i'} x_{i'}$. Now $E[\hat{x_i}] = x_i$, and $Var[\hat{x_i}] \leq x_i + \frac{|x-\frac{B}{4}|_2^2}{B}$ since the top $\frac{B}{4}$ entries do not contribute to the variance. By the same argument as in the previous part, the estimate $\sigma_i c_{h(i)}$ will equal $x_i \pm O(\frac{|x-\frac{B}{4}|_2}{\sqrt{B}})$ with probability 1 - 1/poly(n).

Do we need a lot of independence? No. We only need 2-wise independence for the top $\frac{B}{4}$ entries! Because we do not care about where the top $\frac{B}{4}$ coordinates of x go to. We only care about whether they are hashed to the same bucket as x_i. Thus 2-wise independence suffices.

What happens if x is $\frac{B}{4}$ sparse? We will get 0 error and recover x. This can be another algorithm (more efficient) to recover x but it is randomized.

1.5 Deterministic l-1 Heavy Hitter

We know that for constant ϕ, l-2 guarantee implies l-1 guarantee. Why do we care about l-1 guarantee? Because l-1 guarantee can be solved deterministically!

An $s \times n$ matrix is ϵ **-Incoherent** if

1. for all columns in S_i, $|S_i|_2 = 1$.
2. for all pairs of columns S_i and S_j, $| < S_i, S_j > | \leq \epsilon$. (i.e. the columns are close to orthogonal)
3. entries can be specified with O$(\log n)$ bits of space.

By definition, we can compute $S \cdot x$ in a stream using O$(s \log n)$ bits of space.

Estimate $\hat{x_i} = S_i^T Sx$, where S_i is the i-th column in S:

$\hat{x_i} = \sum_{j=1,\ldots,n} < S_i, S_j > x_j = |S_i|_2^2 x_i \pm \max_{i,j} | < S_i, S_j > | |x|_1 = x_i \pm \epsilon |x|_1$

Therefore, we can figure out which $|x_i| \geq \phi |x|_1$ and which $|x_i| \leq (\phi - \epsilon)|x|_1$.

Now let's find an ϵ incoherent matrix.

1.6 ϵ-Incoherent Matrices

Consider a prime $q = \Theta((\log n)/\epsilon)$. Let degree $d = \epsilon \cdot q = O(\log n)$

Consider n distinct non-zero polynomials $p_1, \ldots, p_n$ each of degree less than d.

By definition, Polynomial $= \sum_{i=0}^{d-1} x^i c_i \mod q$. So there are q^d polynomials and $q^d - 1$ non-zero polynomials. We let $q^d - 1 > n$.

Associate p_i with i-th column of S. Split $s = q^2$ rows into q groups of size q. In j-th group the i-th column has a single non-zero on the $p_i(j)$-th entry. $p_i(j)$-th entry is set to be $1/q^{1/2}$.

Each column S_i has $|S_i|_2^2 = (\frac{1}{q^{1/2}})^2 \cdot q = 1$.

For two columns S_i and S_j, each has the same non-zero in the k th group iff $p_i(k) = p_j(k)$. Since all polynomials have degree $< d$, S_i and S_j can agree with $\leq d - 1$ non-zero entries. Since each non-zero entry is set to be $\frac{1}{q^{1/2}}$, $| < S_i, S_j > |_2^2 \leq \frac{d-1}{q} \leq \epsilon$ as $d = \epsilon q$.

Thus by definition, the above constructed matrix is ϵ-incoherent.