CS 15-851: Algorithms for Big Data	Spring 2024

Lecture 7 - 2/29/2024

Prof. David Woodruff — *Scribe: Siyuan Chen*

1 Recap: Turnstile Streaming Model

Given a size-n vector $\mathbf{x}$ we want to approximate the result of the following procedure using only $O(\log n)$ bits of space:

1. Initialize $\mathbf{x}$ to $\mathbf{0}_n$.
2. Sequentially update $\mathbf{x}$ using the formula $x_i \leftarrow x_i + \Delta_j$ where $\Delta_j \in \{-M, \ldots, M\}$, $M \in \text{poly}(n)$.

In the previous class, we have shown:

1. To test if $\mathbf{x} = \mathbf{0}_n$ we can use CountSketch which can be hashed using $O(\log n)$ bits.
2. Any deterministic algorithm would require $\Omega(n \log n)$ bits of space to test if $\mathbf{x} = \mathbf{0}_n$.

2 Recovering a k-Sparse Vector

In network traffic, it is common that only a few entries change across time. This leads us to the problem of k-sparse vector recovery: promised that there are k non-zero entries at the end of the stream, can you recover the k non-zero entries? It turns out there exists a solution to this problem that only requires $k\ ploy(log\ n)$ bits of memory.

Remark 1. Recovering here means there is a bijection between our representation and x. For now, we focus on the memory requirement, and leave the discussion of how this process is carried out to future classes.

Remark 2. The algorithm puts no requirement for the number of non-zero entries in the middle of the stream.

Let A be an $s \times n$ matrix such that any $2k$ columns are linearly independent, we claim that

Claim 1. From $A \cdot x$ you can recover the subset S of k non-zero entries and there values.

Proof. Proof by contradiction. Suppose there were vectors x and y with at most k non-zero entries and $A \cdot x = A \cdot y$. Now $A(x - y) = 0$. However, $x - y$ has at most $2k$ non-zero entries, and any $2k$ columns of A are linearly independent. So $x - y = 0$, i.e., x = y. ■

We can deterministically recover vector x with a fixed A. However, A has shape $s \times n$, and a naive way to store A exceeds the memory budget. The question remains to find a memory efficient way to store A.

The solution is the Vandermonde matrix. For a Vandermonde matrix $A \in \mathbb{R}_{2k \times n}$, $A_{i,j} = j^{i-1}$. We can verify that any $2k \times 2k$ sub-matrix of A are linearly independent by computing the determinant of it, which is $\Pi_j i_j \Pi_{j<j'}(i_j - i_{j'})$ for sub-matrix of columns $\{i_1, ..., i_{2k}\}$.

As the enrties of A grow exponentially with n, it may require $O(n)$ bits to store each entry of $A \cdot x$. We can improve this by storing $A \cdot x \mod p$ for large enough prime $p = poly(n)$ because the determinant remains non-zero for every sub-matrix.

Till now, we have found a deterministic approach that can solve the k-sparse vector in $kO(log(n))$ space. Given that we need at least $klog(n)$ bits to write down the the outputs [1], our solution is only one constant factor to the optimum.

3 Estimating Norms in the Streaming

In this section, we want to find z such that

$$(1-\epsilon)|\mathbf{x}|_p^p \leq z \leq (1+\epsilon)|\mathbf{x}|_p^p \quad \text{with probability} > \frac{9}{10} \tag{1}$$

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

3.1 Euclidean Norm: $p = 2$

To find z such that $(1-\epsilon)|\mathbf{x}|_2^2 \leq z \leq (1+\epsilon)|\mathbf{x}|_2^2$, we do the following:

1. Sample a CountSketch matrix S with $\frac{1}{\epsilon^2}$ rows.
2. For each update $x_i \leftarrow x_i + \Delta_j$ do $S\mathbf{x} \leftarrow S\mathbf{x} + \Delta_j S_{*i}$.
3. Output $|S\mathbf{x}|_2^2$ at the end.

Using the subspace embedding property of S, with probability $\geq \frac{9}{10}$, we have $|S\mathbf{x}|_2^2 = (1 \pm \epsilon)|\mathbf{x}|_2^2$. The space complexity of the algorithm is $\frac{1}{\epsilon^2}$ words, and each word is $O(log\ n)$ bits.

3.2 1-Norm: $p = 1$

To find z such that $(1-\epsilon)|\mathbf{x}|_1 \leq z \leq (1+\epsilon)|\mathbf{x}|_1$, we do the following:

1. Sample a Cauchy matrix S with $\frac{1}{\epsilon^2}$ rows.
2. For each update $x_i \leftarrow x_i + \Delta_j$ do $S\mathbf{x} \leftarrow S\mathbf{x} + \Delta_j S_{*i}$.

[1] Writing down the indices needs $log(\Sigma_{i=0}^k \binom{N}{i}) = kO(log\ n)$ potential choices of indices.

3. Output median of $|(S\mathbf{x})_1|, |(S\mathbf{x})_2|, \ldots, |(S\mathbf{x})_{1/\epsilon^2}|$.

Lemma 1. *Let $S \in \mathbb{R}^{r\times n}$ be a matrix of Cauchy random variables. For any $x \in \mathbb{R}^n$, constant $\delta \in (0,1)$, $\epsilon > 0$, and $r = O(\frac{1}{\epsilon^2})$, we have $z = median_{i=1,\ldots,r}|(Sx)_i| = (1 \pm \epsilon)\|x\|_1$ with probability $1-\delta$.*

Proof. By the 1-stable property of the Cauchy random variables, $|(Sx)_i| = |x|_1|C_i|$, where C_is are Cauchy random variables. So it suffies to prove $median_i|C_i| = 1 \pm \epsilon$.

The PDF function of $|C_i|$ is $f(x) = \frac{2}{\pi(1+x^2)},\ x > 0$. The CDF function of $|C_i|$ is $F(z) = \int_0^z f(x)dx = \frac{2}{\pi}arctan(z),\ z > 0$.

Let $Z_i = \mathbb{1}_{F(|C_i|)\le\frac{1}{2}-\epsilon}$ and $Z = \Sigma_i Z_i$. Then $\mathbb{E}[Z] = r(\frac{1}{2} - \epsilon_0)$. By applying Chernoff bound, we have $Pr[Z \ge \frac{r}{2}] \le Pr[|Z - \mathbb{E}[Z]| \ge \epsilon_0 r] \le e^{-\epsilon_0^2 r}$.

By setting $r = \frac{1}{\epsilon_0^2}log\frac{2}{\delta}$, $Pr[Z \le \frac{r}{2}] \ge 1 - \frac{\delta}{2}$. This means half of the $|C_i|$s is smaller than $\frac{1}{2} - \frac{1}{\epsilon_0}$.

Similarly, we can prove that half of the $|C_i|$s are larger than $\frac{1}{2} + \epsilon_0$ with probability $1 - \frac{\delta}{2}$.

Finally, by Union bound, with probability $1-\delta$, the median of the $median_i|C_i| \in (F^{-1}(\frac{1}{2} - \epsilon_0), F^{-1}(\frac{1}{2} + \epsilon_0)) = 1 \pm 4\epsilon_0$. Setting $\epsilon_0 = \frac{\epsilon}{4}$ concludes the proof. ∎

3.3 *p*-Norm estimation $0 < p < 2$

This can be achieved in $\frac{1}{\epsilon^2}$ words of space by using a *p*-stable distribution.

Definition. A distribution Π is *p*-stable if given any fixed vector $\mathbf{V} \in \mathbb{R}^n$ and independent samples $X_1, \ldots, X_n \sim \Pi$, $X_i V_i \equiv X \cdot |\mathbf{V}|_p$ where $X \sim \Pi$.

Remark 3. p-stable distribution only exists for $0 < p \le 2$. There is no close formed expression for general p-stable distribution, but they can be efficiently sampled: if $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ and $r \in [0,1]$ are uniformly random, then $\frac{sin(p\theta)}{cos^{\frac{1}{p}}\theta}(\frac{cos(\theta(1-p))}{ln\frac{1}{r}})^{\frac{1-p}{p}}$ is sampled from a p-stable distribution.

3.4 p-Norm Estimation for $p > 2$

For $p > 2$, since *p*-stable distributions do not exist, we need to consider alternative methods for estimating the *p*-norm. This estimation requires $\Omega(n^{1-2/p})$ bits of space. We approach this challenge using exponential random variables (R.V.s) and properties of the minimum of these variables.

The method involves the use of exponential random variables $E(\lambda)$ with the following properties:

- PDF: $f(x) = \lambda e^{-\lambda x}$ for $x \ge 0$, 0 otherwise.
- CDF: $F(x) = 1 - e^{-\lambda x}$ for $x \ge 0$.
- For any scalar $t \ge 0$, if $t \cdot E$ is considered, the CDF is $F(x) = 1 - e^{-\frac{\lambda}{t}x}$.

The min-stability property of exponential random variables is crucial for our estimation technique. Given independent exponential random variables $E_1, ..., E_n$ and scalars $|y_1|, ..., |y_n|$, let $q = \min(E_1/|y_1|^p, ..., E_n/|y_n|^p)$. The probability that $q > x$ is given by the product of individual probabilities, leading to $Pr[q > x] = e^{-x \sum |y_i|^p} = e^{-x|\mathbf{y}|_p^p}$, indicating that q behaves as an exponential random variable parameterized by $|\mathbf{y}|_p^p$.

To construct our estimator, we use a $P \cdot D$ sketch, where P is an $O(n^{1-2/p}) \times n$ CountSketch matrix, and D is a diagonal matrix with entries $1/E_i^{1/p}$, where E_i are standard exponential R.V.s. For any vector $\mathbf{y}$, this setup allows us to approximate $|\mathbf{y}|_p^p$ efficiently.

The estimation process is then as follows:

1. Construct P and D as described.
2. Calculate using $P \cdot D \cdot \mathbf{y}$.
3. Estimate $|\mathbf{y}|_p^p$ using the maximum of the result vector.

We first look at $|D \cdot y|_\infty^p$. Because of the min-stability property, we have $|Dy|_\infty^p = \max_i(\frac{|y_i|^p}{E_i}) = \frac{1}{\min_i \frac{E_i}{|y_i|^p}} = \frac{|y|_p^p}{E}$. Then, because $Pr[E \in (\frac{1}{10}, 10)] = (1 - e^{-\frac{1}{10}}) - (1 - e^{-1}) > \frac{4}{5}$, we know that $|Dy|_\infty \in [\frac{|y|_p}{10^{1/p}}, 10^{1/p}|y|_p]$ with probability at least $\frac{4}{5}$.

Although $|Dy|_\infty$ is a good estimation for the p-norm of (y), it takes n bits to store the result. In the later of this lecture, we will investigate how to use the count sketch matrix P to reduce the space requirement to preserve the p-Norm. The intuition is that count sketch can preserve the maximum by randomly distributing values into buckets.