CS 229r Information Theory in Computer Science April 11, 2019

Lecture 20

Instructor: Madhu Sudan *Scribe: Santhoshini Velusamy*

Topics covered

- Streaming Algorithms and their limits
  - Description of the model
  - Example algorithms
  - Lower bounds on the space complexity of streaming algorithms

Streaming setting

Model

Consider an input stream $x_1, x_2, \ldots, x_m$, where $x_i \in [n]$ for every i. The goal is to compute a function on the stream, $f(x_1 x_2, \ldots, x_m) \in \mathcal{T}$ using as less space as possible. Typically, we are interested in algorithms that require $\mathsf{poly}(\log n, \log m)$ space and there are no constraints on the runtime of the algorithm except that it must be finite.

More formally, the algorithm is a function $A : \{0,1\}^s \times [n] \to \{0,1\}^s$ and the output is a function $O : \{0,1\}^s \to \mathcal{T}$. We can treat $\sigma_0 \in \{0,1\}^s$ as a seed fed to the algorithm. Define $\sigma_i \triangleq A(\sigma_{i-1}, x_i)$ for $i \geq 1$. We want

$$\Pr_{\sigma_0}\left[O(\sigma_m) = f(x_1, x_2, \ldots, x_m)\right] \geq 1 - \epsilon.$$

In the next subsection, we describe an example problem in the streaming setting and outline some algorithms and proofs of lower bounds. Though several of the algorithms described here use randomness, they can be made deterministic with additional work.

Frequency moments

For $i \in [n]$, $f_i \triangleq f_i(x_1, \ldots, x_m) = \#\{j | x_j = i\}$. The k^{th} moment of $(f_1, f_2, \ldots, f_n)$ is defined as $F_k \triangleq \sum_{i=1}^n f_i^k$. The zeroth moment is defined as

$$F_0 = \lim_{k \to 0} F_k = \#\{i | f_i \neq 0\}.$$

Brief history

1. It is easy to see that $F_1 = m$. Every item is counted in exactly one of the $f_i's$.
2. (1985[1]) Flajolet-Martin: An algorithm to approximate F_0 with $S = \mathsf{poly}(\log(n, m))$.
3. (1998[2]) Alon-Matias-Szegedy: An algorithm to approximate F_2 in space $\mathsf{poly}(\log(n, m))$. Algorithms to approximate F_k in space $n^{1-\frac{1}{k}}$.
4. (2005[4]) [BJKS] Set-disjointness via Information Complexity
   (a) Approximating F_k requires at least space $n^{1-\frac{2}{k}}$.
   (b) Even multi-pass F_k (allowing multiple passes over the data stream) requires at least space $n^{1-\frac{3}{k}}$.

Flajolet-Martin algorithm [1]

Algorithm 1

Let $h : [n] \to [0,1]$ be a uniform hash function. Define $h_{\min}(x_1, x_2, \ldots, x_m) \triangleq \min_j \{h(x_j)\}$. The algorithm outputs $\frac{1}{h_{\min}} - 1$. The claim is $\mathbb{E}[h_{\min}] = \frac{1}{F_0+1}$. Intuitively, the elements that appear in the stream divide the space uniformly and hence, on expectation, the algorithm outputs F_0. The formal proof is given below :

Theorem 1. $\mathbb{E}_h[h_{\min}] = \frac{1}{F_0+1}$.

Proof. Fix a stream S. Let $F_0 = k$ for this stream. Without loss of generality, let us assume that $\{1, 2, \ldots, k\}$ are the elements that appear in the stream. We will prove that $\mathbb{E}_h\left[\min_{j\in[k]}\{h(j)\}\right] = \frac{1}{k+1}$. Consider the event when $\min_{j\in[k]}\{h(j)\} = r$. This happens when $h(j) = r$ for some $j \in [k]$ and $1 \geq h(i) \geq r$ for all $i \neq j, i \in [k]$. Since h is a uniformly random hash function, $\Pr\left[\min_{j\in[k]}\{h(j)\} = r\right] = k(1-r)^{k-1}$ and the expected value is given by

$$\begin{aligned}\mathbb{E}_h\left[\min_{j\in[k]}\{h(j)\}\right] &= k\int_0^1 r(1-r)^{k-1}\,dr \\ &= k\beta(2,k) \\ &= \frac{1}{k+1}.\end{aligned}$$

□

Algorithm 2

We take a hash function $h : [n] \to \mathbb{R}^{\geq 0}$ such that $\forall i,\ h(i) \sim \exp(1)$ (Exponential random variable with mean 1). Similar to Algorithm 1, we compute $h_{\min}(x_1, x_2, \ldots, x_m) = \min_j\{h(x_j)\}$. We have $\min_j\{h(x_j)\} \sim \exp\left(\frac{1}{F_0}\right)$ and the variance is $\left(\frac{1}{F_0}\right)^2$. If we repeat the algorithm $\frac{1}{\epsilon^2}$ times and take the average $\bar{h}$, the variance reduces to $\frac{\epsilon^2}{F_0^2}$. Thus, with high probability we have $\bar{h} = \frac{1}{F_0} \pm \frac{\epsilon}{F_0}$. Hence, we have a $(1 \pm \epsilon)$ approximation algorithm to compute F_0 that uses space $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)\mathsf{poly}(\log(m,n))$. It turns out that this is the best that we can do. A space complexity lower bound of $\Omega\left(\frac{1}{\epsilon^2}\right)$ can be proved for any $(1 \pm \epsilon)$ approximation algorithm to compute F_0, using the communication complexity of Gap Hamming distance [5].

Alon-Matias-Szegedy algorithm [2]

We will first describe their algorithm to compute F_2. Consider a uniformly random hash function $h : [n] \to \{+1, -1\}$ (Note: It takes n bits to remember this hash function. We can reduce the space to $\mathsf{poly}(\log n)$ by using pseudorandom hash functions). Compute $v = \sum_{j=1}^{m} h(x_j)$. Output v^2.

Claim. $\mathbb{E}_h\left[v^2\right] = F_2$

Proof. The expectation of v^2 is given by

$$\begin{aligned}
\mathbb{E}_h\left[v^2\right] &= \mathbb{E}_h\left[\sum_{j=1}^{m} h\left(x_j\right)\sum_{l=1}^{m} h\left(x_l\right)\right] \\
&= \sum_{j,l} \mathbb{E}\left[h(x_j)h\left(x_l\right)\right] \\
&= \sum_{i}\sum_{j,l} \mathbb{E}\left[1_{x_j=x_l=i}\right] \\
&= \sum_{i} f_i^2 = F_2,
\end{aligned}$$

where $1_{x_j=x_l=i}$ is the indicator random variable for the event that $x_j = x_l = i$ and last equality follows from the fact that for $x_j = x_l$, $\mathbb{E}\left[h(x_j)h\left(x_l\right)\right] = 1$ and for $x_j \neq x_l$, $\mathbb{E}\left[h\left(x_j\right)h\left(x_l\right)\right] = 0$. □

We will now describe their algorithm to compute F_k for $k \geq 3$ that uses space $\mathcal{O}\left(kn^{1-\frac{1}{k}}\right)$. Pick $j \in \{1, 2, \ldots, m\}$ uniformly at random. Let $x_j = a$ and a appears $r_j - 1$ times later, i.e., $|\{j' \geq j : x_{j'} = a\}| = r_j$. Output $m\left(r_j^k - (r_j-1)^k\right)$. The formal proof of correctness is given in Theorem 2 below.

Fact 1. *For every n positive reals $m_1, m_2, \ldots m_n$*

$$\left(\sum_i m_i\right)\cdot\left(\sum_i m_i^{2k-1}\right) \leq n^{1-1/k}\left(\sum_i m_i^k\right)^2.$$

Fact 2. *For any numbers $a > b > 0$*

$$a^k - b^k \leq (a-b)\,ka^{k-1}.$$

Theorem 2. *Let $Y_j = m\left(r_j^k - (r_j-1)^k\right)$. The expected value of Y_j is $\mathbb{E}_j\left[Y_j\right] = F_k$ and the variance $\mathsf{var}\left(Y_j\right) \leq kn^{1-1/k}F_k^2$.*

Proof. Since $j \sim [m]$ uniformly, computing the expected value of Y_j is the same as computing the sum $\sum_{j\in[m]} r_j^k - (r_j-1)^k$. Fix any particular $i \in [n]$. Let $i_1, i_2, \ldots i_{f_i}$ denote the indices where i appears in the stream (in order). By definition, we have $r_{i_1} = f_i$, $r_{i_2} = f_i - 1$ and so on until $r_{f_i} = 1$. Thus, we can write the expected value of Y_j as

$$\begin{aligned}
\mathbb{E}_j\left[Y_j\right] &= \sum_{i\in[n]} 1^k + \left(2^k - 1^k\right) + \cdots + \left(f_i^k - (f_i-1)^k\right) \\
&= \sum_{i\in[n]} f_i^k = F_k.
\end{aligned}$$

The variance of Y_j is $\mathsf{var}\left(Y_j\right) = \mathbb{E}\left[Y_j^2\right] - \mathbb{E}\left[Y_j\right]^2$. To bound the variance, we will give an upper bound on

$\mathbb{E}\left[Y_j^2\right]$. We have

$$\begin{aligned}
\mathbb{E}\left[Y_j^2\right] &= m\left[\sum_{i\in[n]} 1^{2k} + \left(2^k - 1^k\right)^2 + \cdots + \left(f_i^k - (f_i-1)^k\right)^2\right] \\
&\le m\left[\sum_{i\in[n]} k1^{2k-1} + k2^{k-1}\left(2^k - 1^k\right) + \cdots + kf_i^{k-1}\left(f_i^k - (f_i-1)^k\right)\right] \\
&\le km \sum_{i\in[n]} f_i^{2k-1} \\
&= kF_1F_{2k-1} \\
&\le kn^{1-1/k}F_k^2,
\end{aligned}$$

where the first inequality follows from Fact 2 and the last inequality follows from Fact 1. □

The variance can be further reduced to cF_k^2 by repeating the algorithm $N = \mathcal{O}\left(\lceil kn^{1-\frac{1}{k}}\rceil\right)$ times and taking the mean. The mean doesn't change but the variance decreases by a factor of N. Thus, we get an algorithm that uses $\mathcal{O}\left(kn^{1-\frac{1}{k}}\right)$ space and succeeds with a constant probability (follows from Chebyshev's inequality). Note: If m is not known beforehand, we can slightly change the algorithm and run it in parallel for different values of m or potentially use "reservoir sampling" to sample a uniform j.

In the following subsection, using the known communication complexity of a variant of the set disjointness problem, we prove worst case lower bounds on the space complexity of any streaming algorithm that allows a single pass or multiple passes over the data and computes F_k.

t-party communication complexity of t-set disjointness

t-set disjointness

- Inputs: YES and NO instances. The task is to distinguish between the two instances. $S_i \subseteq [n]$ for all i.
  - YES: $(S_1, S_2, \ldots, S_t)$ such that
    - $\exists i \in S_1 \cap S_2 \cdots \cap S_t$,
    - $\forall i' \neq i$, $\#\{j | S_i \ni i'\} \le 1$.
  - NO: $(S_1, S_2, \ldots, S_t)$ such that $\forall i, \#\{j | S_j \ni i\} \le 1$.

t-party communication model

There are t players $p_1, p_2, \ldots, p_t$ with inputs $x_1, x_2, \ldots x_t$ respectively and the goal is to compute a function $f(x_1, x_2, \ldots, x_t)$. Communication is allowed only from the i^{th} player to $(i+1)^{\text{st}}$ player. The last player p_t outputs the function value.

In the model with r rounds, player p_t is allowed to send a message to player p_1 and at most r rounds of communication is allowed before p_t outputs the function value.

Let us consider the t-set disjointness problem in the t-party communication model. The input given to the i^{th} player is $x_i = S_i$.

Fact 3. *[3]One-way communication complexity of t-set disjointness is at least $\Omega\left(\frac{n}{t}\right)$.*

Fact 4. *[3]Communication complexity of t-set disjointness is at least $\Omega\left(\frac{n}{t\log t}\right)$.*

We will now demonstrate how to use the above communication complexity lower bounds to prove worst case lower bounds on the space complexity of any streaming algorithm that computes F_k for $k \geq 2$. Specifically, we will show how a streaming algorithm that computes F_k can be used to solve the t-party t-set disjointness problem.

Let there be a one-pass streaming algorithm that computes F_k using space c. Starting from player p_1, we can assume that each player p_{i+1} can receive the previous state of the streaming algorithm from player p_i and run the algorithm starting from this state on the set S_{i+1} (assume that the elements in the set are streamed one by one) and sends the final state to player p_{i+2}. The overall communication is upper bounded by ct. In the NO instance, we have $F_k \leq n$. This is because every element in $[n]$ occurs at most once in the stream. In the YES instance, we have $F_k \geq t^k$ as there exists an element $i \in [n]$ that occurs in every set S_j and hence appears t times in the stream. By choosing $t = (2n)^{\frac{1}{k}}$, we can distinguish the YES and the NO instances. Using Fact 3, we know that $ct \geq \Omega\left(\frac{n}{t}\right)$ and hence $c \geq \Omega\left(\frac{n}{t^2}\right)$. Substituting $t = (2n)^{\frac{1}{k}}$, we get $c \geq \Omega\left(n^{1-\frac{2}{k}}\right)$.

Similarly, by using Fact 4, we can prove that any multi-pass streaming algorithm that computes F_k requires space at least $\Omega\left(n^{1-\frac{3}{k}}\right)$.

References

[1] Philippe Flajolet and G. Nigel Martin. 1985. Probabilistic counting algorithms for data base applications. J. Comput. Syst. Sci. 31, 2 (September 1985), 182-209. DOI=http://dx.doi.org/10.1016/0022-0000(85)90041-8

[2] Noga Alon, Yossi Matias, and Mario Szegedy. 1996. The space complexity of approximating the frequency moments. In Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing (STOC '96). ACM, New York, NY, USA, 20-29. DOI: https://doi.org/10.1145/237814.237823

[3] Chakrabarti, A & Khot, S & Sun, Xiaodong. (2003). Near-optimal lower bounds on the multi-party communication complexity of set disjointness. 107- 117. 10.1109/CCC.2003.1214414.

[4] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. 2002. An Information Statistics Approach to Data Stream and Communication Complexity. In Proceedings of the 43rd Symposium on Foundations of Computer Science (FOCS '02). IEEE Computer Society, Washington, DC, USA, 209-218.

[5] A. Sherstov, Alexander. (2011). The Communication Complexity of Gap Hamming Distance. Theory of Computing. 8(1):197-208, 2012.