Sketching Algorithms for Big Data Fall 2017

Lecture 20 – November 9, 2017

Prof. Piotr Indyk *Scribe: Kavya Ravichandran*

1 Overview

Today in lecture, we considered lower bounds for streaming. So far, we have seen streaming algorithms for norm estimation, heavy hitters, and sparse approximation. A natural question that arises is whether these algorithms are optimal in terms of space and time complexities.

Typical lower bound considerations proceed via one of two methods: either we use the **Pigeonhole Principle** to bound the amount of space required to distinguish different inputs, or we use the formality of **Communication Complexity** to bound the same. (Note, Pigeonhole Principle can be considered to be a special case of a communication complexity argument.)

Today, we show that randomness **and** approximation are necessary to estimate $||x||_0$ in space sub-linear in the dimension m and that we need $\Omega(\frac{1}{\varepsilon^2})$ bits to $(1+\varepsilon)-$approximate $||x||_2$.

2 Estimating $||x||_0$

2.1 Warmup Theorem

Theorem Any *deterministic exact* algorithm for computing $||x||_0$ needs $\Omega(m)$ bits of space.

Proof Assume there is an algorithm A using M bits of space. Now take any vector $y \in \{0,1\}^m, ||y||_0 = \frac{m}{2}$. Feed the coordinates of y to A and let $A[y]$ be the state of A at the end of this process, and E be its estimation of $||y||_0$.

We can decode y from $A[y]$ using the following procedure. For any $z \in \{0,1\}^m$, $||z||_0 = \frac{m}{2}$, feed z to A in state $A[y]$, obtaining $A[y+z]$. The algorithm computes an estimation E' of $||y+z||_0$. We have two cases now:

- If $y = z$, then $||y+z||_0 = \frac{m}{2}$.
- Otherwise, $y \neq z$ and $||y+z||_0 > \frac{m}{2}$.

We have $y = z$ if and only if $E = E'$. The number of distinct states $A[y]$ is lower bounded by $\binom{m}{m/2}$ $= \exp(\Omega(m))$, so M is $\Omega(m)$.

2.2 Upgraded Theorem

Theorem Any *deterministic c-approximate* algorithm for computing $||x||_0$ needs $\Omega(m)$ bits of space for $c = 1 + \varepsilon < 2$. Our algorithm, then, will produce an output estimate E such that $||x||_0 \leq E \leq c||x||_0$.

Proof For the proof here, we utilize error-correcting codes. For any $y \in \{0,1\}^m$, let $\text{ECC}(y) \in 0, 1^{m'}$, $m' = O(m)$ be such that $||\text{ECC}(y)||_0 = \frac{m'}{a}, a = \Theta(1)$ and for any $y \neq z$, the distance $||\text{ECC}(y) - \text{ECC}(z)||_0 \geq 2m'\frac{\varepsilon}{a}$. (which implies that $||\text{ECC}(y) + \text{ECC}(z)||_0 \geq \frac{m'}{a} + m'\frac{\varepsilon}{a} = m'\frac{c}{a}$). Now, take any $y \in \{0,1\}^m$ and feed the coordinates of $\text{ECC}(y)$ to A. The remainder of the argument essentially as before (except that $y = z$ if and only if $E' < m'\frac{c}{a}$).

2.3 Upgraded Theorem 2

Theorem Any *randomized exact* algorithm for computing $||x||_0$ needs $\Omega(m)$ bits of space.

Proof Assume the probability of error is less than 1/16. Take any ECC with minimum distance greater than $\frac{m'}{4}$, which would mean that we can recover from errors of $\frac{m'}{8}$ errors. (Consider a ball of radius $\frac{m'}{8}$ around the code word. Since this is empty, we simply need to approximate to the closest code word.)Then, take any y and feed the coordinates of $\text{ECC}(y)$ to A. With probability $\frac{1}{2}$ we can recover z such that $||z - \text{ECC}(y)||_0 < \frac{m'}{8}$, which means that we can recover y. In parallel, for any $i = 1...m'$, feed e_i to A with state $A[\text{ECC}(y)]$, obtaining estimate E_i and set $z_i = 0$ if and only if $E_i > ||\text{ECC}(y)||_0$ (which fails with probability 1/16). Markov's inequality implies that the fraction of errors is less than $\frac{1}{8}$ with probability greater than $\frac{1}{2}$.

2.4 Formal Recap

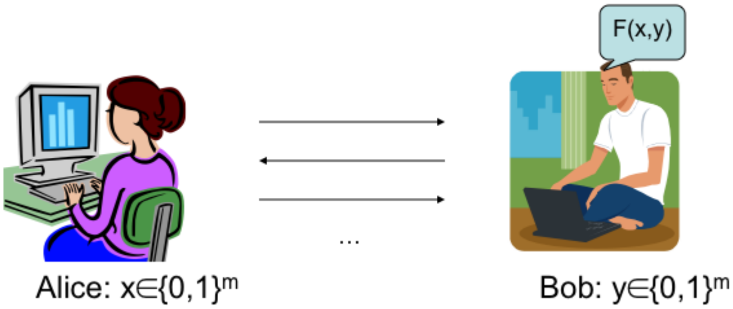
For any $y \in \{0,1\}^m$, by feeding $\text{ECC}(y)$ to A and then recovering a vector in $\{0,1\}^m$, we correctly recover y with probability $\frac{1}{2}$. Formally, we have two mappings F, G that satisfying the following property: for each $y \in \{0,1\}^m$ the probability $\mathbb{P}_r[G(F_r(y)) = y] \geq \frac{1}{2}$:

- The mapping F_r is such that given $y \in \{0,1\}^m$ and a sequence r of random bits used by the algorithm, it returns a state of the algorithm (obtained by feeding $\text{ECC}(y)$ to A).
- The mapping $G(S)$ maps a state S of the algorithm to a vector in $\{0,1\}^m$ (the mapping is defined by the recovery process).

This implies that there exists r such that $G(F_r(y)) = y$ holds for at least $\frac{1}{2}$ of $y \in \{0,1\}^m$, which then implies that the number of the states of the algorithms is at least $\frac{2^m}{2}$.

"That was basic principles. Now let's see how professionals prove lower bounds."

3 Communication Complexity



3.1 Overview

Alice and Bob are communicating in some number of rounds. We want to know what is the minimum amount of information Alice needs to transmit to Bob and vice versa in order to solve the problem. More formally, our resources are the number of bits and the number of rounds. (Today, we will only consider one-round protocols.) We have a constant $\delta > 0$ probability of error. [See Kushilevitz and Nisan [1] for more about the communication complexity.]

For a streaming algorithm, we note that if Alice has to transmit M bits to Bob, this is exactly the amount of space required to store information.

3.2 Balanced Indexing Problem

Here, Alice has a vector $x \in \{0,1\}^m$ such that $||x||_0 = \frac{m}{2}$. Bob has an index $i = 1...m$. Our goal is to compute $f(x,i) = x_i$.

Theorem Any randomized one-round protocol for indexing has $\Omega(m)$ bit complexity.

Proof We proceed using the Pigeonhole Principle as earlier. Bob cannot know the value of the bit he wants to know without knowing at least $\Omega(m)$ bits of the vector.

3.3 Gap Dot Product

Here, we are given a gap parameter Δ. Alice has a vector $u \in \mathbb{R}^m, ||u||_2 = 1$, and Bob has a vector $v \in \mathbb{R}^m, ||v||_2 = 1$. We output 0 if $u \cdot v = 0$ and 1 if $u \cdot v \geq \Delta$. Note that the difficulty of this problem is dependent upon the size of Δ. Namely, if $\Delta > 1$, it is trivial.

Theorem The randomized one-round communication complexity of GDP with gap $\Delta = \frac{1}{(m/2)^{\frac{1}{2}}}$ is $\Omega(m)$.

Proof We prove this via reduction to the balanced indexing problem. In particular, $u = \Delta x$ and $v = e_i$, where e_i is the i^{th} basis vector in the simplex. Then, we know that $u \cdot v = \Delta x_i$. Thus, the space bounds are the same.

3.4 Space Complexity of L_2 Norm Estimation

Theorem Any streaming algorithm for estimating the L_2 norm of an $m-$ dimensional vector x up to a factor of $1 \pm \Delta$, $\Delta = \frac{c}{m^{\frac{1}{2}}}$, requires $\Omega(m)$ bits for some constant $c > 0$ (even if coordinates of x have $O(\log m)$ bits).

Proof Assume we have an M-space streaming algorithm that computes $(1 \pm \Delta)||x||_2$. Then we have an $M-$space streaming algorithm that, given a stream $u \cdot v, ||u||_2 = ||v||_2 = 1$, computes $u \cdot v \pm O(\Delta)$, using the equality

$$||u - v||_2^2 = ||u||_2^2 + ||v||_2^2 - 2u \cdot v.$$

Then we have an $M-$bit one-round protocol that solves GDP with gap $\frac{1}{(m/2)^{\frac{1}{2}}}$ (assuming c small enough). Therefore, $M = \Omega(m)$.

References

[1] E. Kushilevitz and N. Nisan. *Communication Complexity.* New York, NY, USA: Cambridge University Press, 1997.