**CS 229r: Algorithms for Big Data** Fall 2015

Lecture 1 — September 3, 2015

*Prof. Jelani Nelson* *Scribes: Zhengyu Wang*

## 1 Course Information

- Professor: Jelani Nelson
- TF: Jarosław Błasiok

## 2 Topic Overview

1. Sketching/Streaming
   - "*Sketch*" $C(X)$ with respect to some function $f$ is a *compression* of data $X$. It allows us computing $f(X)$ (with approximation) given access only to $C(X)$.
   - Sometimes $f$ has 2 arguments. For data $X$ and $Y$, we want to compute $f(X, Y)$ given $C(X), C(Y)$.
   - Motivation: maybe you have some input data and I have some input data, and we want to compute some similarity measure of these two databases across the data items. One way is that I can just send you the database, and you can compute locally the similarity measure, and vise versa. But image these are really big data sets, and I don't want to send the entire data across the wire, rather what I will do is to compute the sketch of my data, and then send the sketch to you, which is something very small after compression. Now the sketch $C(X)$ is much smaller than $X$, and given the sketch you can compute the function.
   - Trivial example: image you have a batch of numbers, and I also have a batch of numbers. We want to compute their sum. The sketch I can do is just locally sum all my input data, and send you the sum.
   - *Streaming*: we want to maintain a sketch $C(X)$ on the fly as $x$ is updated. In previous example, if numbers come on the fly, I can keep a running sum, which is a streaming algorithm. The streaming setting appears in a lot of places, for example, your router can monitor online traffic. You can sketch the number of traffic to find the traffic pattern.
2. Dimensionality Reduction
   - Input data is high-dimensional. Dimensionality reduction transforms high-dimensional data into lower-dimensional version, such that for the computational problem you are considering, once you solve the problem on the lower-dimensional transformed data, you can get approximate solution on original data. Since the data is in low dimension, your algorithm can run faster.
   - Application: *speed up* clustering, nearest neighbor, etc.

3. Large-scale Machine Learning

   - For example, regression problems: we collect data points $\{(z_i, b_i) | i = 1, \ldots, n\}$ such that $b_i = f(z_i) + \text{noise}$. We want to recover $\tilde{f}$ "close" to $f$.
   - Linear regression: $f(z) = \langle x, z \rangle$, where $x$ is the parameter that we want to recover. If the noise is Gaussian, the popular (and optimal to some sense) estimator we use is Least Squares

     $$x^{LS} = \arg\min \|Zx - b\|_2^2 = (Z^T Z)^{-1} Zb, \tag{1}$$

     where $b = (b_1, \ldots, b_n)^T$ and $Z = (z_1, \ldots, z_n)^T$. If $Z$ is big, matrix multiplication can be very expensive. In this course, we will study techniques that allow us to solve least squares much faster than just computing the closed form $(Z^T Z)^{-1} Zb$.
   - Other regression problems: PCA (Principal Component Analysis), matrix completion. For example, matrix completion for Netflix problem: you are given a big product-customer matrix of customer ratings of certain products. The matrix is very sparse because not every user is going to rate everything. Based on limited information, you want to guess the rest of the matrix to do product suggestions.

4. Compressed Sensing

   - Motivation: *compress* / cheaply *acquir*e high dimensional signal (using linear measurement)
   - For example, images are very high dimensional vectors. If the dimension of an image is thousands by thousands, it means that the image has millions of pixels. If we write the image in standard basis as pixels, it is likely that the pixels are not sparse (by sparse we mean almost zero), because just image that if we take a photo in a dark room, most of the pixels have some intensity. But there are some basis called *wavelet basis*, pictures are usually very sparse on that basis. Once something is sparse, you can compress it.
   - JPEG (image compression).
   - MRI (faster acquisition of the signal means less time in machine).

5. External Memory Model

   - Motivation: measure disk I/O's instead of number of instructions (because random seeks are very expensive).
   - Model: we have infinite *disk* divided into *blocks* of size $b$ bits, and *memory* of size $M$ divided into *pages* of size $b$ bits. If the data we want to read or the location we want to write is in the memory, we can just simply do it for free; if the location we want to access is not in the memory, we cost 1 unit time to load the block from the disk into the memory, and vise versa. We want to minimize the time we go to the disk.
   - B trees are designed for this model.

6. Other Models (if time permitting)

   - For example, map reduce.

## 3 Approximate Counting Problem

In the following, we discuss the problem appearing in the first streaming paper [1].

**Problem.** There are a batch of events happen. We want to count the number of events while minimizing the *space* we use.

Note that we have a trivial solution - maintaining a counter - which takes $\log n$ bits where $n$ is the number of events. On the other hand, by Pigeonhole Principle, we cannot beat $\log n$ bits if we want to count exactly.

For *approximate* counting problem, we want to output $\tilde{n}$ such that

$$\mathbb{P}(|\tilde{n} - n| > \varepsilon n) < \delta, \tag{2}$$

where let's say $\varepsilon = 1/3$ and $\delta = 1\%$.

First of all, we can say that if we want to design a deterministic algorithm for approximate counting problem, we cannot beat against $\log \log n$ bits, because similar to previous lower bound argument, there are $\log n$ different bands (of different powers of 2), and it takes $\log \log n$ bits to distinguish them. Therefore, we maybe hope for $O(\log \log n)$ bits algorithm. Actually, the following Morris Algorithm can give us the desired bound:

1. Initialize $X \leftarrow 0$.
2. For each event, increment $X$ with probability $\frac{1}{2^X}$.
3. Output $\tilde{n} = 2^X - 1$.

Intuitively, we have $X \approx \lg n$ where $\lg x = \log_2(2+x)$. Before giving rigorous analysis (in Section 5) for the algorithm, we first give a probability review.

## 4 Probability Review

We are mainly discussing discrete random variables. Let random variable $X$ takes values in $S$. Expectation of $X$ is defined to be $\mathbb{E}\, X = \sum_{j \in S} j \cdot \mathbb{P}(X = j)$.

**Lemma 1** (Linearity of expectation)**.**

$$\mathbb{E}(X + Y) = \mathbb{E}\, X + \mathbb{E}\, Y \tag{3}$$

**Lemma 2** (Markov)**.**

$$X \textit{is a non-negative random variable} \Rightarrow \forall \lambda > 0, \mathbb{P}(X > \lambda) < \frac{\mathbb{E}\, X}{\lambda} \tag{4}$$

**Lemma 3** (Chebyshev)**.**

$$\forall \lambda > 0, \mathbb{P}(|X - \mathbb{E}\, X| > \lambda) < \frac{\mathbb{E}(X - \mathbb{E}\, X)^2}{\lambda^2} \tag{5}$$

*Proof.* $\mathbb{P}(|X - \mathbb{E}\, X| > \lambda) = \mathbb{P}((X - \mathbb{E}\, X)^2 > \lambda^2)$. It follows by Markov. □

Moreover, Chebyshev can be generalized to be:

$$\forall p > 0, \forall \lambda > 0, \mathbb{P}(|X - \mathbb{E}\, X| > \lambda) < \frac{\mathbb{E}(X - \mathbb{E}\, X)^p}{\lambda^p}. \tag{6}$$

**Lemma 4** (Chernoff)**.** *$X_1, \ldots, X_n$ are independent random variables, where $X_i \in [0, 1]$. Let $X = \sum_i X_i$, $\lambda > 0$,*

$$\mathbb{P}(|X - \mathbb{E}\, X| > \lambda \cdot \mathbb{E}\, X) \leq 2 \cdot e^{-\lambda^2 \cdot \mathbb{E}\, X/3}. \tag{7}$$

*Proof.* Since it's quite standard, and the proof detail can be found in both previous scribe[1] (Lecture 1 in Fall 2013) and wiki[2], we only include a proof sketch here. We can prove that both $\mathbb{P}(X - \mathbb{E}\, X > \lambda \cdot \mathbb{E}\, X)$ and $\mathbb{P}(X - \mathbb{E}\, X < -\lambda \cdot \mathbb{E}\, X)$ are smaller than $e^{-\lambda^2 \cdot \mathbb{E}\, X/3}$, and then apply union bound to prove the lemma.

The proof for $\mathbb{P}(X - \mathbb{E}\, X < -\lambda \cdot \mathbb{E}\, X) < e^{-\lambda^2 \cdot \mathbb{E}\, X/3}$ is symmetric to $\mathbb{P}(X - \mathbb{E}\, X > \lambda \cdot \mathbb{E}\, X) < e^{-\lambda^2 \cdot \mathbb{E}\, X/3}$. So we can focus on how to prove $\mathbb{P}(X - \mathbb{E}\, X > \lambda \cdot \mathbb{E}\, X) < e^{-\lambda^2 \cdot \mathbb{E}\, X/3}$. Since $\mathbb{P}(X - \mathbb{E}\, X > \lambda\, \mathbb{E}\, X) = \mathbb{P}(e^{t(X - \mathbb{E}\, X)} > e^{t\, \mathbb{E}\, X}) < \frac{\mathbb{E}\, e^{t(X - \mathbb{E}\, t)}}{e^{t\, \mathbb{E}\, X}}$ for any $t > 0$, we can optimize $t$ to get the desired bound. □

**Lemma 5** (Bernstein)**.** *$X_1, \ldots, X_n$ are independent random variables, where $|X_i| \leq K$. Let $X = \sum_i X_i$ and $\sigma^2 = \sum_i \mathbb{E}(X_i - \mathbb{E}\, X_i)^2$. For $\forall t > 0$,*

$$\mathbb{P}(|X - \mathbb{E}\, X| > t) \lesssim e^{-ct^2/\sigma^2} + e^{-ct/K}, \tag{8}$$

*where $\lesssim$ means $\leq$ up to a constant, and $c$ is a constant.*

*Proof.* First, we define $p$ ($p \geq 1$) norm for random variable $Z$ to be $\|Z\|_p = (\mathbb{E}\, |Z|^p)^{1/p}$. In the proof, we will also use Jensen Inequality: $f$ is convex $\Rightarrow f(\mathbb{E}\, Z) \leq \mathbb{E}\, f(Z)$.

To prove Bernstein, it's equivalent to show (equivalence left to pset)

$$\forall p \geq 1, \| \sum_i X_i - \mathbb{E} \sum_i X_i \|_p \lesssim \sqrt{p} \cdot \sigma + p \cdot K. \tag{9}$$

Let $Y_i$ be identically distributed as $X_i$, with $\{X_i | i = 1, \ldots, n\}, \{Y_i | i = 1, \ldots, n\}$ independent.

We have

[1] `http://people.seas.harvard.edu/~minilek/cs229r/fall13/lec/lec1.pdf`

[2] `https://en.wikipedia.org/wiki/Chernoff_bound`

$$\begin{aligned}
\| \sum_i X_i - \mathbb{E} \sum_i X_i \|_p &= \| \mathop{\mathbb{E}}_{Y} (\sum_i X_i - \sum_i Y_i) \|_p \qquad && \qquad (10)\\
&\leq \| \sum_i (X_i - Y_i) \|_p && \text{(Jensen Inequality)} \qquad (11)\\
&= \| \sum_i \alpha_i (X_i - Y_i) \|_p && \text{(Add uniform random signs } \alpha_i = \pm 1) \qquad (12)\\
&\leq \| \sum_i \alpha_i X_i \|_p + \| \sum_i \alpha_i Y_i \|_p && \text{(Triangle Inequality)} \qquad (13)\\
&= 2 \| \sum_i \alpha_i X_i \|_p && \qquad (14)\\
&= 2 \cdot \sqrt{\frac{\pi}{2}} \cdot \| \mathop{\mathbb{E}}_{g} \sum_i \alpha_i |g_i| X_i \|_p && \text{(Let } g \text{ be vector of iid Gaussians)} \qquad (15)\\
&\lesssim \| \sum_i \alpha_i |g_i| X_i \|_p && \text{(Jensen Inequality)} \qquad (16)\\
&= \| \sum_i g_i X_i \|_p && \qquad (17)
\end{aligned}$$

Note that $\sum_i \alpha_i |g_i| X_i$ is Gaussian with variance $\sum_i X_i^2$. The $p$th moment of Gaussian $Z \sim N(0,1)$:

$$\mathbb{E}\, Z^p = \begin{cases} 0, & p \text{ is odd.} \\ \frac{p!}{(p/2)! 2^{p/2}} \leq \sqrt{p}^p, & p \text{ is even.} \end{cases} \tag{18}$$

Therefore,

$$\begin{aligned}
\| \sum_i g_i X_i \|_p &\leq \sqrt{p} \cdot \| (\sum_i X_i^2)^{1/2} \|_p && \qquad (19)\\
&= \sqrt{p} \cdot \| \sum_i X_i^2 \|_{p/2}^{1/2} && \qquad (20)\\
&\leq \sqrt{p} \cdot \| \sum_i X_i^2 \|_p^{1/2} && (\|Z\|_p \leq \|Z\|_q \text{ for } p < q) \qquad (21)\\
&= \sqrt{p} [\| \sum_i X_i{}^2 - \mathbb{E} \sum_i X_i^2 + \mathbb{E} \sum_i X_i^2 \|_p^{\frac{1}{2}}] && \qquad (22)\\
&\leq \sqrt{p} [\| \mathbb{E} \sum_i X_i^2 \|_p^{1/2} + \| \sum_i X_i^2 - \mathbb{E} \sum_i X_i^2 \|_p^{1/2}] && \qquad (23)\\
&= \sigma \sqrt{p} + \sqrt{p} \cdot \| \sum_i X_i^2 - \mathbb{E} \sum_i X_i^2 \|_p^{1/2} && \qquad (24)\\
&\lesssim \sigma \sqrt{p} + \sqrt{p} \cdot \| \sum_i g_i X_i^2 \|_p^{1/2} && \text{(Apply the same trick (10)-(17))} \qquad (25)
\end{aligned}$$

Note that $\sum_i g_i X_i^2$ is Gaussian with variance $\sum_i X_i^4 \leq K^2 \cdot \sum X_i^2$, and $\sum_i g_i X_i$ is Gaussian with variance $\sum_i X_i^2$,

$$\|\sum_i g_i X_i^2\|_p \leq K \cdot \|\sum_i g_i X_i\|_p. \tag{26}$$

Let $Q = \|\sum_i g_i X_i\|_p^{1/2}$, we have

$$Q^2 - C\sigma\sqrt{p} - C\sqrt{p}\sqrt{K}Q \leq 0, \tag{27}$$

where $C$ is a constant.

Because it's a quadratic form, $Q$ is upper bounded by the larger root of

$$Q^2 - C\sigma\sqrt{p} - C\sqrt{p}\sqrt{K}Q = 0. \tag{28}$$

By calculation, $Q^2 \leq C\sqrt{p}\sqrt{K}Q + C\sigma\sqrt{p} \lesssim \sqrt{p}\cdot\sigma + p\cdot K$.

□

## 5 Analysis

Let $X_n$ denote $X$ after $n$ events in Morris Algorithm.

**Claim 6.**

$$\mathbb{E}\,2^{X_n} = n + 1. \tag{29}$$

*Proof.* We prove by induction on $n$.

1. Base case. It's obviously true for $n = 0$.
2. Induction step.

$$\begin{aligned}
\mathbb{E}\,2^{X_{n+1}} &= \sum_{j=0}^{\infty} \mathbb{P}(X_n = j)\cdot\mathbb{E}(2^{X_{n+1}}|X_n = j) \\
&= \sum_{j=0}^{\infty} \mathbb{P}(X_n = j)\cdot(2^j(1-\frac{1}{2^j}) + \frac{1}{2^j}\cdot 2^{j+1}) \\
&= \sum_{j=0}^{\infty} \mathbb{P}(X_n = j)2^j + \sum_j \mathbb{P}(X_n = j) \\
&= \mathbb{E}\,2^{X_n} + 1 \\
&= (n+1) + 1
\end{aligned} \tag{30}$$

□

By Chebyshev,

$$\mathbb{P}(|\tilde{n}-n|>\varepsilon n)<\frac{1}{\varepsilon^2n^2}\cdot\mathbb{E}(\tilde{n}-n)^2=\frac{1}{\varepsilon^2n^2}\,\mathbb{E}(2^X-1-n)^2. \tag{31}$$

Furthermore, we can prove the following claim by induction.

**Claim 7.**

$$\mathbb{E}(2^{2X_n})=\frac{3}{2}n^2+\frac{3}{2}n+1. \tag{32}$$

Therefore,

$$\mathbb{P}(|\tilde{n}-n|>\varepsilon n)<\frac{1}{\varepsilon^2n^2}\cdot\frac{n^2}{2}=\frac{1}{2\varepsilon^2}. \tag{33}$$

### 5.1 Morris+

We instantiate $s$ independent copies of Morris and average their outputs. Then the right hand side of (33) becomes $\frac{1}{2s\varepsilon^2}<\frac{1}{3}$ for $s>\frac{3}{2\varepsilon^2}=\Theta(\frac{1}{\varepsilon^2})$. (or $<\delta$ for $s>\frac{1}{2\varepsilon^2\delta}$)

### 5.2 Morris++

Run $t$ instantiations of Morris+ with failure probability $\frac{1}{3}$. So $s=\Theta(\frac{1}{\varepsilon^2})$. Output median estimate from the $s$ Morris+'s. It works for $t=\Theta(\lg\frac{1}{\delta})$, because if the median fails, then more than 1/2 of Morris+ fails.

Let

$$Y_i=\begin{cases}1, & \text{if } i\text{th Morris+ fails.}\\ 0, & \text{otherwise.}\end{cases} \tag{34}$$

By Chernoff bound,

$$\mathbb{P}(\sum_i Y_i>\frac{t}{2})\le\mathbb{P}(|\sum_i Y_i-\mathbb{E}\sum_i Y_i|>\frac{t}{6})\le e^{-ct}<\delta \tag{35}$$

## References

[1] Robert Morris. Counting Large Numbers of Events in Small Registers. *Commun. ACM*, 21(10): 840-842, 1978.