CS 229r: Algorithms for Big Data Fall 2013

Lecture 1 — September 3, 2013

Prof. Jelani Nelson *Scribes: Andrew Wang and Andrew Liu*

1 Course Logistics

- The problem sets can be found on the course website:
  `http://people.seas.harvard.edu/~minilek/cs229r/index.html`
- Guest lecturer on the 19th of September

1.1 Colloboration Policy

- You can work with others but please cite others when you do and write your own problem set solutions.

1.2 Prerequisites

- Algorithms (i.e. CS124)
- Discrete Math
- Discrete Probability (e.g. Linear Expectation)
- Linear Algebra

1.3 Grading

- 40% of your grade is problem sets.
- 10% is from scribing a lecture.
- 40% is from the final project paper.
- 10% is from the final project presentation.

1.4 Problem Set Information

- Problem sets are assigned every week (or week and a half).
- All problem sets must be in LaTeX and emailed.

1.5 Final Project Information

- You may work with one partner at most.
- For the final project, you should first try to make a new theoretical research contribution. The contribution does not have to involve solving an extremely hard problem, since there are so many things out there to solve. This final project can be done in pairs.
- If new research doesn't work out, you can write a survey covering many related areas in the field. Try many different things before falling back on a survey.
- Or do some work on a synthetic data set and analyze that.

1.6 Scribe Notes Information

- Scribe notes have to be emailed to Prof. Nelson the night after the lecture at 9pm.

1.7 TF Information

- There is currently no TF.
- The TF (whoever that may be) will grade psets and final projects.

2 Course Content/Topics for the semester

1. Sketching/Streaming
   - A *sketch of the data set* is a compression of the data set. Example: An algorithm can approximate document similarity between two documents by applying the similarity function to their sketches.
   - *Streaming* is creating a sketch for online data that is continually updated. Example: Consider a router with packets flying at you. You may want to keep an updated sketch.
2. Dimensionality Reduction
   - Example: You may want to run an algorithm on a data set but it scales poorly with the dimension of data, so you need to find a structure-preserving lower-dimensional representation.
3. Numerical Linear Algebra
   - Motivation: you may want to solve some linear algebra problems algorithmically.
   - Example: matrix completion for the Netflix Prize - you a have products-customer matrix of customer ratings of certain products. The matrix is sparse (i.e. mostly empty) 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, and you can do so by making assumptions on matrix structure.

4. Compressed Sensing

   - Motivation: You want high dimensional signal with structure from sparse or approximately sparse data.
   - Example: Consider images that are pixelated (m by n) and every entry has a value corresponding to pixel color. These are usually not sparse, but if you think about these images in another representation, they could be sparse. How can you acquire such signals very quickly and recover them from that compressed acquisition?

5. External Memory Model

   - Motivaton: In CS124, we measure running time by simple steps (like arithmetic operations) to predict performance of an algorithm.
   - But sometimes this isn't accurate, because accesses to memory have significantly different time costs (6 orders of magnitude).
   - We want to use a model that takes this into account.
   - The *external memory model* assumes bounded memory has size M and an infinite disk, where touching data in memory is free and data on disk costs 1.
   - The cost is primarily the seeking time, not the reading time, because surrouding blocks are easy to read but seeking is expensive.

6. Mapreduce/Hadoop

   - MapReduce and Hadoop are technologies dealing with parallel computing on massive datasets that go beyond single machine capabilities.

3 Probability Review

Let $X_1, \ldots, X_n$ be discrete r.v.s on $S \subseteq \mathbb{R}$.

Definition 1. *(Expectation).* $\mathbb{E}\, X = \sum_{j \in S} \mathbb{P}(X = j) \cdot j$.

Definition 2. *(Variance).* $Var(X) = \mathbb{E}(X - \mathbb{E}\, X)^2 = \mathbb{E}\, X^2 - (\mathbb{E}\, X)^2$.

Lemma 3. *(Markov's inequality). If X is a non-negative r.v., then $\mathbb{P}(X > \lambda) < \frac{\mathbb{E}\, X}{\lambda}$, for any $\lambda > 0$.*

Proof. If not (if $\exists$ bad λ), then $\mathbb{E}\, X$ would be too big (check this for yourself). □

Lemma 4. *(Chebyshev's inequality). If $\forall \lambda > 0, \mathbb{P}(|X - \mathbb{E}\, X| > \lambda) < \frac{Var(X)}{\lambda^2}$.*

Proof. $|X - \mathbb{E}\, X| > \lambda \Leftrightarrow (X - \mathbb{E}\, X)^2 > \lambda^2$. By Markov,

$$\mathbb{P}(|X - \mathbb{E}\, X| > \lambda) = \mathbb{P}((X - \mathbb{E}\, X)^2 > \lambda^2) < \frac{\mathbb{E}(X - \mathbb{E}\, X)^2}{\lambda^2}$$

□

Lemma 5. *(Chernoff bound). Suppose we have n independent Bernoulli r.v.s $X_1, ..., X_n$ with $X_i \sim$ Bernoulli(p_i). Also let $X := \sum_i X_i$ and $\mu = \mathbb{E}\, X$. Then for constants $k, c > 0$,*

$$\mathbb{P}(|X - \mathbb{E}\, X| > \lambda\mu) \leq ke^{-c\lambda^2\mu}$$

Proof. By the union bound,

$$\mathbb{P}(|X - \mathbb{E}\, X| > \lambda\mu) \leq \mathbb{P}(X > (\lambda + 1)\mu) + \mathbb{P}(X < (1 - \lambda)\mu)$$

Let's first bound $\mathbb{P}(X > (\lambda + 1)\mu)$, denoted by **(*)**. Because $X > (1 + \lambda)\mu \Leftrightarrow e^{tX} > e^{t(1+\lambda)\mu}$ for positive t, we apply Markov's inequality to e^{tX} (a positive r.v.) to get

$$\textbf{(*)} = \mathbb{P}(e^{tX} > e^{t(1+\lambda)\mu}) < \frac{\mathbb{E}\, e^{tX}}{e^{t(1+\lambda)\mu}} \forall t > 0$$

Bounding the numerator, we have

$$\mathbb{E}\, e^{tX} = \mathbb{E} \prod_i e^{tX_i} = \prod_i \mathbb{E}\, e^{tX_i}$$

by independence of the X_i. Because the X_i are Bernoulli, this becomes

$$\mathbb{E}\, e^{tX} = \prod_i (p_i e^t + (1 - p_i)) = \prod_i (p_i(e^t - 1) + 1) \leq \prod_i e^{p_i(e^t - 1)} = e^{(e^t - 1)\mu}$$

with the last inequality given by $1 + x \leq e^x$ by Taylor's Theorem. Setting $t = \ln(1 + \lambda)$, we get

$$\frac{\mathbb{E}\, e^{tX}}{e^{t(1+\lambda)\mu}} \leq \frac{e^{\lambda\mu}}{e^{\ln(1+\lambda)(1+\lambda)\mu}} = \left(\frac{e^{\lambda}}{e^{\ln(1+\lambda)(1+\lambda)}}\right)^{\mu} \leq \left(\frac{e^{\lambda}}{e^{(1+\lambda)(\lambda - \frac{\lambda^2}{2} + O(\lambda^3)}}\right)^{\mu} = e^{(\lambda - \lambda - \frac{\lambda^2}{2})\mu} = e^{-c\lambda^2\mu}$$

We can do a similar proof to bound the other half of the union bound, $\mathbb{P}(X < (-\lambda + 1)\mu)$. This time, we have $\mathbb{P}(X < (1 - \lambda)\mu) = \mathbb{P}(e^{-tX} > e^{-t(1-\lambda)\mu})$ for any positive t. So we can apply Markov's inequality:

$$\mathbb{P}(e^{-tX} > e^{-t(1-\lambda)\mu}) < \frac{\mathbb{E}\, e^{-tX}}{e^{-t(1-\lambda)\mu}} \leq \frac{e^{(e^{-t} - 1)\mu}}{e^{-t(1-\lambda)\mu}}$$

by very similar steps as before (we're essentially substituting $-t$ for t). Choosing $t = -\ln(1 - \lambda)$ gives

$$\mathbb{P}(X < (1 - \lambda)\mu) < \left(\frac{e^{-\lambda}}{e^{\ln(1-\lambda)(1-\lambda)}}\right)^{\mu} \leq \left(\frac{e^{-\lambda}}{e^{(1-\lambda)(-\lambda - \frac{\lambda^2}{2} - O(\lambda^3)}}\right)^{\mu} = e^{(-\lambda + \lambda - \frac{\lambda^2}{2})\mu} = e^{-c\lambda^2\mu}$$

This gives the desired bound. □

4 Algorithms for Big Data Example 1

Motivating question: How do you maintain an approximate counter for the number of elements n seen in a data stream that can be stored in fewer than $\log n$ bits? ($\log n$ bits can be done by just incrementing with every new stream object.)

4.1 Preliminary Solution: Morris Algorithm

- Maintains a counter using $\log \log n$ bits.
- Algorithm Steps: Have counter X.
  1. Initialize X to 0.
  2. If asked to increment, then do so with probability $\frac{1}{2^X}$. Else, do nothing.
  3. When done, output $2^X - 1$.

4.2 Proof/Analysis

- Compute expectation to show that Morris provides an unbiased estimate. Then check our estimator's variance.

Claim 6. $\mathbb{E}\, 2^X = n + 1$

Proof. Let counter's state after seeing n items be X_n. $\mathbb{E}\, 2^{X_n} = \sum_{j=0}^{\infty} \mathbb{P}(X_{n-1} = j)\, \mathbb{E}(2^{X_n} | X_{n-1} = j) = \sum_{j=0}^{\infty} \mathbb{P}(X_{n-1} = j)(\frac{1}{2^j} 2^{j+1} + (1 - \frac{1}{2^j})2^j) = \sum_{j=0}^{\infty} \mathbb{P}(X_{n-1} = j)(2^j + 1) = 1 + \mathbb{E}\, 2^{X_{n-1}} \rightarrow$ Then the claim is proved by induction. □

Lemma 7. $\mathbb{E}\, 2^{2X} = \frac{3}{2}n^2 + \frac{3}{2}n + 1$

Proof. The proof is by induction. For the inductive step,

$$\begin{aligned}
\mathbb{E}\, 2^{2X_n} &= \sum_{j=0}^{\infty} \mathbb{P}(2^{X_{n-1}} = j) \cdot \mathbb{E}(2^{2X_n} | 2^{X_{n-1}} = j) \\
&= \sum_{j=0}^{\infty} \mathbb{P}(2^{X_{n-1}} = j) \cdot \left[\frac{1}{j} \cdot 4j^2 + \left(1 - \frac{1}{j}\right) \cdot j^2 \right] \\
&= \sum_{j=0}^{\infty} \mathbb{P}(2^{X_{n-1}} = j) \cdot (j^2 + 3j) \\
&= \mathbb{E}\, 2^{2X_{n-1}} + 3 \cdot \mathbb{E}\, 2^{X_{n-1}} \\
&= 3(n-1)^2/2 + 3(n-1)/2 + 1 + 3n.
\end{aligned}$$

The lemma now follows by rearranging terms. □

4.3 Revised Morris's Algorithm

We've shown our estimator is unbiased and the above lemma shows that its variance is $O(n^2)$. We can lower the variance by having t counters run in parallel and averaging them.

- We have t counters, $X_1, ..., X_t$ and we will output $\frac{1}{t}(\sum_{j=1}^{t} 2^{X_j} - 1)$

- Then the new variance $Var(\frac{1}{t}(\sum_{j=1}^{t}(2^{X_{j-1}})) \leq O(\frac{n^2}{t})$ due to independence of the parallel counters. The new estimator is still unbiased.

Claim 8. *If $t \geq \frac{c}{\epsilon^2}$ then $\mathbb{P}(|\hat{n} - n| > \epsilon n) < \frac{1}{3}$ (where $\hat{n}$ is the average of the t trials).*

Proof. $\mathbb{P}(|\hat{n} - n| > \epsilon n) < O(\frac{n^2}{t})\frac{1}{\epsilon^2 n^2}$ by Chebyshev, and we can set $t = O(\frac{1}{\epsilon^2})$ for the constant in the big-Oh "big enough" to make the final expression less than or equal to $\frac{1}{3}$. □

4.4 Final Morris(ish) Algorithm

- 1. Initialize $X_1, ..., X_n$ where $t = O(\frac{1}{\epsilon^2})$ each to 0.
  2. Upon incrementing, run each step of X_j independently.
  3. Output sum $\frac{1}{t}\sum_j 2^{X_{j-1}}$.

- Do these three steps $m = O(\log \frac{1}{\delta})$ times independently in parallel and output the median result. Let this median result be n_{tw}.

4.5 Analysis

Claim 9. $\mathbb{P}(|n_{tw} - n| > \epsilon n) < \delta$

Proof. Let Y_i be an indicator r.v. for the event $|\hat{n_i} - n| \leq \epsilon n$ where $\hat{n_i}$ be the ith trial. Let $Y = \sum_i Y_i$. $\mathbb{P}(|n_{tw} - n| > \epsilon n) \leq \mathbb{P}(Y \leq \frac{m}{2}) \leq \mathbb{P}(|Y - \mathbb{E}\,Y| > 2\frac{m}{3} - \frac{m}{2}) = \mathbb{P}(|Y - \mathbb{E}\,Y| \geq \frac{m}{6}) \leq \mathbb{P}(|Y - \mathbb{E}\,Y| \geq \frac{\mu}{4}) < e^{-c(\frac{1}{4})^2\frac{2m}{3}} < e^{-c\log\frac{1}{\delta}} < \delta$, with the second-to-last inequality given by the stipulated $m = O(\log \frac{1}{\delta})$, and the last few inequalities holding up to a constant. □

References

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