CS:4980:0002 Algorithmic Excursions **Spring 2016**

Lectures 11 & 12: February 25, 2016

Lecturer: Kasturi Varadarajan *Scribe: Richard Blair*

11.1 Items from last week's lectures

11.1.1 k-means and k-median

There are algorithms that can find a $(1+\epsilon)$-approximation in $O(nd\exp(\frac{R}{d}))$ time.

Term paper topics:

- "Guess centers" problem
- *Coresets*: Approximate the input by a "coreset" and run a more expensive algorithm on the coreset.

For k-median there exists an algorithm to obtain a $(1+\epsilon)$-approximation in time $O(\text{poly}(n,k))$. It is a long-standing open problem whether such an algorithm exists for k-means.

11.2 Streaming algorithms: counting events in sublinear space

The course site of *Algorithms for Big Data* by Jelani Nelson of Harvard University is a good resource for information about many of these algorithms.

Problem: Suppose some events are happening, and we are notified of events when they happen. We want to count the number of events that have happened.

Trivial solution: Maintain a counter, and increment it for each event. The space we use to keep track of n events is $\approx \log n$. Let us see if we can do better. Given $0 < \epsilon, \delta < 1$, we would like to keep an estimate $\bar{n}$ such that

$$\Pr[|\bar{n} - n| > \epsilon n] < \delta$$

This question was first considered by Morris in a 1978 paper in *Communications of the ACM* [MO78].

Morris' algorithm:

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

Why does this work?

Let X_n denote X after n events.

Certainly

- $X_0 = 0$.
- $X_1 = 1$
- $X_2 = 1$ with probability $\frac{1}{2}$ and $X_2 = 2$ with probability $\frac{1}{2}$
- We also have $\Pr[X_3 = 1] = \frac{1}{4}$, $\Pr[X_3 = 2] = \frac{5}{8}$, $\Pr[X_3 = 3] = \frac{1}{8}$.

What is $E[2^{X_2}]$?

$E[2^{X_2}] = \frac{1}{2} \cdot 2^1 + \frac{1}{2} \cdot 2^2 = 1 + 2 = 3$. So $E[\bar{n}]$=2.

Similarly we obtain

$$\begin{aligned} E[2^{X_3}] &= \frac{1}{4} \cdot 2 + \frac{5}{8} \cdot 2^2 + \frac{1}{8} \cdot 2^3 \\ &= \frac{1}{2} + \frac{5}{2} + 1 \\ &= 4 \end{aligned}$$

Lemma 11.1 $E[2^{X_n}] = n + 1$.

Proof: The proof is by induction. The base case concerns $E[2^{X_0}]$. $E[2^{X_0}] = 2^0 \cdot \Pr[X_0 = 0] = 1 \cdot 1 = 2$, so the base case holds for $n = 0$. Let us now assume $E[2^{X_n}] = n + 1$. Now, we have

$$\begin{aligned} E[2^{X_{n+1}}] &= \sum_{j=0}^{\infty} \Pr[X_n = j] \cdot E[2^{X_{n+1}} \mid X_n = j] \\ &= \sum_{j=0}^{\infty} \Pr[X_n = j] \cdot \left(\left(1 - \frac{1}{2^j} \right) \cdot 2^j + \frac{1}{2^j} \cdot 2^{j+1} \right) \\ &= \sum_{j=0}^{\infty} \Pr[X_n = j] \cdot \left(2^j - 1 + 2 \right) \\ &= \sum_{j=0}^{\infty} \Pr[X_n = j] \cdot 2^j + \sum_{j=0}^{\infty} \Pr[X_n = j] \\ &= E[2^{X_n}] + 1 \end{aligned}$$

■

Homework problem: Show that $Var(\bar{n}) := E[(\bar{n} - n)^2]$ is at most $\frac{n^2}{2}$, by showing that $E[2^{2X_n}] = \frac{3}{2}n^2 + \frac{3}{n} + 1$.

Chebyshev's inequality yields the bound

$$Pr[|\bar{n} - n| > t] < \frac{Var(\bar{n})}{t^2} \leq \frac{n^2}{2t^2}$$

We would like to improve on this bound.

11.2.1 Morris+

Consider a new algorithm.

Morris+:

1. Instantiate s copies of Morris.
2. Let $\bar{n}_i$ be the estimate given by the i^{th} copy.
3. Return the estimate of Morris+, $\hat{n} = \frac{1}{s}\sum_{i=1}^{s} \bar{n}_i$

Note that $Var(\hat{n}) = \frac{Var(\bar{n})}{s}$ by the independence of the copies of Morris.

Chebyshev's inequality now yields

$$Pr[|\hat{n} - n| > t] < \frac{Var(\bar{n})}{t^2} \leq \frac{n^2}{2st^2}.$$

To get the bound we wanted, substitute ϵn for t. Then

$$Pr[|\hat{n} - n| > \epsilon n] \leq \frac{n^2}{2s\epsilon^2 n^2} = \frac{1}{2s\epsilon^2}.$$

Example: if we have $\delta = \frac{1}{3}$, then we need $s \geq \frac{3}{2\epsilon^2}$. If δ be chosen, then $s \geq \frac{1}{2\delta\epsilon^2}$.

It is possible to improve on this even further, with the following:

Morris++: Run t copies of Morris+ and return the median of the t estimates.

One can apply the Chernoff bound to show that the total number of copies of Morris necessary in the Morris++ algorithm to obtain the guarantee

$$\Pr[|\hat{n} - n| > \epsilon n] \leq \delta$$

is $O\left(\log\log n \cdot \frac{1}{\epsilon^2} \cdot \log\frac{1}{\delta}\right)$.

To this end, define 0-1 random variables Y_i thus:

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

Now, $E[Y] = \sum_i E[Y_i]$, by linearity of expectation.

Since we have a sum of independent random variables, we may apply a special case of the Chernoff bound for a collection of i.i.d. 0-1 random variable to obtain

$$\Pr\left[|Y - E[Y]| > \lambda E[Y]\right] \leq \frac{2}{e^{\lambda^2 E[Y]/3}}.$$

Since we are using the "median trick", and Morris++ fails only in the case that at least $\frac{t}{2}$, Morris+ copies fail, we have

$$\Pr\left[\text{Morris++ fails}\right] \leq \Pr\left[Y \geq \frac{t}{2}\right] \leq \Pr\left[|Y - E[Y]| > \frac{t}{6}\right].$$

So we plug in $\lambda E[Y] = \frac{t}{6}$, which gives $\lambda \geq \frac{1}{2}$.

We then have

$$\Pr\left[|Y - E[Y]|\right] \leq \frac{2}{e^{\frac{t}{6} \cdot \frac{1}{2} \cdot \frac{1}{3}}} = 2e^{-\frac{t}{36}}.$$

If we want this bound to be less than δ, take

$$t = c \cdot \log \frac{1}{\delta}.$$

The space needed by Morris++ is thus $O\left(\log \log n \cdot \frac{1}{\epsilon^2} \cdot \log \frac{1}{\delta}\right)$.

11.3 Streaming algorithms: Estimating the number of distinct elements in a stream

First we shall introduce some notation. We shall write [n] to denote the set $0, 1, \ldots, n-1$.

We will consider the situation in which there is a stream $\sigma = \langle a_1, a_2, \ldots, a_m \rangle$ with each $a_i \in [n]$. Now n is known in advance, but σ is not.

Example: $\sigma = \langle 23, 12, 7, 23, 9, 7, 16, 7 \rangle$. There are here 5 distinct elements in 8 elements.

Now, define $f_i := |\{j \mid a_j = i\}|$, i.e. the frequency of i.

Let $f := (f_0, f_1, \ldots, f_{n-1})$, i.e., a vector of all frequencies of elements of $[n]$.

Let $d := |\{j \mid f_j > 0\}|$

Our goal now is to output an (ϵ, δ)-approximation $\bar{d}$, that is, output a $\bar{d}$ such that

$$\Pr\left[\left|d - \bar{d}\right| > \epsilon d\right] \leq \delta.$$

Another good resource for this topic is the set of lecture notes *Data Stream Algorithms* by Amit Chakrabarti of Dartmouth College [AC15].

Sublinear space for this estimator is not attainable if $\epsilon = 0$ or $\delta = 0$.

Homework: Read the literature and explore why this is so.

We shall now pick a hash function $h : [n] \to [0, 1]$. Imagine it is uniformly random. Intuitively, the more distinct elements are in our stream, the smaller the minimum value of our hash function on the stream should be. If the minimum $= \alpha$, the estimate is $\frac{1}{\alpha}$.

For an integer $p \geq 0$, let zeroes(p) be the maximum element in the set $\left\{i \in \mathbb{N} \mid 2^i \text{ divides } p.\right\}$.

$$\begin{aligned} \text{zeroes}(2) &= 1 \\ \text{zeroes}(3) &= 0 \\ \text{zeroes}(12) &= 2 \end{aligned}$$

Now assume n is a power of 2.

The following is our algorithm.

Data: A stream $\sigma = \langle a_1, a_2, \ldots, a_m \rangle$
Result: An estimate $\bar{d}$ of the number of distinct elements in the stream
Choose a random hash function $h : [n] \to [n]$ from a 2-universal family of hash functions.
$z \leftarrow 0$
for *each element* $\mathrm{a_i}$ *of the stream* **do**
  if $\mathrm{zeroes(h}(a_i)) > z$ **then**
    $z \leftarrow \mathrm{zeroes(h}(a_i))$
  end
end
return $2^{z+\frac{1}{2}}$

Algorithm 1: Counting distinct elements in sublinear space

Now, fix $j \in [n]$, and fix an integer k.

What is $\Pr[\mathrm{zeroes(h}(j)) > k]$?

This is equal to $\Pr\left[2^k \text{ divides } \mathrm{h}(j)\right]$, by definition, and

$$\Pr\left[2^k \text{ divides } \mathrm{h}(j)\right] = \frac{1}{2^k}.$$

Set $k = \log d$. Then

$$\Pr\left[2^{\log d} \text{ divides } \mathrm{h}(j)\right] = \log \frac{1}{d}.$$

There is a fair chance that $\mathrm{zeroes(h}(j)) \geq \log d$ for at least one j. Also, the chance that $\mathrm{zeroes(h}(j)) \gg \log d$ for at least one j is quite small. Next time shall see if we can bound the probability that $\bar{d}$ is very far from d.

References

[AC15] Amit Chakrabarti, 2015. "Data Stream Algorithms." http://www.cs.dartmouth.edu/ ac/Teach/data-streams-lecnotes.pdf.

[RM78] Robert Morris, 1978. "Counting large numbers of events in small registers." *Commun. ACM* 21, 10 (October 1978), 840-842. DOI=http://dx.doi.org.proxy.lib.uiowa.edu/10.1145/359619.359627

[JN15] Jelani Nelson, 2015. "Algorithms for Big Data (CS:229r)." http://people.seas.harvard.edu/ minilek/cs229r/fall15/lec.html.