CSE 522: Sublinear (and Streaming) Algorithms **Spring 2014**

Lecture 3: Estimating the Number of Distinct Elements

April 7, 2014

Lecturer: Paul Beame *Scribe: Paul Beame*

1 A constant factor approximation algorithm for F_0

We begin by analyzing the basic estimation algorithm we saw last time.

Basic F_0 Estimation Algorithm

Initialize: Choose $h : [M] \to [M]$ from H a pairwise independent family of hash functions
$z \leftarrow 0$
Process:
for each i **do**
  $z \leftarrow \max(\text{zeroes}(h(x_i), z)$
end for
Output:
$2^{z+1/2}$ (geometric mean of 2^z and 2^{z+1})

It takes $2\log_2 M$ bits to represent h and $\log\log n$ bits for z so the total space is $O(\log M + \log\log n)$.

We will see that with constant probability, the output of this algorithm is between $F_0/3$ and $3F_0$ and then see how to improve the success probability to $1-\delta$ for any $\delta > 0$.

Analysis Let $z^* =$ the final value of z.

For each $j \in [M]$ and integer $r \geq 0$, let

$$X_{r,j} = \begin{cases} 1 & \text{if } \text{zeroes}(h(j)) \geq r \\ 0 & \text{otherwise} \end{cases}$$

and let

$$Y_r = \sum_{j:\ f_j > 0} X_{r,j}$$

which represents the total number of distinct input values whose hashes have at least r trailing zeroes. Observe that $Y_r = 0 \Leftrightarrow z^* \leq r-1$. Then

$$\mathbb{E}(X_{r,j}) = \mathbb{P}[\text{zeroes}(h(j)) \geq r] = \mathbb{P}[2^r \text{ divides } h(j)] = 1/2^r$$

since $h(j)$ is uniformly random. Therefore

$$\mathbb{E}(Y_r) = \sum_{j:\ f_j>0} \mathbb{E}(X_{r,j}) = \sum_{j:f_j>0} 1/2^r = F_0/2^r.$$

Furthermore, Y_r is the sum of $X_{r,j}$ which are pairwise independent because of the pairwise independence of h, hence $Var(Y_r) \leq \mathbb{E}(Y_r) = F_0/2^r$. Let $\tilde{F}_0 = 2^{z^*+1/2}$ be the estimate produced by the algorithm. We will argue that $F_0/3 < \tilde{F}_0 < 3F_0$ with constant probability.

$\tilde{F}_0 < 3F_0$**:** Let a be the smallest integer such that $2^{a+1/2} \geq 3F_0$. Then

$$\begin{aligned}
\mathbb{P}[\tilde{F}_0 \geq 3F_0] &= \mathbb{P}[z^* \geq a] \\
&= \mathbb{P}[Y_a > 0] \\
&= \mathbb{P}[Y_a \geq 1] \\
&\leq \mathbb{E}[Y_a] \qquad \text{by Markov's inequality} \\
&= F_0/2^a \\
&\leq \sqrt{2}/3 \approx .471... \qquad \text{by definition of } a
\end{aligned}$$

$\tilde{F}_0 > F_0/3$**:** Let b be the largest integer such that $2^{b+1/2} \leq F_0/3$. Then

$$\begin{aligned}
\mathbb{P}[\tilde{F}_0 \leq F_0/3] &= \mathbb{P}[z^* \leq b] \\
&= \mathbb{P}[Y_{b+1} = 0] \\
&\leq \mathbb{P}[|Y_{b+1} - \mathbb{E}(Y_{b+1})| \geq \mathbb{E}(Y_{b+1})] \\
&\leq \frac{Var(Y_{b+1})}{\mathbb{E}(Y_{b+1})^2} \qquad \text{by Chebyshev's Inequality} \\
&\leq 1/\mathbb{E}(Y_{b+1}) \\
&= 2^{b+1}/F_0 \\
&\leq \sqrt{2}/3 \approx .471... \qquad \text{by definition of } b
\end{aligned}$$

The guarantee for success is very weak, little more than 5% of the time. However we can easily modify the above algorithm to obtain success with probability at least $1 - \delta$ for any $\delta > 0$.

Median Trick We use the fact that each of the two failure conditions happens with probability bounded strictly below 1/2 to obtain our desired algorithm:

Run $k = \Theta(\log(1/\delta))$ independent copies of the basic F_0 estimation algorithm to yield estimates $\tilde{F}_{0,1}, \ldots, \tilde{F}_{0,k}$ and then output $\tilde{F}_0$ as the median of these answers.

$\mathbb{P}[\tilde{F}_0 \geq 3F_0]$ is the probability that at least $k/2$ of the estimates $F_{0,i}$ is at least $3F_0$. Since these are independent trials, $\mathbb{P}[\tilde{F}_0 \geq 3F_0]$ is at most the probability that the binomial distribution $B(k, \sqrt{2}/3)$ has value at least $k/2$, a constant factor larger than its expectation. By the Chernoff Bound, this is at most e^{-ck} for some constant $c > 0$. For $k = \frac{1}{c}\ln(2/\delta)$ this is at most $\delta/2$.

The same analysis applies to show $\mathbb{P}[\tilde{F}_0 \leq F_0/3] \leq \delta$, yielding a total failure probability at most δ.

The total space used by the algorithm is now $O(\log(1/\delta)(\log M + \log\log n))$.

2 A $1 \pm \varepsilon$ factor approximation algorithm for F_0

Observe that the Markov's inequality analysis of the upper bound on the probability of z being at least 3 times F_0 is actually asymptotically tight. In particular, prior to the median trick, it only takes a single bad event where the hash of an element hashes has too many zeroes to raise the estimate. (In fact, this estimate will be a factor of K larger than F_0 with probability $\Omega(1/K)$.) Moreover, the smallest granularity that the estimate can change is only a factor of 2.

The algorithm, due to Bar-Yossef, Jayram, Kumar, Sivakumar, and Trevisan, that we will describe for obtaining a $1 \pm \varepsilon$ factor approximation will be somewhat less aggressive in incrementing z – it will only increment z with the evidence of $\Omega(1/\varepsilon^2)$ elements – and will keep track of more detailed information about the hashes that have close to the maximum number of zeroes.

We first describe a variant of this algorithm that is a bit simpler to analyze but is less space efficient.

Large space $1 \pm \varepsilon$ Factor F_0 Estimation Algorithm

Initialize:
Choose $h : [M] \to [M]$ from H a pairwise independent family of hash functions
$z \leftarrow 0$
$B \leftarrow \emptyset$
Process:
for each i **do**
  if $\text{zeroes}(h(x_i)) \geq z$ **then**
    $B \leftarrow B \cup \{(x_i, \text{zeroes}(h(x_i))\}$
  end if
  while $|B| \geq c/\varepsilon^2$ **do**
    $z \leftarrow z + 1$
    Remove all (x_i, β) where $\beta < z$ from B
  end while
end for
Output: $\tilde{F}_0 = |B|2^z$.

Note: The constant c in this algorithm will be a large constant determined by the analysis. In this version it is unnecessary to store the number of zeroes in $h(x_i)$ but we do so to make it closer to the small space version.

It is easy to see that the space required by this algorithm is $O(\frac{1}{\varepsilon^2}(\log M + \log\log n))$ primarily in order to store the set B.

Analysis Let $z^* =$ the final value of z. Define $X_{x,r}$ and Y_r as in the case of the basic algorithm. As before we have $\mathbb{E}(Y_r) = F_0/2^r$ and $Var(Y_r) \leq \mathbb{E}(Y_r) = F_0/2^r$. Observe that the value of $|B|$ when the algorithm finishes is precisely Y_{z*} so the output $\tilde{F}_0 = Y_{z^*}2^{z^*}$.

We consider cases based on the value of z^*.

If $z^* = 0$ then no elements were ever removed from $B = Y_0$ so the output $Y_0 2^0 = Y_0$ is precisely F_0 and there is no error.

Now suppose that $z^* \geq 1$. Let $FAIL$ be the event that the algorithm produces an output $\tilde{F}_0$ that is not within a $1 \pm \varepsilon$ factor of F_0. By definition,

$$\begin{aligned} FAIL &\Leftrightarrow |\tilde{F}_0 - F_0| \geq \varepsilon F_0 \\ &\Leftrightarrow |Y_{z^*}2^{z^*} - F_0| \geq \varepsilon F_0 \\ &\Leftrightarrow |Y_{z^*} - F_0/2^{z^*}| \geq \varepsilon F_0/2^{z^*} \end{aligned}$$

Since $\mathbb{E}(Y_r) = F_0/2^r$ for each r, the task about bounding the probability of $FAIL$ looks a bit like bounding the deviation of Y_{z^*} from its expected value. However, the choice of z^* is not fixed and so $\mathbb{E}(Y_{z^*})$ will not in general be $F_0/2^{z^*}$. Instead, we break things up into cases depending on the value of z^* and use the value of F_0 to determine how we handle these cases.

Let s be the unique integer such that

$$\frac{12}{\varepsilon^2} \leq \frac{F_0}{2^s} < \frac{24}{\varepsilon^2}.$$

We will choose c large enough so that it will be very unlikely $z^* \geq s$ and handle the cases when

$z^* < s$ using our analysis based on the deviation. Since $1 \leq z^* \leq \log_2 M$

$$\begin{aligned}
\mathbb{P}[FAIL] &= \sum_{r=1}^{\log_2 M} \mathbb{P}[(|Y_r - \frac{F_)}{2^r}| \geq \frac{\varepsilon F_0}{2^r}) \wedge (z^* = r)] \\
&\leq \sum_{r=1}^{s-1} \mathbb{P}[|Y_r - \frac{F_0}{2^r}| \geq \frac{\varepsilon F_0}{2^r}] + \sum_{r=s}^{\log_2 M} \mathbb{P}[z^* = r] \\
&= \sum_{r=1}^{s-1} \mathbb{P}[|Y_r - \frac{F_0}{2^r}| \geq \frac{\varepsilon F_0}{2^r}] + \mathbb{P}[z^* \geq s] \\
&= \sum_{r=1}^{s-1} \mathbb{P}[|Y_r - \frac{F_0}{2^r}| \geq \frac{\varepsilon F_0}{2^r}] + \mathbb{P}[Y_{s-1} > c/\varepsilon^2]
\end{aligned}$$

which by Chebyshev's inequality and by Markov's inequality is

$$\begin{aligned}
&\leq \sum_{r=1}^{s-1} \frac{Var(Y_r)}{(\varepsilon F_0/2^r)^2} + \frac{\mathbb{E}(Y_{s-1})}{(c/\varepsilon^2)} \\
&\leq \sum_{r=1}^{s-1} \frac{2^r}{\varepsilon^2 F_0} + \frac{F_0 \varepsilon^2}{c 2^{s-1}} \\
&< \frac{2^s}{\varepsilon^2 F_0} + \frac{F_0 \varepsilon^2}{c 2^{s-1}} \\
&< 1/12 + 48/c
\end{aligned}$$

since $F_0/2^s \geq 12/\varepsilon^2$ for the first term and $F_0/2^s < 24/\varepsilon^2$ for the second. By choosing $c = 576$, the second term is also at most $1/12$ and the total failure probability is at most $1/6$.

In the small space algorithm, instead of storing each input x_i in B, we only store a hash of x_i using a separate pairwise independent hash function to a small set. A key observation is that, over the course of the algorithm execution, there are very few entries that will ever need to be stored: There are only $\log_2 M$ values of z and at most c/ε^2 different input values that will be hashed for each value of z. This yields a total of at most $K = c\varepsilon^{-2} \log_2 M$ values that will ever be hashed by this function over the course of the whole execution.

Lemma 2.1. *Let g be chosen from a 2-universal family of hash functions G that matp $[M] \to [aK^2]$. For any subset $A \subseteq [M]$ with $|A| \leq K$,*

$$\mathbb{P}[g \textit{ collides on } A] \leq \frac{1}{2a}.$$

Proof. By the 2-universal property, for all $i \in j \in [M]$,

$$\mathbb{P}[g(i) = (j)] = \frac{1}{aK^2}.$$

Therefore

$$\begin{aligned}\mathbb{P}[g \text{ collides on } A] &\leq \mathbb{E}(\#\text{ of collisions of } g \text{ on pairs in } A) \\ &\leq \binom{K}{2}\frac{1}{aK^2} \leq \frac{1}{2a}.\end{aligned}$$

□

The resulting algorithm is now:

$1 \pm \varepsilon$ **Factor** F_0 **Estimation Algorithm**

Initialize:
Choose $h : [M] \to [M]$ from H a pairwise independent family of hash functions
Choose $g : [M] \to [K']$ from G a 2-universal family of hash functions where $K' = \lceil 3c^2\varepsilon^{-4}\log^2 M \rceil$
$z \leftarrow 0$
$B \leftarrow \emptyset$
Process:
for each i **do**
  if $\text{zeroes}(h(x_i)) \geq z$ **then**
    $B \leftarrow B \cup \{(g(x_i), \text{zeroes}(h(x_i))\}$
  end if
  while $|B| \geq c/\varepsilon^2$ **do**
    $z \leftarrow z + 1$
    Remove all (α, β) where $\beta < z$ from B
  end while
end for
Output: $\tilde{F}_0 = |B|2^z$.

Except with failure probability of at most 1/6 over the choice of the function g, the estimate produced by this algorithm will be the same as that produced by the large space version. Combined with the analysis of that version this yields success probability at least 2/3. The space has been reduced to $O(\log M +$
$frac1\varepsilon^2(\log 1/\varepsilon + \log\log M + \log\log n))$.

This algorithm has success probability of only 2/3. We again compute the median of $O(\log(1/\delta))$ independent trials run in parallel to obtain a $1 \pm \varepsilon$ factor estimate of F_0, with probability at least $1 - \delta$. The final space bound is

$$O(\log(1/\delta(\log M + frac1\varepsilon^2(\log 1/\varepsilon + \log\log M + \log\log n))).$$

This bound is very close to the theoretical optimum. Even to achieve success probability 2/3, one needs space $\Omega(1/\varepsilon^2 + \log M)$. An algorithm achieving this was developed by Kane, Nelson, and Woodruff in a paper at PODS 2010. (This paper would be a good candidate for a presentation.)

The probably of estimating F_0 is useful in practice. Next time we will briefly discuss a simple practical algorithm HYPERLOGLOG due to Flajolet et al., which uses similar ideas but makes stronger assumptions about the hash function in order to get faster execution.