Sketching and streaming — Notes 2

Jelani Nelson
minilek@seas.harvard.edu

July 18, 2016

Distinct elements (F_0). In this note we will consider the *distinct elements problem*, also known as the F_0 problem, defined as follows. We are given a stream of integers $i_1, \ldots, i_m \in [n]$ where $[n]$ denotes the set $\{1, 2, \ldots, n\}$. We would like to output the number of *distinct* elements seen in the stream. As with Morris' approximate counting algorithm, our goal will be to minimize our space consumption.

There are two straightforward solutions as follows:

1. **Solution 1:** keep a bit array of length n, initialized to all zeroes. Set the ith bit to 1 whenever i is seen in the stream (n bits of memory).
2. **Solution 2:** Store the whole stream in memory explicitly ($\lceil m \lg n \rceil$ bits of memory).

We can thus solve the problem exactly using $\min\{n, \lceil m \lg n \rceil\}$ bits of memory.

1 Randomized approximation in small space

We will instead settle for computing some value $\tilde{t}$ s.t. $\mathbb{P}(|t - \tilde{t}| > \varepsilon t) < \delta$, where t denotes the number of distinct elements in the stream. The first work to show that this is possible using small memory (assuming oracle access to certain random hash functions) is due to Flajolet and Martin (FM) [FM85].

2 Idealized FM algorithm

1. Pick a random function $h : [n] \to [0, 1]$
2. Maintain counter $X = \min_{i \in stream} h(i)$
3. Output $1/X - 1$

Note this algorithm really is idealized, since we cannot afford to store a truly random such function h in $o(n)$ bits (first, because there are n independent random variables $(h(i))_{i=1}^n$, and second because its outputs are real numbers).

Intuition. Note that the value X stored by the algorithm is a random variable that is the minimum of t i.i.d $Unif(0, 1)$ random variables. The key claim is then the following, in which we suppose that the unique integers in the stream are $i_1, \ldots, i_t$.

Claim 1. $\mathbb{E}\, X = \frac{1}{t+1}$.

Proof.

$$\begin{aligned}
\mathbb{E}\, X &= \int_0^\infty \mathbb{P}(X > \lambda) d\lambda \\
&= \int_0^\infty \mathbb{P}(\forall i \in stream, h(i) > \lambda) d\lambda \\
&= \int_0^\infty \prod_{r=1}^{t} \mathbb{P}(h(i_r) > \lambda) d\lambda \\
&= \int_0^1 (1 - \lambda)^t d\lambda \\
&= \frac{1}{t+1}
\end{aligned}$$

□

We will also need the following claim in order to execute Chebyshev's inequality to bound the failure probability in our final algorithm.

Claim 2. $\mathbb{E}\, X^2 = \frac{2}{(t+1)(t+2)}$

Proof.

$$\begin{aligned}
\mathbb{E}\, X^2 &= \int_0^1 \mathbb{P}(X^2 > \lambda) d\lambda \\
&= \int_0^1 \mathbb{P}(X > \sqrt{\lambda}) d\lambda \\
&= \int_0^1 (1-\sqrt{\lambda})^t d\lambda \\
&= 2\int_0^1 u^t(1-u)du \qquad\qquad \text{(substitution } u = 1-\sqrt{\lambda}\text{)} \\
&= \frac{2}{(t+1)(t+2)}
\end{aligned}$$

□

This gives $Var[X] = \mathbb{E}\, X^2 - (\mathbb{E}\, X)^2 = \frac{t}{(t+1)^2(t+2)}$, or the simpler $Var[X] < (\mathbb{E}\, X)^2 = \frac{1}{(t+1)^2}$.

3 FM+

To obtain an algorithm providing a randomized approximate guarantee, just as with Morris+ we form an algorithm FM+ which averages together the outputs from s independent instantiations of the basic FM algorithm.

1. Instantiate $s = \lceil 1/(\varepsilon^2\delta) \rceil$ FMs independently, FM_1,...,FM_s.
2. Let X_i be the output of FM_i.
3. Upon a query, output $1/Z - 1$, where $Z = \frac{1}{s}\sum_i X_i$.

We have that $\mathbb{E}(Z) = \frac{1}{t+1}$, and $Var(Z) = \frac{1}{s}\frac{t}{(t+1)^2(t+2)} < \frac{1}{s(t+1)^2}$.

Claim 3. $\mathbb{P}(|Z - \frac{1}{t+1}| > \frac{\varepsilon}{t+1}) < \delta$

Proof. We apply Chebyshev's inequality.

$$\mathbb{P}(|Z - \frac{1}{t+1}| > \frac{\varepsilon}{t+1}) < \frac{(t+1)^2}{\varepsilon^2}\frac{1}{s(t+1)^2} = \eta$$

□

Claim 4. $\mathbb{P}(|(\frac{1}{Z}-1)-t| > O(\varepsilon)t) < \eta$

Proof. By the previous claim, with probability $1-\eta$ we have

$$\frac{1}{(1\pm\varepsilon)\frac{1}{t+1}} - 1 = (1\pm O(\varepsilon))(t+1) - 1 = (1\pm O(\varepsilon))t \pm O(\varepsilon)$$

□

4 FM++

To obtain our final algorithm, again as with Morris++ We take the median output from multiple independent instantiations of FM+.

1. Instantiate $q = \lceil 36\ln(2/\delta)\rceil$ independent copies of FM+ with $\eta = 1/3$.
2. Output the median $\hat{t}$ of $\{1/Z_j - 1\}_{j=1}^{q}$ where Z_j is the output of the jth copy of FM+.

Claim 5. $\mathbb{P}(|\widehat{t}-t| > \varepsilon t) < \delta$

Proof. Let

$$Y_j = \begin{cases} 1 & \text{if } |(1/Z_j - 1) - t| \le \varepsilon t \\ 0 & \text{else} \end{cases}$$

and put $Y = \sum_{j=1}^{q} Y_j$. We have $\mathbb{E}\,Y > 2q/3$ by our choice of η. The probability we seek to bound is equivalent to the probability that the median fails, i.e. at least half of the FM+ estimates have $Y_j = 0$. In other words,

$$\sum_{j=1}^{q} Y_j \le q/2$$

We then get that

$$\mathbb{P}(\sum Y_j \le q/2) \le \mathbb{P}(|Y - \mathbb{E}\,Y| > \frac{1}{4}\,\mathbb{E}\,Y) \tag{1}$$

Then by a Chernoff bound, the above is at most

$$2e^{-\frac{(\frac{1}{4})^2 2s/3}{3}} < \delta$$

as desired. □

The final space required, ignoring h, is that required to store $O(\frac{\lg(1/\delta)}{\varepsilon^2})$ real numbers since $O(\lg(1/\delta))$ copies of FM+ are instantiated, each averaging $O(1/\varepsilon^2)$ copies of FM. Each single FM just stores a single number (the minimum hash ever seen).

5 k-wise independent hash families

We next describe a modified algorithm for F_0-estimation which does not assume access to a truly random hash function $h : [n] \to [0, 1]$. Before we can continue, however, we must discuss k-wise independent hash families.

Definition 1. *A family $\mathcal{H}$ of functions mapping $[a]$ to $[b]$ is k-wise independent if $\forall j_1, \ldots, j_k \in [b]$ and $\forall$ distinct $i_1, \ldots, i_k \in [a]$,*

$$\mathop{\mathbb{P}}_{h \in \mathcal{H}}(h(i_1) = j_1 \wedge \ldots \wedge h(i_k) = j_k) = 1/b^k$$

Example. The set $\mathcal{H}$ of all functions $[a] \to [b]$ is k-wise independent for every k. $|\mathcal{H}| = b^a$ so $h \in \mathcal{H}$ is representable in $a \lg b$ bits.

Example. Let $a = b = q$ for $q = p^r$ a prime power and define $\mathcal{H}_{poly}$ to be the set of all degree $\leq k - 1$ polynomials with coefficients in $\mathbb{F}_q$, the finite field of order q. $|\mathcal{H}_{poly}| = q^k$ so $h \in \mathcal{H}$ is representable in $k \lg p = k \lg a$ bits.

Claim 6. *$\mathcal{H}_{poly}$ is k-wise independent.*

Proof. Given $((i_r, j_r))_{r=1}^k$, there is exactly one degree at most $k-1$ polynomial h over $\mathbb{F}_q$ with $h(i_r) = j_r$ for $r = 1, \ldots, k$. Thus the probability

$$\mathop{\mathbb{P}}_{h \in \mathcal{H}}(h(i_1) = j_1 \wedge \ldots \wedge h(i_k) = j_k)$$

exactly equals $1/\mathcal{H}_{poly} = 1/p^k = 1/b^k$. □

6 Non-idealized FM

We present an algorithm of [BYJK+02], later known as the "KMV algorithm" (for "k minimum values"). We will assume $1/\varepsilon^2 < n$, since we will be shooting for a space bound that is at least $1/\varepsilon^2$, and there is always a trivial solution

for exact F_0 computation using n bits. We also assume, without loss of generality, that $\varepsilon < 1/2$.

The algorithm is quite similar to the idealized FM algorithm, but rather than maintain only the *smallest* hash evaluation, we maintain the k smallest hash evaluations for some appropriately chosen $k \in \Theta(1/\varepsilon^2)$. The intuition is that the smallest hash evaluation leads us to an estimator with high variance (all it takes is for one item to hash to something really tiny, which will throw off our estimator). Meanwhile, if we consider the kth smallest hash evaluation for some large k, then the variance should be smaller since a single item (or small number of items) cannot throw off our statistic by having a wildly small or big hash value.

Specifically, choose a hash function $h : [n] \to [M]$ for $M = n^3$ from a 2-wise independent hash family (the idea here is to discretize $[0, 1]$ into multiples of $1/M$). We pick $k = \lceil 24/\varepsilon^2 \rceil$. We keep track in memory of the k smallest hash evaluations. If at the time of the query we have seen less than k distinct hash values, then we just output the number of distinct hash values seen. Otherwise, if X is the kth smallest then we output our estmate of $t = F_0$ as $\tilde{t} = kM/X$.

For some intuition: note if we had $t \geq k$ independent hash values in $[0, 1]$, we expect the kth smallest value v to be $k/(t+1)$ (namely, we expect k equally spaced values between 0 and 1). Thus a reasonable estimate for t would be $k/v - 1$. Since $t \geq k \gg 1/\varepsilon^2$, we expect $k/v > 1/\varepsilon^2$, and thus neglecting to subtract the 1 and simply outputting k/v gives the same answer up to a $1+\varepsilon$ factor. Since in our actual algorithm we discretize $[0, 1]$ into multiples of $1/M$, our value X is actually representing Mv, and thus we output kM/X. Furthermore, note by our choice of M that h perfectly hashes $[n]$ with high probability. Conditioned on that event, if we see less than k distinct hash values then we can be sure that our output is exactly correct.

We now provide a more formal analysis. Note that our hash function h is only 2-wise independent! We would like to say that with good probability,

$$(1 - \varepsilon)t \leq \tilde{t} \leq (1 + \varepsilon)t.$$

We consider the two bad events that $\tilde{t}$ is too big or too small, and show that each happens with probability at most $1/6$, and thus the probability that either happens is at most $1/3$ by the union bound.

First let us consider the case that $\tilde{t} > (1 + \varepsilon)t$. Since $\tilde{t} = kM/X$, i.e. $X = kM/\tilde{t}$, this can only happen if at least k distinct indices in the stream

hashed to a value smaller than $kM/((1+\varepsilon)t)$. Let Y_i be an indicator random variable for the event that the ith distinct integer in the stream hashed to a value below $kM/((1+\varepsilon)t)$, and let Y denote $\sum_{i=1}^{t} Y_i$. Then $\mathbb{E}\, Y_i < k/((1+\varepsilon)t)$ and thus

$$\mathbb{E}\, Y < k/(1+\varepsilon).$$

We also have $Var[Y_i] < \mathbb{E}\, Y_i^2 = \mathbb{E}\, Y_i < k/((1+\varepsilon)t)$, and thus

$$Var[Y] < k/(1+\varepsilon)$$

as well (note $Var[Y] = \sum_i Var[Y_i]$ since the Y_i are pairwise independent). Thus by Chebyshev's inequality,

$$\mathbb{P}(Y \geq k) \leq \mathbb{P}(|Y - \mathbb{E}\, Y| > k(1-1/(1+\varepsilon)) < \frac{k}{1+\varepsilon} \cdot \frac{(1+\varepsilon)^2}{k^2\varepsilon^2} = \frac{1+\varepsilon}{\varepsilon^2 k} < 1/6.$$

We can similarly bound the probablity that $\tilde{t} < (1-\varepsilon)t$. This can only happen if there weren't *enough* distinct indices in the stream that hashed to small values under h. Specifically, let Z_i be an indicator random variable for the event that the ith distinct integer in the stream hashed to a value below $kM/((1-\varepsilon)t)$, and define $Z = \sum_i Z_i$. Then $\tilde{t} < (1-\varepsilon)t$ can only occur if $Z < k$. But we have $k/((1-\varepsilon)t) \geq \mathbb{E}\, Z_i > k/((1-\varepsilon)t) - 1/M \geq (1+\varepsilon)k/t - 1/M$ (the $1/M$ is due to rounding). We note $1/M < \varepsilon k/(4t)$ since $\varepsilon > 1/\sqrt{n}$ and $t < n$. Thus $k/((1-\varepsilon)t) \geq \mathbb{E}\, Z_i > (1+3\varepsilon/4)k/t$, implying

$$k/(1-\varepsilon) \geq \mathbb{E}\, Z > (1+3\varepsilon/4)k$$

and also since Z is a sum of pairwise independent Bernoulli, again

$$Var[Z] \leq \mathbb{E}\, Z \leq k/(1-\varepsilon).$$

Thus by Chebyshev's inequality,

$$\mathbb{P}(Z < k) < \mathbb{P}(|Z - \mathbb{E}\, Z| > (3/4)\varepsilon k) < \frac{k}{1-\varepsilon} \cdot \frac{16}{9\varepsilon^2 k^2} < \frac{16}{9(1-\varepsilon)} \cdot \frac{1}{\varepsilon^2 k} < 1/6,$$

as desired.

Other comments. It is known, via a different algorithm, that for constant failure probability space $O(1/\varepsilon^2 + \log n)$ is achievable [KNW10], and furthermore this is optimal [AMS99, Woo04] (also see [TSJ08]). An algorithm that is more commonly used in practice is HyperLogLog [FEFGM07]. Although it assumes access to a truly random hash function, and asymptotically uses a factor $\lg\lg n$ more space than the algorithm of [KNW10], its performance in practice is superior.

References

[AMS99] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

[BYJK+02] Z. Bar-Yossef, T.S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *RANDOM*, pages 1–10, 2002.

[FEFGM07] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorith. In *Proceedings of the 2007 International Conference on the Analysis of Algorithms (AoFA)*, 2007.

[FM85] Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, 1985.

[KNW10] Daniel M. Kane, Jelani Nelson, and David P. Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 41–52, 2010.

[TSJ08] D. Sivakumar T. S. Jayram, Ravi Kumar. The one-way communication complexity of hamming distance. *Theory of Computing*, 4(1):129–135, 2008.

[Woo04] David P. Woodruff. Optimal space lower bounds for all frequency moments. In *SODA*, pages 167–175, 2004.