Estimating Distinct Elements in a Data Stream

Among practical algorithms, 3 are state of the art

- Hyperloglog [Flajolet et al. 2007]
- K-minimum values (KMV)
- Adaptive Sampling

} space usage can all be made $O\left(\log n + \frac{1}{\epsilon^2}\cdot\left(\log\left(\frac{1}{\epsilon}\right) + \log\log n\right)\right)$ for $\delta = \frac{1}{50}$ (say).

- I will describe KMV + Adaptive Sampling as having space usage $O\left(\frac{\log n}{\epsilon^2}\right)$. I will explain why shortly.

In theory: [KNW] gave an $O\left(\log n + \frac{1}{\epsilon^2}\right)$ space algorithm with $O(1)$ update time.

KMV: . Let $h : [n] \to [0,1]$ be a random function.

- Track the k-smallest hash values observed in the stream. Let m_k denote k'th smallest hash value seen.
- Output the estimate $\frac{k-1}{m_k}$.

Adaptive sampling! Let $h : [n] \to [0,1]$ be a random hash function.

- Initialize $i \leftarrow 0$.
- While processing stream, store all hash values $< 2^{-i}$. If more than k hash values are stored, set $i \leftarrow i+1$.
- If S is set of hash values stored at end of stream, output $|S| \cdot 2^i$

Facts about KMV and Adaptive Sampling.

- KMV has a natural min-heap based implementation. However, this requires $O(\log k)$ time per stream update, and doubles its space usage compared to just keeping the k hash values in a hash table. Adaptive Sampling is faster ($O(1)$-amortized update time) and doesn't have the space cost of a min-heap. But the error of Adaptive Sampling oscillates due to periodic purges.

- Both KMV and Adaptive Sampling are unbiased. We will prove this for KMV
- $\mathrm{Var}[\mathrm{KMV}] \le \dfrac{F_0^2}{K-1}$
- $\mathrm{Var}[\text{Adaptive Sampling}] \approx \dfrac{1.44\, F_0^2}{K-1}$.

We will not prove either of these facts. Here we note that by Chebyshev, the variance bounds mean setting $K = O(\frac{1}{\epsilon^2})$ is enough for an (ϵ,δ)-multiplicative approximation

i.e. $\Pr[|\mathrm{KMV} - F_0| > \epsilon F_0] \le \frac{1}{4}$.

$$\sigma(\mathrm{KMV} \text{ with } K = \tfrac{4}{\epsilon^2}+1 \text{ counters}) \lesssim \frac{F_0}{\sqrt{4/\epsilon^2}} = \frac{\epsilon}{2}\cdot F_0$$

- What is the space usage of KMV and adaptive sampling?

- It turns out it is enough for the hash function h to be from a pairwise independent hash family mapping $[n]$ to a set of size $O(\frac{1}{\epsilon^4}\log^2 n)$. Hence, h requires $O(\log n + \log(\frac{1}{\epsilon}))$ bits to represent, and each of the $O(\frac{1}{\epsilon^2})$ hash values take $O(\log(\frac{1}{\epsilon}) + \log\log n)$ bits to represent, for a final space bound of $O(\log n + \frac{1}{\epsilon^2}\cdot(\log(\frac{1}{\epsilon}) + \log\log n))$.

- In practice, it can be very useful to store not just the hash values but also the corresponding identifiers. If you do this, the space usage is $O(\frac{\log n}{\epsilon^2})$. This lets you, e.g. obtain estimates for ad hoc subsets of users, where the subset of interest is only determined at query time.

Hyperloglog:

- Let h be a random function mapping $[n]$ to $[0,1]$
- Let g be a random function mapping $[n]$ to $[k]$
- For each $j \in [k]$, track $v_j := \max_{i: g(a_i) = j} \text{zeros}(h(a_i))$
- Let Z be the harmonic mean of 2^{v_j}

- Output $\alpha \cdot k \cdot Z$. (α is a constant meant to correct a small bias in $k \cdot Z$

Intuition: 2^{v_j} should be about $\frac{F_0}{k}$, so $k \cdot Z$ should be about F_0.

To get an (ϵ, δ)-multiplicative approximation, can take $k \approx \frac{1}{\epsilon^2}$, so total space usage is $O(\log n + \frac{\log\log n}{\epsilon^2})$.

Proof that KMV is unbiased. For any stream σ,

Fix any $j \in [n]$ appearing one or more times in the stream.

Define $V_j := \begin{cases} \frac{1}{m_k} & \text{if } h(j) \text{ is one of the } k\text{-smallest hash values observed} \\ 0 & \text{otherwise.} \end{cases}$

Let H_{-j} denote the $F_0 - 1$ hash values of all other items in the stream other than j.

Let m_k^j denote the kth smallest hash value in H_{-j}, and m'_{k-1} the $(k-1)$'st smallest.

Then $\mathbb{E}[V_j \mid H_{-j}] = \Pr_h[h(j) < m'_{k-1}] \cdot \frac{1}{m'_{k-1}}$

$+ \Pr[m'_{k-1} < h(j)] \cdot 0$

$= m'_{k-1} \cdot \frac{1}{m'_{k-1}} = 1$

Note the KMV estimate is $\sum_j V_j$, so $\mathbb{E}[\text{KMV estimate}] \overset{\text{linearity of expectation}}{=} \sum_j \mathbb{E}[V_j] = \sum_{j: f_j > 0} 1 = F_0$

Analysis of Adaptive Sampling with $K = \frac{c}{\epsilon^2}$ (c to be specified later)

Let $X_{r,j} = \begin{cases} 1 & \text{if } h(j) \le 2^{-r} \\ 0 & \text{otherwise} \end{cases}$

and let $Y_r = \sum_{j: f_j > 0} X_{r,j}$. Let i_{max} denote the value of i at the end

So the returned estimate is $\hat{F_0} = |S| \cdot 2^{i_{max}}$ (S = set of hash values less than $2^{-i_{max}}$.)

Exactly as in last lecture, we obtain

Fact 1: $\mathbb{E}[Y_r] = \frac{F_0}{2^r}$, Fact 2: $Var[Y_r] \le \frac{F_0}{2^r}$.

Note: if $i_{max} = 0$ then the algorithm never saw more than K hash values so it returns exactly $|S| \cdot 1 = F_0$.

Otherwise, we need to bound the probability $|Y_{i_{max}} \cdot 2^{i_{max}} - F_0| \ge \epsilon F_0$

$\Leftrightarrow \left| Y_{i_{max}} - \frac{F_0}{2^{i_{max}}} \right| \ge \frac{\epsilon F_0}{2^{i_{max}}}$. $\leftarrow$ Call this event FAIL.

Let s be the unique integer s.t. $\frac{12}{\epsilon^2} \le \frac{F_0}{2^s} \le \frac{24}{\epsilon^2}$.

Then $Pr[FAIL] = \sum_{r=1}^{\infty} Pr\left[\left| Y_r - \frac{F_0}{2^r} \right| \ge \frac{\epsilon F_0}{2^r} \text{ and } i_{max} = r \right]$

$\le \left(\sum_{r=1}^{s-1} Pr\left[\left| Y_r - \frac{F_0}{2^r} \right| \ge \frac{\epsilon F_0}{2^r} \right] \right) + \left(\sum_{r \ge s} Pr[i_{max} = r] \right)$

$= \quad \Downarrow$

$\le \quad \Downarrow \qquad + \; Pr[i_{max} \ge s]$

$+ \; Pr\left[Y_{s-1} \ge \frac{K}{\epsilon^2} \right]$

By Chebyshev and Fact 2, ~~the first term is~~

$$\Pr\left[\left|Y_r - \frac{F_0}{2^r}\right| \ge \frac{\epsilon F_0}{2^r}\right] \le \frac{2^r}{\epsilon^2 F_0}$$

So the first term is at most

$$\sum_{r=1}^{s-1} \frac{2^r}{\epsilon^2 F_0} \le \frac{2^s}{\epsilon^2 F_0} \le \frac{1}{12} \qquad 2^s \le \frac{\epsilon^2}{12}$$

By Markov's inequality and Fact 1, the second term is at most

$$\frac{E[Y_{s-1}]}{c/\epsilon^2} = \underbrace{\frac{F_0}{2^{s-1}}}_{\le \frac{48}{\epsilon^2}} \cdot \frac{\epsilon^2}{c} \le \frac{48}{c} \le \frac{1}{12} \text{ if we choose } c=4$$

In total, $\Pr[\text{Fail}] \le \frac{1}{12} + \frac{1}{12} = \frac{1}{6}$ so this is an $(\epsilon, \frac{1}{6})$-approximation algorithm