Lecture 10

AMS estimator and estimation F_2

Recall we want to estimate frequency moment F_k. Let $\sigma = e_1, \ldots, e_m \in [n]^m$.

$$F_k = \sum_{i=1}^{n} (f_i)^k$$

We will consider a more abstract and general estimation problem. Let $g_i : \mathbb{Z}_{\geq 0} \to \mathbb{R}$ be a real valued function with $g_i(0) = 0$. Say we want to estimate

$$g(\sigma) = \sum_{i=1}^{n} g_i(f_i)$$

Note that we are allowing different functions for different i.

$$F_k = \sum_{i=1}^{n} g(f_i) \text{ where } g(x) = x^k.$$

Alon-Matias-Szegedy (AMS) in their influential paper provided an unbiased estimator for computing $g(\sigma)$.

AMS-Estimator (g)

- Sample e_J uniformly at random from stream. Say $e_J = i$ where $i \in [n]$.
- Let $R = |\{j \mid J \leq j \leq m, e_j = i\}|$.
- Output $m \cdot (g_i(R) - g_i(R-1))$.

Implementation via Reservoir Sampling

$s \leftarrow$ null
$m \leftarrow 0$
$R \leftarrow 0$
while (stream is not empty)
$m \leftarrow m + 1$
$e_m \leftarrow$ current item
If $(s == e_m)$ $R \leftarrow R + 1$
with prob $\frac{1}{m}$
$s \leftarrow e_m$
$R \leftarrow 1$
end while
Output $m \cdot (g_i(R) - g_i(R-1))$ where $s = i$.

Analysis

Lemma: Let Y be the output of the algorithm. Then $E[Y] = g(\sigma)$.

Proof:

$$\begin{aligned} E[Y] &= m \sum_{i=1}^{n} E[Y|e_J = i] \Pr[e_J = i] \\ &= m \sum_{i=1}^{n} \frac{f_i}{m} E[Y|e_J = i] \\ &= m \sum_{i=1}^{n} \frac{f_i}{m} \cdot \sum_{j=1}^{f_i} \frac{g_i(j) - g_i(j-1)}{f_i} \\ &= \sum_{i=1}^{n} g_i(f_i) \end{aligned}$$

Thus we can use AMS estimator for F_k as well. But we need to understand the variance of the estimator.

Lemma: For $g(x) = x^k$, we have

$$Var(Y) \leq kF_1F_{2k-1} \leq kn^{1-\frac{1}{k}}F_k^2$$

Proof: $Var(Y) \leq E[Y^2]$

$$\begin{aligned} E[Y^2] &= \sum_{i=1}^{n} \Pr[e_J = i] \cdot \sum_{l=1}^{f_i} \frac{m^2}{f_i} (l^k - (l-1)^k)^2 \\ &\approx \sum_{i=1}^{n} \frac{f_i}{m} \cdot \frac{m^2}{f_i} \sum_{l=1}^{f_i} (l^k - (l-1)^k)^2 \\ &\leq F_1 \sum_{i=1}^{n} \sum_{l=1}^{f_i} kl^{k-1}(l^k - (l-1)^k) \\ &\leq kF_1 \sum_{i=1}^{n} f_i^{k-1} f_i^k \\ &\leq kF_1F_{2k-1} \end{aligned}$$

$F_1F_{2k-1} = (\sum_{i=1}^{n} f_i)(\sum_{i=1}^{n} f_i^{2k-1}) \leq n^{1-\frac{1}{k}}F_k^2$
since $Var(Y) \leq kn^{1-\frac{1}{k}}F_k^2$
and $E[Y] = F_k$.

Using median trick we can get an (ϵ, δ)-approximation in space $O(k\frac{1}{\epsilon^2}n^{1-\frac{1}{k}} \log \frac{1}{\delta})$.

F_k / F_2 estimation

For $k = 2$, we get (ϵ, δ)-approx in $\tilde{O}(\frac{\sqrt{n}}{\epsilon^2} \log \frac{1}{\delta})$ space. Can we do better? For F_2, AMS showed that $polylog(n)$ suffices! For $k > 2$ the right bound is $\tilde{O}(n^{1-2/k})$. For $0 < k \leq 2$ we can get $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta} polylog(n))$.

AMS F_2 Estimation

- Choose $h : [n] \to \{-1, 1\}$ from a 4-wise independent hash family H.
- $z \leftarrow 0$
- **while** (stream is not empty)
  e_m is current element
  $z \leftarrow z + h(e_m)$
- **end while**
- Output z^2.

A 4-wise independent family can be stored via $O(1) \log n$ bit numbers.

Analysis

Let $Y_i = h(i)$. $E[Y_i] = 0$, $E[Y_i^2] = 1$. $Y_i \in \{-1, 1\}$. $Y_1, \ldots, Y_n$ are 4-wise independent. $Z = \sum_{i=1}^{n} f_i Y_i$. Output is Z^2.

$$\begin{aligned} E[Z^2] &= E\left[\left(\sum_{i=1}^{n} f_i Y_i\right)^2\right] = E\left[\sum_i f_i^2 Y_i^2 + \sum_{i \neq j} f_i f_j Y_i Y_j\right] \\ &= \sum_{i=1}^{n} f_i^2 E[Y_i^2] + \sum_{i \neq j} f_i f_j E[Y_i Y_j] \\ &= \sum_{i=1}^{n} f_i^2 = F_2 \end{aligned}$$

$Var(Z^2) = E[Z^4] - (E[Z^2])^2 = E[Z^4] - F_2^2$.

$$E[Z^4] = E\left[\sum_i \sum_j \sum_k \sum_l f_i f_j f_k f_l Y_i Y_j Y_k Y_l\right]$$

Any term with only one occurrence of a term Y_i becomes 0.

$$E[Z^4] = \sum_{i=1}^{n} f_i^4 E[Y_i^4] + 6 \sum_{i=1}^{n} \sum_{j=i+1}^{n} f_i^2 f_j^2 E[Y_i^2 Y_j^2] = \sum_{i=1}^{n} f_i^4 + 6 \sum_{i<j} f_i^2 f_j^2$$

$$\begin{aligned} Var(Z^2) &= \sum_{i=1}^{n} f_i^4 + 6 \sum_{i=1}^{n} \sum_{j=i+1}^{n} f_i^2 f_j^2 - \left(\sum_{i=1}^{n} f_i^2\right)^2 \\ &= \sum_{i=1}^{n} f_i^4 + 6 \sum_{i<j} f_i^2 f_j^2 - \left(\sum_{i=1}^{n} f_i^4 + 2 \sum_{i<j} f_i^2 f_j^2\right) \\ &= 4 \sum_{i<j} f_i^2 f_j^2 \leq 2 \left(\sum f_i^2\right)^2 = 2F_2^2 \end{aligned}$$

An (ϵ, δ)-approximation requires $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ counters. We make an observation that non-negativity of f_i did not play a role in the proof. This will lead us to a generalization of the streaming model. Recall we had $e_t \in [n]$ for each time t. Now we have $e_t = (i_t, \Delta_t)$ where $i_t \in [n]$ and Δ_t is an update to coordinate i_t.

We use $x \in \mathbb{R}^n$ that starts at $\vec{0}$ and is updated as

$$x_{i_t} = x_{i_t} + \Delta_t$$

after e_t. Now F_2 becomes $||x||_2^2$. We see that the AMS-F_2 estimator works in this model.

AMS F_2 Estimation (Generalized)

- Choose $h : [n] \to \{-1, 1\}$ from a 4-wise independent hash family H.
- $z \leftarrow 0$
- **while** (stream is not empty)
  $e_t \leftarrow (i_t, \Delta_t)$
  $z \leftarrow z + \Delta_t h(i_t)$
- **end while**
- Output z^2.

Exercise: Show $E[Z^2] = ||x||_2^2$ and $Var(Z^2) \leq 2||x||_2^4$. $||x||_2^2$ is the length of the vector x.

Should remind you of dimensionality reduction! Interpreting AMS-F_2 estimator as a linear sketch. We can view the streaming computation as

$$\begin{pmatrix} +1 & -1 & +1 & \dots & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = z$$

where the row vector is obtained from h.

Recall that in dimensionality reduction we picked a $k \times n$ matrix A where we chose A_{ij} as an independent Gaussian. In F_2 estimation we are picking A where each row of A is obtained from a 4-wise independent hash function with entries in $\{-1, +1\}$. In dimensionality reduction we chose $k = \Theta(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ and showed that $||\frac{1}{\sqrt{k}} Ax||_2$ is an (ϵ, δ)-approximation to $||x||_2$. But in F_2 estimation we seem to be getting the same! $\Theta(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ rows suffice to get an ϵ-approximation with probability $(1 - \delta)$. But we are only using 4-wise independence in each row. Why not use this for dimensionality reduction?

The difference is the following. Ax with $k = \Theta(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ has sufficient information to recover a $(1 \pm \epsilon)$-approximation for $||x||_2$ but $||Ax||_2$ is itself not the way we compute the approximation. We use a median estimator which is not a

linear function. Nevertheless the information that the algorithm computes is a linear sketch, Ax.

A **sketch** of a data stream σ is some function $C(\sigma)$ that is a compact representation of σ. We want sketches to have **comparability**. Given σ_1 and σ_2 and sketches $C(\sigma_1)$ and $C(\sigma_2)$, we would like to compute $C(\sigma_1 \cdot \sigma_2)$ from $C(\sigma_1)$ and $C(\sigma_2)$. A particularly nice sketch is a **linear sketch**, where $C(\sigma_1 \cdot \sigma_2) = C(\sigma_1) + C(\sigma_2)$.

The F_2 estimation can be seen as a linear sketch.

$$Ax = \begin{pmatrix} \longleftarrow & h_1 & \longrightarrow \\ \longleftarrow & h_2 & \longrightarrow \\ & \vdots & \\ \longleftarrow & h_k & \longrightarrow \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

Each row corresponds to a hash function. Note that the way we use the output of the sketch to compute some information about the data can be some non-linear function of the sketch itself. Linear sketches naturally allow for deletions.