CS761 Derandomization and Pseudorandomness **2022-23 Sem I**

Lecture 5: 22/08/22

Scribe: Raj Aryan Agrawal *Lecturer: Rohit Gurjar*

Let us see a problem in the streaming setting where it turns out that the use of pseudorandomness is essential (as opposed to true randomness). Suppose you want to count the number of distinct visitors to your website in a day. However, you do not have enough space to store the names of all the visitors. Let us define the problem formally.

Streaming Data

Problem 5.1 (Number of distinct elements)**.** *The problem is to find number of distinct elements in a streaming data. More formally, given a sequence of values*

$$a_1, a_2, \ldots, a_m \in \{1, 2, \ldots N\}$$

We wish to find the number of distinct elements in the sequence using only $O(\log N)$ space.

As one can guess, to find the exact count of distinct elements, one has to use at least $O(m)$ or $O(N)$ space (whichever is smaller). So, we will have to relax the question to only finding an approximate count. Even an approximate count cannot be obtained via a deterministic algorithm (see for example, `https://people.csail.mit.edu/rrw/6.045-2020/notestream.pdf`). Hence, to make it work in only $O(\log N)$ space, we must use randomness.

Let us first see a simple randomized algorithm to the problem that assumes we have a hash function distributed uniformly over $[0, 1]$

Independent Hash Values

Algorithm 5.2. *Suppose we are given $h : \{1, 2, \ldots, N\} \to [0, 1]$, where each $h(i)$ is uniformly randomly and independently assigned from the range $[0, 1]$. Set $M = \infty$ and then*

$$\textit{For each } a_i \textit{ in stream } : \; M \leftarrow \textit{min}(M, h(a_i))$$

Finally output $\frac{1}{M} - 1$.

The intution for this algorithm comes from the following claim.

Claim 5.3.

$$\mathbb{E}[M] = \frac{1}{d+1}$$

where d is the number of distinct elements

Proof. We first find the probability distribution of M. Let S be the set of elements which appear in the stream, so $|S| = d$.

$$\begin{aligned}
\Pr[M > \lambda] &= \Pr[\forall i \in S, h(i) > \lambda] \\
&= \prod_{i \in S} \Pr(h(i) > \lambda) && \text{independence of values of } h(i) \\
&= (1 - \lambda)^d
\end{aligned}$$

Let the probability density function for M be $f(\lambda)$. Then,

$$\begin{aligned} f(\lambda) &= \frac{d}{d\lambda}\Pr[M \leq \lambda] \\ &= \frac{d}{d\lambda}(1 - Pr[M > \lambda]) \\ &= -\frac{d}{d\lambda}(1-\lambda)^d \\ &= d(1-\lambda)^{d-1} \end{aligned}$$

Thus, the expected value is

$$\begin{aligned} \mathbb{E}[m] &= \int_0^1 \lambda f(\lambda) d\lambda \\ &= \int_0^1 \lambda d(1-\lambda)^{d-1} d\lambda \\ &= d\int_0^1 (1-\lambda)\lambda^{d-1} d\lambda && \text{by substitution} \\ &= d\left[\frac{\lambda^d}{d} - \frac{\lambda^{d+1}}{d+1}\right]_0^1 \\ &= \frac{1}{d+1} \end{aligned}$$

□

Some issues with the algorithm. One obvious issue is that one cannot obtain a real number randomly uniformly from $[0, 1]$. But, that is not a big issue. We can suitably discretize the range, say for example, $\{0, 1/k, 2/k, \ldots, (k-1)/k\}$. However, the main issue is the generation of $h(i)$. Whenever we generate $h(i)$, we will need to store it for future use when the ith item comes again. That would mean we will need to store at least $O(d)$ bits. But, we do not have such large space.

To fix this, we need a function h such that all $h(i)$ values can be generated using only small amount of storage. However, $h(i)$ should still remain randomly uniform in a certain range. And that brings us to pseudorandomness. The plan is to first generate some small amount of random bits ($O(\log N)$) and store it, before the stream starts. Then whenever we see the ith item, we will compute $h(i)$ using i and the stored random bits. Naturally, now the random values $\{h(1), h(2), \ldots, h(N)\}$ are not completely independent. However, there will be enough independence that will suffice for our purposes. In fact, only pairwise independence will suffice.

Reducing number of random bits

Let $\mathbb{F}$ be a finite field of size $\geq N$.

Algorithm 5.4. *We define our hash function using 2 pre-computed random values $a, b \in_R \mathbb{F}$. Define*

$$h(i) = ai + b \text{ (multiplication and addition is over } \mathbb{F}.)$$

Initialize $M \leftarrow \infty$.

For each a_j in stream : $M \leftarrow min(M, h(a_j))$

Finally output $\frac{N}{M} - 1$.

Note that the min operation is applied while treating the values $h(a_j)$ as integers in the range $\{1, 2, \ldots, N\}$ (min does not make sense over a finite field). Observe that this algorithm only uses $O(\log N)$ bits of randomness. From the previous lectures, we know that the values $\{h(i)\}_i$ are pairwise independent. Recall that in the above analysis, while computing expectation we needed complete independence of the $h(i)$variables.

Now, we cannot reason about $\mathbb{E}[M]$ in the same way. Instead, we shall show that the value of M lies close to what is expected with a decent probability. That is, we wish to show

$$(1-\epsilon)\frac{N}{d+1} \leq M \leq (1+\epsilon)\frac{N}{d+1} \text{ with } \textit{Probability} > \delta,$$

for some reasonable constants ϵ, δ.

Rephrasing the problem

Suppose L and U are some numbers and we wish to show

$$L \leq M \leq U$$

with good probability. We can say that

$M > L$ if and only if no element appearing in the stream is mapped to $\leq L$
$M \leq U$ if and only if at least one element appearing in the stream is mapped to $\leq U$

Define

$$Y_{i,\lambda} = \begin{cases} 1 & \text{if } h(i) \leq \lambda \\ 0 & \text{otherwise} \end{cases}$$

and

$$Y_\lambda = \sum_{i \in S} Y_{i,\lambda}$$

where S is the set of elements which appear in the stream. Clearly, Y_λ is the number of elements in the stream which are mapped to $\leq \lambda$. Then we can write

$M > L$ if and only if $Y_L = 0$ and
$M \leq U$ if and only if $Y_U \geq 1$.

We now compute the probabilities for each of these events.

Left Condition: $M > L$

We will use $\mathbb{E}[Y_L]$ to reason about $\Pr[Y_L = 0]$.

$$\mathbb{E}[Y_{i,L}] = \Pr(h(i) \leq L) = \frac{L}{N}.$$

Then

$$\begin{aligned} \mathbb{E}[Y_L] &= \mathbb{E}\left[\sum_{i \in S} Y_{i,L}\right] \\ &= \sum_{i \in S} \mathbb{E}[Y_{i,L}] && \text{(by linearity of expectation)} \\ &= \sum_{i \in S} \frac{L}{N} = \frac{dL}{N} && (|S| = d). \end{aligned}$$

Now, consider Markov's inequality.

Theorem 5.5 (Markov's Inequality)**.** *If Z is a non negative random variable then,*

$$\Pr(Z \geq \alpha) \leq \frac{\mathbb{E}[Z]}{\alpha}$$

Proof.

$$\begin{aligned}
\mathbb{E}[Z] &= \int_0^\infty z f(z) dz \\
&= \int_0^\alpha z f(z) dz + \int_\alpha^\infty z f(z) dz \\
&\geq \int_\alpha^\infty \alpha f(z) dz \\
&= \alpha \Pr(Z \geq \alpha)
\end{aligned}$$

□

Using the above theorem, we have

$$Pr(Y_L \geq 1) \leq \frac{\mathbb{E}[Y_L]}{1} = \frac{dL}{N}.$$

Substituting $L = (1-\epsilon)\frac{N}{d}$, we have

$$\Pr\left(M > (1-\epsilon)\frac{N}{d}\right) = \Pr[Y_L = 0] = 1 - \Pr(Y_L \geq 1) \geq 1 - \frac{dL}{N} = \epsilon$$

Right Condition: $M \leq U$

Here $U = (1+\epsilon)\frac{N}{d}$, we have

$$\mathbb{E}[Y_U] = 1 + \epsilon$$

Using just the expected value, we cannot get a good lower bound for $\Pr(Y_U \geq 1)$. We also require some knowledge for the variance. That is, we want to reason about $Var[Y_U] = Var[\sum_{i \in S} Y_{i,U}]$.

Theorem 5.6. *Consider random variables $X_1, X_2, \ldots, X_n$, which are pairwise independent. Then*

$$Var\left[\sum_{i=1}^n X_i\right] = \sum_{i=1}^n Var[X_i].$$

Proof.

$$\begin{aligned}
Var\left[\sum_{i=1}^n X_i\right] &= \mathbb{E}\left[\left(\sum_{i=1}^n X_i\right)^2\right] - \mathbb{E}\left[\left(\sum_{i=1}^n X_i\right)\right]^2 \\
&= \sum_{i=1}^n \mathbb{E}[X_i^2] + 2\sum_{i \neq j} \mathbb{E}[X_i X_j] - \sum_{i=1}^n \mathbb{E}[X_i]^2 - 2\sum_{i \neq j} \mathbb{E}[X_i]\mathbb{E}[X_j] \\
&\qquad \text{by linearity of expectation} \\
&= \sum_{i=1}^n \left(\mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2\right) + \sum_{i \neq j} \left(\mathbb{E}[X_i X_j] - \mathbb{E}[X_i]\mathbb{E}[X_j]\right) \\
&= \sum_{i=1}^n Var[X_i] + 0 \\
&\qquad \text{since pairwise independent, } \mathbb{E}[X_i X_j] = \mathbb{E}[X_i]\mathbb{E}[X_j].
\end{aligned}$$

□

Now, as mentioned before, our construction for h ensures that the variables $h(i)$ are pairwise independent. Thus, we can use the above theorem for our use. Since $Y_{i,U}$ is a Bernoulli random variable, we have

$$Var[Y_{i,U}] = \Pr[Y_{i,U} = 1](1 - \Pr[Y_{i,U} = 1]) = \frac{U}{N}\left(1 - \frac{U}{N}\right).$$

Since $U = (1+\epsilon)\frac{N}{d}$, we get

$$\begin{aligned} Var[Y_U] &= \sum_{i \in S} Var[Y_{i,U}] \\ &= \sum_{i \in S} \frac{1+\epsilon}{d}\left(1 - \frac{1+\epsilon}{d}\right) \\ &= (1+\epsilon)\left(1 - \frac{1+\epsilon}{d}\right) \\ &\leq (1+\epsilon) \end{aligned}$$

Now consider Chebyshev's inequality.

Theorem 5.7 (Chebyshev's Inequality)**.** *For any random variable X*

$$Pr(|X - \mathbb{E}[X]| \geq \alpha) \leq \frac{Var[X]}{\alpha^2}.$$

Proof.

$$\begin{aligned} Pr(|X - \mathbb{E}[X]| \geq \alpha) &= Pr((X - \mathbb{E}[X])^2 \geq \alpha^2) \\ &\leq \frac{\mathbb{E}[(X - \mathbb{E}[X])^2]}{\alpha^2} && \text{using Markov's Inequality} \\ &= \frac{Var[X]}{\alpha^2} \end{aligned}$$

□

Using this inequality, we have

$$\begin{aligned} \Pr[Y_U = 0] &= \Pr(|Y_U - \mathbb{E}[Y_U]| \geq \mathbb{E}[Y_U]) \\ &\leq \frac{Var[Y_U]}{\mathbb{E}[Y_U]^2} \\ &\leq \frac{(1+\epsilon)}{(1+\epsilon)^2} \\ &\leq \frac{1}{1+\epsilon}. \end{aligned}$$

Thus, we have

$$\Pr\left(M \leq (1+\epsilon)\frac{N}{d}\right) = \Pr[Y_U \geq 1] = 1 - \Pr[Y_U = 0] \geq \frac{\epsilon}{1+\epsilon}.$$

Combining the 2 results, and taking the union upper bound, we have

$$\Pr\left((1-\epsilon_1)\frac{N}{d} < M \leq (1+\epsilon_2)\frac{N}{d}\right) \geq \epsilon_1 + \frac{\epsilon_2}{1+\epsilon_2} - 1.$$

Taking $\epsilon_1 = 2/3$ and $\epsilon_2 = 2$, we get that

$$\Pr\left(\frac{1}{3}\frac{N}{d} < M \leq \frac{3N}{d}\right) \geq 1/3.$$

This is not a very good probability bound, so to improve the confidence, we can run multiple instances of the algorithm. Since the data is streaming, we can only visit it once, so we can run the multiple instances concurrently. And finally choose the **median** of the results - median will likely give a better answer than the mean as less likelihood that the median will skew, unlike mean which can due to just a single bad seed.