Streaming III : Estimating F_k and Reservoir Sampling[1]

- **Estimating F_k in Insert Only Streams.** In this lecture we look at another estimation algorithm from the Alon-Mattias-Szegedy paper which we mentioned last time. We will estimate $F_k := \sum_{i=1}^{n} \mathbf{f}_i^k$, for $k \geq 2$. Unfortunately, the space needed will be nowhere close to what we obtained for the Tug-of-War algorithm. Indeed, it will be "barely sublinear". For this presentation, we will focus only on insert only streams, and in fact, only on updates of the form $(i, 1)$.

- *"Intuition"*. Recall the main problem in *exactly* computing F_k: we can't know all the $\mathbf{f}_i$'s. But, we can definitely know one of the $\mathbf{f}_i$'s exactly at the end: simply keep track of that particular element and forget the rest. So an idea emerges: sample $i \in [n]$ uniformly at random, evaluate $\mathbf{f}_i$ exactly, and return $Z = n \cdot \mathbf{f}_i^k$. One can easily check Z is unbiased. The problem is the variance.

$$\mathbf{Exp}[Z^2] = \frac{1}{n} \sum_{i=1}^{n} n^2 \mathbf{f}_i^{2k} = n \cdot F_{2k} \quad \Rightarrow \quad \frac{\mathbf{Var}[Z]}{\mathbf{Exp}^2[Z]} = n \cdot \frac{F_{2k}}{(F_k)^2} - 1$$

  In the extreme case when one element dominates the stream, we get $F_{2k} \approx (F_k)^2$, which means that one would need n-repetitions. Not good.

  However, what if we could do "importance sampling": suppose we could somehow sample i proportional to $\mathbf{f}_i$, and then do exactly the same thing? More precisely, suppose we sampled $i \in [n]$ with probability $\frac{\mathbf{f}_i}{F_1}$, evaluate $\mathbf{f}_i$ exactly, and then return $Z = F_1 \cdot \mathbf{f}_i^{k-1}$. Then note

$$\mathbf{Exp}[Z] = \sum_{i=1}^{n} \frac{\mathbf{f}_i}{F_1} \cdot F_1 \mathbf{f}_i^{k-1} = F_k \quad \text{and} \quad \mathbf{Exp}[Z^2] = \sum_{i=1}^{n} \frac{\mathbf{f}_i}{F_1} \cdot F_1^2 \mathbf{f}_i^{2k-2} = F_1 \cdot F_{2k-1}$$

  How good is this? The next "fact" shows that it's not too bad. The proof of this fact is some analytical manipulation which is not really important for the main ideas in this lecture. We defer the proof to the very end.

  Fact 1. For any non-negative numbers $(\mathbf{f}_1, \ldots, \mathbf{f}_n)$, we have $F_1 \cdot F_{2k-1} \leq n^{1-\frac{1}{k}} (F_k)^2$.

  We will provide this proof later. But note that this implies that $o(n)$ samples suffices, and indeed this is what we will achieve.

- Of course, we don't a priori know how to sample $i \in [n]$ with probability $\frac{\mathbf{f}_i}{F_1}$. The first thing we will see today is a procedure of sampling a *random* element from a stream. More precisely, given any stream of elements (with repetitions), we show how to end up with a random element R from the stream. That is, for any $i \in [n]$, $\mathbf{Pr}[R = i] = \frac{\mathbf{f}_i}{F_1}$. This algorithm, which is by itself something worth keeping in ones arsenal, is called ***reservoir sampling.***

  However, we get this random element at the *end* of the stream and not at the *beginning* when we really wanted it. This is where the second cleverness in this AMS algorithm comes in. As we will see,

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 16th April, 2021*
These have not gone through scrutiny and may contain errors. If you find any, or have any other comments, please email me at deeparnab@dartmouth.edu. Highly appreciated!

the reservoir sampling *always* maintains an R which is uniformly at random among the stream seen so far. What the algorithm does is once R is modified, it wipes out all memory and keeps counting the number of occurrences of this element R *henceforth*. If this count is C, then the estimate of the algorithm is $Z = F_1(C^k - (C-1)^k)$. Since the choice of this random element $i = R$ is *equally* likely among all the $\mathbf{f}_i$ possibilities, we will see that the expected value of Z precisely evaluates to $F_1\mathbf{f}_i^{k-1}$. Which is exactly what we wanted above. Details follow.

- **Reservoir Sampling.** How do we sample a random element from a stream? If we knew the number of items $M = F_1$ up-front, then it's easy : we select $j \in \{1, 2, \ldots, M\}$ at random and then just wait for the jth item to come. But we don't know M. In particular, we want to have the following data structure: for every $1 \leq t \leq M$, we want an R_t where R_t is a uniformly-at-random element among the t elements seen so far. If you have never seen this before, then it is worth thinking a bit about it as a nice puzzle.

  Here is one way to do it. The first item has to be R_1. When the second item arrives, we set R_2 to be this with probability $1/2$, and let it remain R_1 with probability $1/2$. Note that R_2 is one of the two items equally likely. But this idea generalizes. When the tth item arrives, we set R_t to be this item with probability $\frac{1}{t}$, and let $R_t = R_{t-1}$ with probability $\frac{t-1}{t}$. What is the probability $R_t = j$ for some element j among the first j items. If j is the tth item, then it is clearly $\frac{1}{t}$ by design. If not, then it is $\frac{t-1}{t} \cdot \mathbf{Pr}[R_{t-1} = j]$. But inductively, $\mathbf{Pr}[R_{t-1} = j] = \frac{1}{t-1}$. Thus, $\mathbf{Pr}[R_t = j] = \frac{1}{t}$ for all j in the first t items.

```
1: procedure RESERVOIR SAMPLING:
2:     ▷ For every t, store R_t which is supposed to uniform among the first t elements
3:     for update e_t do:
4:         if t = 1 then:
5:             Set R_t ← e_1 with probability 1.
6:         else:
7:             Set R_t ← e_t with probability 1/t, and R_t ← R_{t-1} with probability (t-1)/t.
```

- **The AMS Estimate.** The algorithm maintains maintains a counter C which counts the occurrence of an element e in the stream. Initially, $C = 0$ and $e \leftarrow \perp$. It also maintains the reservoir sample R_t at every update t as describe above. If $R_t = R_{t-1}$ (which occurs with probability $1 - \frac{1}{t}$), then it checks if the tth element is e. If so, it increments C by 1, and otherwise ignores e. If $R_t \neq R_{t-1}$, that is, R_t is set to the tth element, then e is reset to be this element *and* the count C is reset to 1. Note that this resetting of the counter is done *even* if the tth element is the same as e. The algorithm also maintains a count of F_1 (the total number of elements in the stream). At the end, it returns $Z \leftarrow F_1 \cdot \left(C^k - (C-1)^k\right)$.

```
procedure AMS-Fk:
    Initialize M ← 0, C ← 0 and e ← ⊥.
    for update e_t do:
        Increment M ← M + 1.
        Maintain R_t using Reservoir Sampling.
        if R_t is kept the same as R_{t-1} then:
            if e_t = e then:
                Increment C ← C + 1
        else: ▷ R_t is now reset to e_t
            Set e ← e_t and C ← 1.
    return Z ← M · (C^k − C^{k−1}).
```

Theorem 1. The estimate Z returned by AMS-Fk is an unbiased estimate of F_k. Furthermore, $\frac{\mathbf{Var}[Z]}{\mathbf{Exp}^2[Z]} \leq k \cdot n^{1-\frac{1}{k}}$.

– Note that $M = F_1$, the sum of frequencies of all elements. Consider the element R_M, and observe a couple of things. First, for any $i \in [n]$, the probability $\mathbf{Pr}[R_M = i] = \frac{\mathbf{f}_i}{F_1}$ since the reservoir sample returns a random element in the stream, and there are $\mathbf{f}_i$ occurrences of i. Now, let t be the index in the stream such that $R_M = t$; that is, after position t, the random variable was not modified. The second observation is that conditioned on $R_M = i$, the value of t is *uniformly distributed* among the $\mathbf{f}_i$ choices of i. Again, this is because t is uniformly distributed in M, and thus uniformly distributed among any subset as well, if conditioned to be there.

What this means is conditioned on $R_M = i$, the value of C at the end is equally likely to be $\{1, 2, \ldots, \mathbf{f}_i\}$. And therefore,

$$\begin{aligned}
\mathbf{Exp}[Z] &= M \cdot \sum_{i=1}^{n} \mathbf{Pr}[R_M = i] \cdot \sum_{t=1}^{\mathbf{f}_i} \mathbf{Pr}[C = t] \cdot \left(t^k - (t-1)^k\right) \\
&= F_1 \cdot \sum_{i=1}^{n} \frac{\mathbf{f}_i}{F_1} \cdot \sum_{t=1}^{\mathbf{f}_i} \frac{1}{\mathbf{f}_i} \cdot \left(t^k - (t-1)^k\right) \\
&= \sum_{i=1}^{n} \underbrace{\sum_{t=1}^{\mathbf{f}_i} \left(t^k - (t-1)^k\right)}_{\text{telescopes to } \mathbf{f}_i^k} = F_k
\end{aligned}$$

– How about the variance? Well, we see

$$\begin{aligned}
\mathbf{Exp}[Z^2] &= M^2 \cdot \sum_{i=1}^{n} \mathbf{Pr}[R_M = i] \cdot \sum_{t=1}^{\mathbf{f}_i} \mathbf{Pr}[C = t] \cdot \left(t^k - (t-1)^k\right)^2 \\
&= F_1 \sum_{i=1}^{n} \sum_{t=1}^{\mathbf{f}_i} \left(t^k - (t-1)^k\right)^2 \underbrace{\leq}_{\text{Fact 2}} k F_1 F_{2k-1} \underbrace{\leq}_{\text{Fact 1}} k n^{1-\frac{1}{k}}
\end{aligned}$$

where we use another analytic massaging encapsulated below

Fact 2. $\sum_{i=1}^{n}\sum_{t=1}^{\mathbf{f}_i}\left(t^k-(t-1)^k\right)^2 \le kF_{2k-1}$

Proof. We will use the following fact: $t^k-(t-1)^k \le kt^{k-1}$ for any $t \ge 1$. This can be seen using elementary calculus (the mean value theorem). Therefore,

$$\begin{aligned}\sum_{i=1}^{n}\sum_{t=1}^{\mathbf{f}_i}\left(t^k-(t-1)^k\right)^2 &\le \sum_{i=1}^{n}\sum_{t=1}^{\mathbf{f}_i} kt^{k-1}\cdot\left(t^k-(t-1)^k\right)\\ &\le k\sum_{i=1}^{n}\mathbf{f}_i^{k-1}\underbrace{\sum_{t=1}^{\mathbf{f}_i}\left(t^k-(t-1)^k\right)}_{\text{telescopes to } \mathbf{f}_i^k}\\ &= k\sum_{i=1}^{n}\mathbf{f}_i^{k-1}\cdot\mathbf{f}_i^k = kF_{2k-1} \quad \square\end{aligned}$$

– We end with a proof of Fact 1, that is, we wish to prove $F_1F_{2k-1} \le n^{1-\frac{1}{k}}\cdot(F_k)^2$.
We first note that if we denote $L := \max_i \mathbf{f}_i$, then $F_{2k-1} \le L^{k-1}F_k$. Thus, it suffices to prove that $F_1\cdot L^{k-1} \le n^{\frac{k-1}{k}}F_k$.
To see this, first note that $L^{k-1} \le (F_k)^{\frac{k-1}{k}}$; the RHS is a sum of the kth powers of a bunch of things taken to the $(k-1)/k$th root, while the RHS is the $(k-1)/k$th root of only a single elements kth power. Thus, we just need to prove that $F_1 \le n^{\frac{k-1}{k}}\cdot F_k^{\frac{1}{k}}$.
The last follows from the *convexity* of the function $x \mapsto x^k$. Jensen's inequality implies

$$\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_i\right)^k \le \frac{1}{n}\cdot\sum_{i=1}^{n}\mathbf{f}_i^k \quad\Rightarrow\quad F_1^k \le n^{k-1}\cdot F_k$$

which implies what we want by taking the kth root.