Weekplan: Streaming II.

Philip Bille Inge Li Gørtz Eva Rotenberg

References and Reading

[1] Amit Chakrabarti: *Data Stream Algorithms* 2011 (updated July 2020) chapter 2.

[2] Kurt Mehlhorn and He Sun: *Streaming Algorithms* 2014.

[3] Jelani Nelson: *Algorithms for Big Data, lecture 3* 2015 section 2.1

[4] P. Flajolet: Approximate Counting: A Detailed Analysis

We recommend reading the specified chapters and sections of [1] and section 3.2 and 3.3.0-3.3.1 of [2] .in detail.

Hash function cheat-sheet

The notation $[x]$ Throughout this sheet I will use the notation $[x]$ to denote the set $\{0, 1, 2, \ldots, x-1\}$.

Definition: Hash function A hash function $h : U \to [m]$ is a random variable in the class of all functions $U \to [m]$.

Definition: 2-independent Also known as *strongly universal* or *pairwise independent*.
A hash function $h : U \to [m]$ is 2-independent if for all $x \neq y \in U$ and $q, r \in [m]$: $P[h(x) = q \wedge h(y) = r] = \frac{1}{m^2}$.
Equivalently, the following two conditions hold:

- for any $x \in U$, $h(x)$ is uniform in $[m]$,
- for any $x \neq y \in U$, $h(x)$ and $h(y)$ are independent.

Exercises

1 Sanity check Hash functions sometimes have collisions. Here, we choose our family of hash functions carefully to avoid collisions. Would collisions lead to overestimating or underestimating the number of distinct elements?

2 Hash functions Suppose h is a 2-independent hash function from $[n]$ to $[n^3]$. Show that h is injective with probability at least $1 - \frac{1}{n}$.

3 Analyse performance of the algorithm. The purpose of this exercise is to walk you through the proof in Section 2.3 of [1].

3.1 Describe the indicator variables $X_{r,j}$ and Y_r in your own words.

3.2 Calculate the expected value of $X_{r,j}$ and of Y_r. (How) Does the expected value of $X_{r,j}$ depend on j?

3.3 Where did we use that the hash function is pairwise independent.

3.4 Bound the variance of Y_r.

3.5 Bound the probability of Y_r being > 0.

3.6 Bound the probability of Y_r being $= 0$.

3.7 Let z' be the value of z when the algorithm ends (as on the slides). Explain why $P[z' \geq a] = P[Y_a > 0]$.

3.8 Show that $P[\hat{d} \geq 3d] \leq \frac{\sqrt{2}}{3}$.

3.9 Explain why $P[z' \leq b] = P[Y_{b+1} = 0]$.

3.10 Show that $P[\hat{d} \leq d/3] \leq \frac{\sqrt{2}}{3}$.

4 Counting rare elements[1] Paul goes fishing. There are u different fish species $U = \{1, \ldots, u\}$. Paul catches one fish at a time. Let a_t be the fish species he catches at time t. Let $ct[j] = |\{a_i | a_i = j, i \leq t\}|$ be the number of times he catches a fish of species j up to time t. Species j is *rare* at time t if it appears precisely once in his catch up to time t. The rarity $\rho[t]$ of his catch at time t is defined as:

$$\rho(t) = \frac{\#\text{rare species}}{u}.$$

4.1 Explain how Paul can calculate $\rho(t)$ precisely, using $2u + \log m$ bits of space.

4.2 However, Paul wants to store only as many bits as will fit his tiny suitcase, i.e., $o(u)$, preferably $O(1)$ bits. Therefore, Paul picks k random fish species each independently, randomly with probability $1/u$ at the beginning and maintains the number of times each of these fish species appear in his bounty, as he catches fish one after another. Paul outputs the estimate

$$\hat{\rho}(t) = \frac{\#\text{rare species in the sample}}{k}.$$

Let $c_1(t), \ldots, c_k(t)$ be the value of the counters at time t. Show that $P[\hat{\rho}(t) \geq 3\rho] \leq 1/3$.

Hint: Calculate first $P[c_i(t) = 1]$.

Graph Streaming In graph streaming, the stream consists of a sequence of edges $(u_1, v_1), (u_2, v_2), \ldots$. In general, these edges can be additions (appearing) or subtractions (disappearing), however, in the following exercises, we will focus on the case where edges are only appearing. We will also assume that the graph is *undirected* and *unweighted*. The goal is to use $O(n \log^{O(1)} n)$ space.

4 Graph Streaming 0: Warmup $[w]$ How much space in terms of n do you need in the worst case do describe a graph with n nodes?

5 Graph Streaming I: Connectivity Design a streaming algorithm that counts the number of connected components. Analyse the space and the update and query time of your algorithm.

[1] This exercise is from Muthukrishnan "Data Streams: Algorithms and Applications