Group Assignment 3.2

Niv Adam, David Kaufmann, Casper Kristiansson, Nicole Wijkman

December 14, 2023

1 Algorithm

The algorithm to fulfil the requirements performs the Misra-Gries algorithm for heavy hitters once. It then also returns a list of k elements x_i but different to Misra-Gries it also returns the counters corresponding to x_i. So the output of the algorithm is a list of k tuples: $T = (x_1, c_1), ..., (x_k, c_k)$. To get an estimate of the frequency of any element y this data structure can be used as follows

$$\hat{f}_y = \begin{cases} c_i & \text{if } \exists (x_i, c_i) \in T : x_i = y \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

To show that this algorithm fulfils the requirement we need to prove that the following equation holds in both cases

$$f_y - \frac{n}{k+1} \leq \hat{f}_y \leq f_y \tag{2}$$

2 Correctness

Let us start with elements x such that $f_x > n/(k+1)$. Lemma 2 in lecture 11 states that every such element must appear in the resulting list. From the lecture, it is also evident that the counter values will never be higher than the actual number of elements in the stream, since the counter is only increased if an element with that value is observed. This means for all $(x, c) \in T : c \leq f_x$. Therefore we are merely interested in a lower bound on the counter. Following the trash bags argument from the lecture we get that if x is part of l trash bags $c = f_x - l$, since each trash back contains only distinct elements. Each trash back contains $k + 1$ elements, which means there can be $n/(k + 1)$ trash bags at most. Thereafter we get $l \leq n/(k+1)$. That means the counter can not be lower than $f_x - n/(k + 1)$. Concluding this gives the desired property that

$$f_x - \frac{n}{k+1} \leq c_i \leq f_x \text{ if } x_i = x \tag{3}$$

This first argument dealt with all elements x with $f_x > n/(k + 1)$. For all other elements y we don't know whether they appear in T or not. But since we know that $f_y \leq n/k + 1$ we get $f_y - n/(k + 1) \leq 0$. Therefore, the lower bound is always satisfied when setting the estimate to 0. Furthermore, if such an element y appears in T it is still true that the counter is lower than the true frequency. Therefore, the upper bound is also always satisfied.

In conclusion, this proof shows that independent of the true frequency of the elements, the given algorithm always returns an estimate within the required guarantees.