Supplementary Appendix for *Delegation Space-Saving*: Fast, Accurate, and Concurrent Frequent Elements Queries and Updates

Anonymous Author(s)

ACM Reference Format:
Anonymous Author(s). 2020. Supplementary Appendix for *Delegation Space-Saving*: Fast, Accurate, and Concurrent Frequent Elements Queries and Updates. In *51st International Conference on Parallel Processing (ICPP 2022) August 29 - Sept 1, 2022 Bordeaux, France.* ACM, New York, NY, USA, 3 pages. https://doi.org/tobedefined

1 APPENDIX

Due to space limitations, below we include some complementary material that could not fit in the 10-page limit:

Proof of lemma 1

Proof. Let ss be a Space-Saving algorithm instance that maintains $k = \frac{1}{\epsilon}$ counters and that processes the stream $\mathcal{S}$, and let ss_i be one of T Space-Saving algorithm instances that each processes the sub-stream S_i. The established guarantee of ss [2] is to report $\mathcal{F}$ by tracking all elements $e \in U$ such that

$$f(e) > N\epsilon = \frac{N}{k} \tag{1}$$

We have that the length of stream $\mathcal{S}_i$ is equal to $N_i = \frac{N}{T}$ due to domain splitting. An algorithm instance ss_i that maintains k counters will then report $\mathcal{F}_i$, the ϵ-approximate frequent elements that are part of U_i, by tracking all elements $e \in U_i$ such that

$$f(e) > \frac{N}{Tk} \tag{2}$$

If we reduce the number of counters maintained by ss_i to $\frac{k}{T}$, then ss_i is guaranteed to track all elements e such that

$$f(e) > \frac{N}{T\frac{k}{T}} = \frac{N}{k} = N\epsilon \tag{3}$$

Since (1) and (3) are the same, we can conclude that each of the T different ss_i instances can keep just $\frac{k}{T} = \frac{1}{\epsilon T}$ counters to have the same accuracy guarantee as ss, given that the union of each $\mathcal{F}_i$ is output by Delegation Space-Saving. □

Proof of lemma 2

Proof. Here we slightly modify the proof of Theorem 5 in [2] (inequality 8 and onward).

Let $k = \frac{1}{\epsilon}$ denote the number of counters and let $H_{n,a} = \sum_{i=k+1}^{n} \frac{1}{i^a}$.

ICPP 2022, August 29 – Sept 1, 2022, Bordeaux, France

ACM ISBN to be defined…$15.00
https://doi.org/tobedefined

Lemma 3.4 and Equation (1) from [2] state that the maximum least value of a counter in Space-Saving is $min_{max} \geq \frac{N-F_k}{k}$, where F_k is the sum of occurrences of the k most frequently occurring elements in $\mathcal{S}$. Any element e with $f(e) > min_{max}$ is guaranteed to be monitored by Space-Saving, regardless of stream permutation (lemma 4.3 [2]). We have that $N - F_k$ is equal to the sum of occurrences of elements ranked k+1 through $|U|$. An element e has occurred $f(e) = \frac{N}{H_{|U|,a} r(e)^a}$ times in a stream of length N constructed by drawing elements from the Zipf distribution with parameter a. We can encode this reasoning in inequality 4

$$f(e) > min_{max} \overset{zipf}{\equiv} \frac{1}{r(e)^a}\frac{N}{H_{|U|,a}} > \frac{1}{k}\frac{N}{H_{|U|,a}}\sum_{i=k+1}^{|U|}\frac{1}{i^a} \tag{4}$$

Simplifying yields:

$$\frac{1}{r(e)^a} > \frac{1}{k}\sum_{i=k+1}^{|U|}\frac{1}{i^a} \tag{5}$$

Metwally et al. then approximates the expression in the right-hand side of inequality (5):

$$\frac{1}{r(e)^a} > \frac{1}{k^a}\sum_{i=2}^{\frac{|U|}{k}}\frac{1}{i^a} \tag{6}$$

$\sum_{i=2}^{\frac{|U|}{k}} \frac{1}{i^a} < \zeta(a) - 1$, so we can let k satisfy a stronger constraint:

$$\frac{1}{r(e)^a} > \frac{1}{k^a}(\zeta(a) - 1) \tag{7}$$

We continue by considering an assumption that the sum of all occurrences of the non-frequent elements represented by $(\zeta(a) - 1)$ are evenly distributed to all threads due to domain splitting. (This assumption is motivated by the fact that $N - F_k$ tends to be much smaller than F_k, and at the same time, $|U| - (k+1) >> k$, in the case that the input follows a Zipf distribution with $a > 1$.)

$$\frac{1}{r(e)^a} > \frac{1}{k^a}\frac{(\zeta(a) - 1)}{T} \tag{8}$$

This simplifies to:

$$k > r(e)\left(\frac{\zeta(a) - 1}{T}\right)^{\frac{1}{a}} \tag{9}$$

We now direct our attention to the inequality $N\epsilon > min_{max} > \frac{N}{r(e)^a \zeta(a)}$, which describes the least ranked element that exceeds ϵN. It can be solved for $r(e)$:

$$r(e) < \left(\frac{1}{\epsilon\zeta(a)}\right)^{\frac{1}{a}} \tag{10}$$

We can now substitute r in equation 9:

$$k > \left(\frac{\zeta(a)-1}{T\epsilon\zeta(a)}\right)^{\frac{1}{a}} = \left(\frac{1}{T\epsilon} - \frac{1}{T\epsilon\zeta(a)}\right)^{\frac{1}{a}} \tag{11}$$

Thus, setting $k = \left(\frac{1}{T\epsilon}\right)^{\frac{1}{a}}$ guarantees that inequality (4) is satisfied, i.e. all elements with $f(e) > \epsilon N$ are *monitored* (and hence might be delivered) by the Space-Saving instance. Therefore, using $k = \left(\frac{1}{T\epsilon}\right)^{\frac{1}{a}}$ counters at each Space-Saving instance in Delegation Space-Saving algorithm solves the ϵ-approximate frequent elements problem, given that all algorithm instances are considered when reporting the frequent elements. □

Proof of lemma 5

Proof. If we consider an unbounded universe of possible elements to target the general case, then the probability of drawing element e is $\frac{1}{\zeta(a)r(e)^a}$, where $r(e)$ denotes the rank of element e. Since there are at most mT counts of an element in Delegation Filters kept for the owner of e by lemma 4, then since these mT elements were drawn from a noiseless Zipf distribution, there can only be at most $\frac{mT}{\zeta(a)r(e)^a}$ counts of element e. □

Proof of claim 1

Proof. We have that $f_{N_S}(e) \leq f_{N_E}(e)$. There are at most $\mathbb{D}(e)$ counts of element e that have not yet been inserted into the Space-Saving instance of the owner of e, therefore $f_{N_S}(e) - \mathbb{D}(e) \leq \hat{f_N}(e)$ is the minimum value a counter can assume. The maximum overestimation of Space-Saving is ϵN, which is maximized at the end of a query, when N_E elements have been processed. Therefore the estimated count of an element is at most $\hat{f_N}(e) \leq f_{N_E}(e) + \epsilon N_E$. □

Proof of claim 2

Proof. The Delegation Space-Saving algorithm reports all elements with estimated count

$$f\hat{(e)}_{N_S} > \phi N_S \tag{12}$$

This is true since N_S is calculated at the start of a query, all elements with $\hat{f_N}(e) > \phi N_S$ are reported, and Space-Saving counters increase monotonically.

As shown in claim 2, $f\hat{(e)}$ is at least (a) $f(e)_{N_S} - \mathbb{D}(e)$ and at most (b) $f(e)_{N_E} + N_E\epsilon$.

Substituting $f\hat{(e)}$ for (a) in (12) yields:

$$f(e)_{N_S} > \phi N_S + \mathbb{D}(e) \tag{13}$$

Substituting $f\hat{(e)}$ for (b) in (12) yields:

$$f(e)_{N_E} > \phi N_S - \epsilon N_E \tag{14}$$

Expression (a) and the negation of (b) together give that all elements $f(e)_{N_S} > \phi N_S$ and no elements $f(e)_{N_E} < \phi N_S - \epsilon N_E$ are reported by the Delegation Space-Saving algorithm. □

Proof of claim 3

Proof. We normalize the expression from inequality (13):

$$\frac{f(e)}{N_S} > \phi + \frac{\mathbb{D}(e)}{N_S} \tag{15}$$

$\mathbb{P}(e) = \frac{f_{N_S}(e)}{N_S}$ is the probability at which element e occurs in the underlying input distribution. We then have that:

$$\lim_{N_S \to \infty} \mathbb{P}(e) > \phi + \frac{\mathbb{D}(e)}{N_S} \to \mathbb{P}(e) > \phi \tag{16}$$

Meaning that all elements with $\mathbb{P}(e) > \phi$ are reported given that enough elements have been processed. □

Proof of claim 4

Proof. Using the expression from inequality (13), we can rewrite it in its Zipfian form:

$$\frac{1}{\zeta(a)r(e)^a} > \phi + \frac{mT}{\zeta(a)r(e)^a N_S} \tag{17}$$

Solving for r(e) gives:

$$\left(\frac{1}{\zeta(a)\phi} - \frac{mT}{\zeta(a)\phi N_S}\right)^{\frac{1}{a}} > r(e) \tag{18}$$

Simplifying yields:

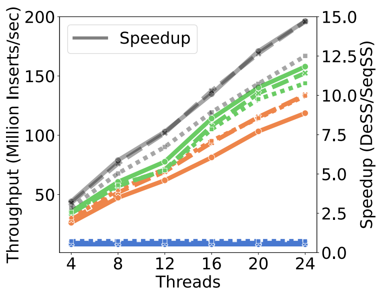
$$\left(\frac{1}{\zeta(a)\phi}\right)^{\frac{1}{a}} \left(1 - \frac{mT}{N_S}\right)^{\frac{1}{a}} > r(e) \tag{19}$$

We then have that:

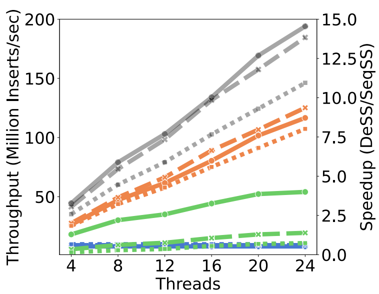
$$\lim_{N_S \to \infty} \left(\frac{1}{\zeta(a)\phi}\right)^{\frac{1}{a}} \left(1 - \frac{mT}{N_S}\right)^{\frac{1}{a}} > r(e) \to \left(\frac{1}{\zeta(a)\phi}\right)^{\frac{1}{a}} > r(e) \tag{20}$$

Meaning that all elements with a rank lower than $\left(\frac{1}{\zeta(a)\phi}\right)^{\frac{1}{a}}$ are guaranteed to be reported given that enough elements have been processed. □

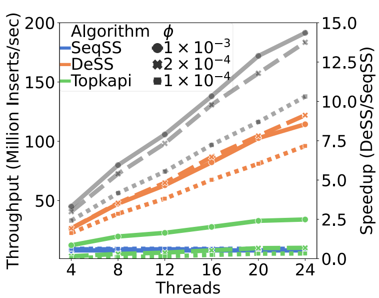
Scalability results, CAIDA direction B



(a) 0% queries.



(b) 0.01% queries.



(c) 0.02% queries.

Figure 1: Dir B scalability

Reproducibility: The code is availiable for reproducibility purposes: [1].

REFERENCES

[1] Anonymous. Delegation Space-Saving. https://github.com/anonymous7495/Delegation-Space-Saving/tree/main, 4 2022.

[2] A. Metwally, D. Agrawal, and A. E. Abbadi. An integrated efficient solution for computing frequent and top- k elements in data streams. *ACM Transactions on Database Systems*, 31(3):1095–1133, 2006.