Optimal Approximations of the Frequency Moments of Data Streams

Piotr Indyk *
MIT
indyk@mit.edu

David Woodruff †
MIT
dpwood@mit.edu

ABSTRACT

We give a 1-pass $\tilde{O}(m^{1-2/k})$-space algorithm for computing the k-th frequency moment of a data stream for any real $k > 2$. Together with the lower bounds of [1, 2, 4], this resolves the main problem left open by Alon et al in 1996 [1]. Our algorithm also works for streams with deletions and thus gives an $\tilde{O}(m^{1-2/p})$ space algorithm for the L_p difference problem for any $p > 2$. This essentially matches the known $\Omega(m^{1-2/p-o(1)})$ lower bound of [13, 2]. Finally the update time of our algorithm is $\tilde{O}(1)$.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: ANALYSIS OF ALGORITHMS AND PROBLEM COMPLEXITY

General Terms

Algorithms

Keywords

Streaming algorithms, frequency moments

1. INTRODUCTION

Computing over data streams is a recent phenomenon that is of growing interest in many areas of computer science, including databases, computer networks and theory of algorithms. In this scenario, it is assumed that the algorithm sees a stream of elements one-by-one in arbitrary order, and needs to compute certain statistics over the input. However, it does not have enough memory to store the whole stream. Therefore, it must maintain only a sketch of the data, which is small but nevertheless powerful enough to compute the desired statistics.

*Supported in part by NSF ITR grant CCR-0220280, David and Lucille Packard Fellowship and Alfred P. Sloan Fellowship.
†Supported by an NDSEG fellowship.

STOC'05, May 22-24, 2005, Baltimore, Maryland, USA.

Within the theory community, the recent interest in these problems can be traced to the paper of Alon, Matias and Szegedy [1]. In that influential paper, the authors give upper and lower bounds for the space complexity of streaming algorithms for the problem of estimating the *frequency moments* of a stream. In this problem, the stream is a sequence $\mathcal{S}$ of elements $a_1, \ldots, a_n$, $a_j \in [m] = \{1 \ldots m\}$. For each element $i \in [m]$, its frequency f_i in $\mathcal{S}$ is the number of times it occurs in $\mathcal{S}$. The k-th frequency moment (for real $k \geq 0$) of $\mathcal{S}$ is defined as

$$F_k = F_k(\mathcal{S}) = \sum_{i \in [m]} f_i^k$$

An algorithm A (ϵ, δ)-approximates F_k if A outputs a number $\tilde{F_k}$ for which $\Pr[|\tilde{F_k} - F_k| > \epsilon F_k] < \delta$. In [1], the authors present $\tilde{O}(1)$[1] space algorithms which (ϵ, δ)-approximate F_k for integers $k \leq 2$, and $\tilde{O}(m^{1-1/k})$ space algorithms for integers $k \geq 3$. For $k > 5$, they also show an $\Omega(m^{1-5/k})$ lower bound. Their ideas spurred vigorous research on problems in this area. Soon, their algorithmic results (for the case $k \leq 2$) were shown [7, 10] to hold for a more general problem of estimating the p-th norm of the vector $(f_1, \ldots, f_m)$; see the definition below. Later, a line of research [2, 4] culminated in an $\tilde{\Omega}(m^{1-2/k})$ space lower bound for estimating F_k for any real $k > 2$. Recently, better upper bounds of $\tilde{O}(m^{1-1/(k-1)})$ and $\tilde{O}(m^{1-2/(k+1)})$ were shown in [6, 8] and [9].

In this paper, we present a one-pass $\tilde{O}(m^{1-2/k})$-space streaming algorithm for estimating F_k for any real $k > 2$. By the aforementioned results, our space bound is tight (up to $\tilde{O}(1)$ factors). Thus, together with the lower bound of [2, 4], our result resolves the main problem left open by [1], resolving a long line of research on the frequency moments. We note that, in addition to matching the known space lower bound, the update time of our algorithm is essentially optimal: $\tilde{O}(1)$ per stream element.

Our algorithm has the additional feature of being able to estimate F_k over streams with *deletions*, instead of just the update-only model defined above. It can therefore be used to estimate L_p differences, $p > 2$, in one-pass with space $\tilde{O}(m^{1-2/p})$. The L_p difference problem is to approximate $\sum_{i=1}^m |a_i - b_i|^p$, where the a_i's and b_i's arrive in arbitrary order. Prior to our work the best known algorithms used space $\tilde{O}(m^{1-1/(p-1)})$ [6, 8]. Since there is an $\Omega(m^{1-2/p-o(1)})$

[1] $f(m) = \tilde{O}(g(m))$ means $f(m) = g(m) \left(\frac{1}{\epsilon}\right)^{O(1)} \log^{O(1)} mn \log 1/\delta$; the $\tilde{\Omega}$ notation is defined similarly.

space lower bound [13, 2] for this problem, our algorithm is essentially optimal.

Our techniques: The earlier algorithms for estimating F_k were obtained by constructing a single estimator, which was shown to equal F_k in expectation, and to have small variance. Our algorithm departs from this approach. Instead, the main idea of our algorithm is as follows. First, we (conceptually) divide the elements into classes S_i, such that the elements in class S_i have frequency $\approx (1+\epsilon)^i$. We observe that, in order for the elements in S_i to contribute significantly to the value of F_k, it must be the case that the size s_i of S_i is comparable to the size of $S_{i+1} \cup \ldots \cup S_{\log_{1+\epsilon} n}$. If this is the case, we have a good chance of finding an element from S_i if we restrict the stream to an $\approx 1/s_i$ fraction of universe elements, and then find the most frequent elements in the substream. The contribution of S_i to F_k can then be approximately estimated by $s_i \cdot (1+\epsilon)^{ik}$. By summing up all estimated contributions, we obtain an estimator for F_k.

Unfortunately, finding the most frequent element in a general stream (even approximately) requires storage that is linear in the stream size. However, if the distribution of the stream elements is not very "heavy-tailed", then a more efficient algorithm is known [5]. This more efficient method is based on the sketching algorithm for F_2 given in [1]. We show that the streams generated by our algorithm (for S_i's that contribute to F_k), satisfy this "tail property", and thus we can use the algorithm of [5] in our algorithm.

A somewhat technical issue that arises in implementing the above algorithm is a need to classify a retrieved element i into one of the classes. For this, we need to know f_i. This information is easy to obtain using a second pass. In the absence of that, we use the estimation of f_i provided by the algorithm of [5], but we define the thresholds defining the classes randomly, to ensure that the error in estimating the frequencies is unlikely to result in misclassification of a frequency.

2. NOTATION AND PRELIMINARIES

We are given a stream $\mathcal{S}$ of n elements, each drawn from the universe $[m]$. Our goal is to output an approximation to the kth frequency moment F_k. For simplicity, we assume $k \geq 2$ is a constant (for $k < 2$, space-optimal algorithms already exist [1, 3]) , while $m, n, 1/\epsilon$ may be growing. Let $0 < \delta, \epsilon < 1$ be the desired confidence and accuracy of our estimate, respectively. We define the following parameters:

$$\epsilon' = O(\epsilon), \quad \alpha = 1+\epsilon', \quad \lambda = \epsilon'/\alpha^k, \quad L = \frac{\lambda}{\log n + 1}$$

W.l.o.g., we may assume that m is a power of two and that n is a power of α. Unless otherwise specified, logs are to the base α. We define the *frequency classes* S_i, for $0 \leq i \leq \log n$, as

$$S_i = \{j \mid \alpha^i \leq f_j < \alpha^{i+1}\}.$$

We use the shorthand notation s_i for $|S_i|$. We say that a class S_i *contributes* if

$$\alpha^{ik} s_i \geq L F_k.$$

The next lemma relates s_i to $|\cup_{l>i} S_l| = \sum_{l>i} s_l$ when S_i contributes.

LEMMA 1. *If S_i contributes, then $s_i \geq L \sum_{l>i} s_l$.*

PROOF. Notice that

$$\alpha^{ik} s_i \geq L F_k \geq L \sum_{l>i} s_l \alpha^{lk} \geq L \sum_{l>i} s_l \alpha^{ik},$$

and the lemma follows by dividing by α^{ik}. □

We define F_k^C to be the component of F_k due to the contributing frequency classes, namely,

$$F_k^C = \sum_{\text{contributing } S_i} \sum_{j \in S_i} f_j^k.$$

We define F_k^{NC} to be the component due to the non-contributing classes, so $F_k^C + F_k^{NC} = F_k$. The next lemma shows that F_k^{NC} is small.

LEMMA 2.

$$F_k^{NC} < \lambda \alpha^k F_k.$$

PROOF. We note that if $j \in S_i$, then $f_j < \alpha^{i+1}$. Therefore,

$$F_k^{NC} < \sum_{\text{non-contr. } S_i} s_i \alpha^{(i+1)k} < L\alpha^k \sum_{\text{non-contr. } S_i} F_k \leq \lambda \alpha^k F_k.$$

□

3. THE ALGORITHM

3.1 An Idealized Algorithm

We start by making the unrealistic assumption that we have the following oracle algorithm. Later we remove this assumption by approximating the oracle with the Countsketch algorithm of [5].

ASSUMPTION 3. *For some $B = B(m,n)$, there exists a 1-pass B-space algorithm* Max *that outputs the maximum frequency of an item in its input stream.*

We start by describing our algorithm which outputs a $(1\pm\epsilon)$-approximation to F_k with probability at least 8/9. The main idea is to estimate F_k by estimating each of the set sizes s_i and computing $\sum_i s_i \alpha^{ik}$. Although in general it will not be possible to estimate all of the s_i, we show that we can estimate the sizes of those S_i that contribute. By Lemma 2, this will be enough to estimate F_k. The space complexity will be B up to poly $\left(\frac{1}{\epsilon} \ln n \ln m\right)$ terms.

The algorithm approximates s_i by restricting the input stream to randomly-chosen *substreams*. By this, we mean that it randomly samples subsets of items from $[m]$, and only considers those elements of $\mathcal{S}$ that lie in these subsets. More precisely, the algorithm creates $b = O(\ln(m/L))$ families of $R = O\left(\frac{1}{L^3} \ln(\ln m \log n)\right)$ substreams $\mathcal{S}_j^r$, for $j \in [b]$ and $r \in [R]$. For each r, $\mathcal{S}_j^r$ will contain about $m/2^j$ randomly chosen items. If a class contributes, then using Lemma 1 we can show there will be some j for which a good fraction of the maximum frequencies of the family $\{\mathcal{S}_j^r\}_{r \in [R]}$ come from the class. This fraction will be used to estimate the class's size s_i via a helper algorithm Estimate. Since the fraction is a function $f(s_i, \sum_{l>i} s_l)$ of s_i together with the sizes of previously estimated classes, Estimate can use the previous estimates with the value of f to estimate s_i.

There are a few "extra checks" performed by our algorithm in step 4 to ensure that Estimate is accurate. First, we only invoke Estimate on the family $\{\mathcal{S}_j^r\}_{r \in [R]}$ if the number of maxima crosses a certain threshold. Second, we make sure that both $\sum_{l>i} \tilde{s}_l > 2^j$ (if $\mathsf{Estimate}$ is invoked on the family $\{\mathcal{S}_j^r\}_{r \in [R]}$) and that $L \cdot 2^j$ is at least as large as the output of $\mathsf{Estimate}$. If either one of these conditions fails to hold, $\tilde{s}_i$ is set to 0. As the analysis will show, these conditions effectively allow $\mathsf{Estimate}$ to inductively approximate the s_i from the function f and its previous estimates.

We separate the description of the algorithm from its helper algorithm $\mathsf{Estimate}$ used in step 4.

F_k-Approximator (stream $\mathcal{S}$):

1. For $j \in [b]$ and $r \in [R]$, independently sample functions h_j^r: $[m] \to [2^j]$ from $O(\ln(1/L))$-wise independent families $\mathcal{H}_j$, or by using the PRG in section 3.5
2. Let $\mathcal{S}_j^r$ be the restriction of $\mathcal{S}$ to those items x for which $h_j^r(x) = 1$
3. For each j, R, compute $M_j^r = \mathsf{Max}(\mathcal{S}_j^r)$
4. For $i = \log n, \ldots, 0$,
   - Find the largest j for which at least $RL/64$ indices r satisfy $\alpha^i \le M_j^r < \alpha^{i+1}$. If no such j exists, or if $4\sum_{l>i} \tilde{s}_l > 2^j$, set $\tilde{s}_i = 0$.
   - Otherwise, set: $\mathsf{temp} = \mathsf{Estimate}(i, j, \sum_{l>i} \tilde{s}_l, M_j^1, \ldots, M_j^R)$
   - If $2^j \ge \mathsf{temp}/L$, set $\tilde{s}_i = \mathsf{temp}$, else $\tilde{s}_i = 0$
5. Output $\tilde{F}_k = \sum_i \tilde{s}_i \alpha^{ik}$.

Before describing the algorithm $\mathsf{Estimate}$, we observe that if the $\tilde{s}_i$ are good approximations to the s_i, then $\tilde{F}_k$ is a good approximation to F_k. More precisely, define the event $\mathcal{E}$ as follows:

- for all i, $0 \le \tilde{s}_i \le (1+\epsilon)s_i$, and
- for all i, if S_i contributes, then $\tilde{s}_i \ge (1 - \epsilon/(k+2))s_i$.

We claim that proving $\tilde{F}_k$ is a $(1 \pm \epsilon)$-approximation reduces to bounding the probability that event $\mathcal{E}$ occurs. More precisely,

Claim 4. *Suppose that with probability at least $8/9$, event $\mathcal{E}$ occurs. Then with probability at least $8/9$, we have $|\tilde{F}_k - F_k| \le \epsilon F_k$.*

Proof. Assume $\mathcal{E}$ occurs. Put $\epsilon^* = \epsilon/(k+2)$. Then,

$$\tilde{F}_k = \sum_i \tilde{s}_i \alpha^{ik} \le \sum_i (1+\epsilon) s_i \alpha^{ik} \le (1+\epsilon) F_k.$$

For the other direction, write $\tilde{F}_k = \tilde{F}_k^C + \tilde{F}_k^{NC}$, where $\tilde{F}_k^C$ denotes the contribution to $\tilde{F}_k$ due to the contributing S_i. Then, using Lemma 2, and assuming $\epsilon' \le \epsilon^*$,

$$\begin{aligned}
\tilde{F}_k^C &= \sum_{\text{contributing } S_i} \tilde{s}_i \alpha^{ik} \\
&\ge \frac{(1-\epsilon^*)}{\alpha^k} \sum_{\text{contributing } S_i} s_i \alpha^{(i+1)k} \\
&\ge \frac{(1-\epsilon^*)}{\alpha^k} F_k^C \\
&> \frac{(1-\epsilon^*)(1-\lambda\alpha^k)}{\alpha^k} F_k \\
&= (1-\epsilon^*)\left(\frac{1}{\alpha^k} - \lambda\right) F_k \ge \frac{(1-\epsilon^*)^2}{\alpha^k} F_k \\
&\ge (1-\epsilon^*)^2 (1 - k\epsilon^*) F_k \\
&\ge (1 - (k+2)\epsilon^*) F_k \\
&= (1-\epsilon) F_k.
\end{aligned}$$

Noting that $\tilde{F}_k^{NC} \ge 0$, we conclude that with probability at least $8/9$, we have $|\tilde{F}_k - F_k| \le \epsilon F_k$. $\square$

We now describe $\mathsf{Estimate}$, and in the next section prove it yields the premise of Claim 4. Define $r_{i,j} = (1 - (1-2^{-j})^{s_i})$. For any stream $\mathcal{S}_j^r$, the probability that $\alpha^i \le M_j^r < \alpha^{i+1}$ is precisely

$$p_{i,j} = (1-2^{-j})^{\sum_{l>i} s_l}(1-(1-2^{-j})^{s_i}) = (1-2^{-j})^{\sum_{l>i} s_l} r_{i,j}.$$

$\mathsf{Estimate}$ computes an approximation $\tilde{r}_{i,j}$ to $r_{i,j}$, and uses it to estimate s_i through the expression for $p_{i,j}$.

Estimate $(i, j, \sum_{l>i} \tilde{s}_l, M_j^1, \ldots, M_j^R)$:

1. Set $A_{i,j} = \#r$ for which $\alpha^i \le M_j^r < \alpha^{i+1}$
2. Set $\tilde{r}_{i,j} = \min\left(\frac{A_{i,j}}{R(1-2^{-j})^{\sum_{l>i} \tilde{s}_l}}, 1\right)$
3. Output $\frac{\ln(1-\tilde{r}_{i,j})}{\ln(1-2^{-j})}$

3.2 Analysis

The goal of the analysis is to prove the following theorem.

Theorem 5. *For sufficiently large m, n, with probability at least $8/9$, event $\mathcal{E}$ occurs.*

In the bulk of the analysis we will assume the h_j^r are truly random functions, but we remove this assumption in section 3.5 using the techniques of [10]. Alternatively, one can adapt the inclusion-exclusion approach used in [3] to our setting, where the families $\mathcal{H}_j$ should be chosen to be $O(\ln(1/L))$-wise independent. We omit the details of the latter approach. To simplify the analysis, we start by showing that with probability at least $8/9$, a very natural event occurs. We then condition on this event in what follows.

Observe that in $\mathsf{Estimate}$,

$$\mathbf{E}[A_{i,j}] = Rp_{i,j} = R(1-2^{-j})^{\sum_{l>i} s_l} r_{i,j}.$$

We define $\mathcal{F}$ to be the event that for all $A_{i,j}$,

- If $\mathbf{E}[A_{i,j}] \ge RL/(128e)$, then $|A_{i,j} - \mathbf{E}[A_{i,j}]| \le L\mathbf{E}[A_{i,j}]$.
- If $\mathbf{E}[A_{i,j}] \le RL/(128e)$, then $A_{i,j} < RL/64$.

LEMMA 6. $\Pr[\mathcal{F}] \geq 8/9$.

PROOF. Fix any i,j for which $\mathbf{E}[A_{i,j}] \geq RL/(128e)$. By Chernoff bounds,

$$\Pr[|A_{i,j}-\mathbf{E}[A_{i,j}]| \geq L\mathbf{E}[A_{i,j}]] \leq e^{-\Theta(L^2\mathbf{E}[A_{i,j}])} = O\left(\frac{1}{\ln m \log n}\right).$$

Now suppose $A_{i,j}$ is such that $\mathbf{E}[A_{i,j}] \leq RL/(128e)$. Then

$$\Pr[A_{i,j} \geq RL/64] \leq 2^{-RL/64} = O\left(\frac{1}{\ln m \log n}\right).$$

The lemma follows by a union bound over all i and j. $\square$

3.2.1 Consequences of $\mathcal{F}$

In the remainder, we assume that $\mathcal{F}$ occurs.

DEFINITION 7. *We say that* temp *is* **set** *if in step 4 of the main algorithm,* temp *is set to the output of* Estimate. *We say that $\tilde{s}_i$ is* **set** *if in step 4, $\tilde{s}_i$ is set to* temp.

We'd like to approximate s_i by approximating $p_{i,j}$. We start with the following proposition.

PROPOSITION 8. *Suppose $0 < s_i/L \leq 2^j$ and $0 < \gamma < 1$. Then if $\tilde{r}_{i,j} - r_{i,j} \leq \gamma r_{i,j}$, then $\tilde{s}_i$, defined by*

$$\tilde{s}_i = \frac{\ln(1-\tilde{r}_{i,j})}{\ln(1-1/2^j)},$$

satisfies $\tilde{s}_i - s_i \leq (\gamma + O(L))s_i$. If in addition, $r_{i,j} - \tilde{r}_{i,j} \leq \gamma r_{i,j}$, then $s_i - \tilde{s}_i \leq (\gamma + O(L))s_i$.

PROOF. By a Taylor expansion, $2^{-j} \leq -\ln(1-2^{-j}) \leq 2^{-j} + \eta_1$, where $\eta_1 = O(1/4^j)$. Similarly, $\tilde{r}_{i,j} \leq -\ln(1-\tilde{r}_{i,j}) \leq \tilde{r}_{i,j}+\eta_2$, where $\eta_2 = O(\tilde{r}_{i,j}^2)$. Since $s_i 2^{-j} - s_i^2 2^{-2j-1} \leq r_{i,j} \leq s_i 2^{-j}$, we have

$$\begin{aligned}
\tilde{s}_i &\leq 2^j(\tilde{r}_{i,j}+\eta_2) \leq 2^j(\tilde{r}_{i,j} + \tilde{r}_{i,j}O(r_{i,j})) \\
&\leq 2^j r_{i,j}(1+\gamma)(1+O(r_{i,j})) \\
&\leq s_i(1+\gamma+O(L))
\end{aligned}$$

If also $r_{i,j} - \tilde{r}_{i,j} \leq \gamma r_{i,j}$, then

$$\begin{aligned}
\tilde{s}_i &\geq \frac{\tilde{r}_{i,j}}{2^{-j}+\eta_1} \\
&\geq \frac{2^j(1-\gamma)r_{i,j}}{1+2^j\eta_1} \\
&\geq \frac{s_i(1-\gamma)(1-s_i 2^{-j-1})}{1+2^j\eta_1} \\
&\geq s_i(1-\gamma-O(L))
\end{aligned}$$

$\square$

LEMMA 9. *Suppose for some i and some $0 < \gamma < 1/3$, $|\sum_{l>i} s_l - \sum_{l>i} \tilde{s}_l| \leq \gamma \sum_{l>i} s_l$. If* temp *is set for S_i, then $|\tilde{r}_{i,j} - r_{i,j}| \leq (1+\gamma+2L)r_{i,j}$.*

PROOF. Fix such an i, put $\sigma = \sum_{l>i} s_l$, and $\sigma' = \sum_{l>i}\tilde{s}_l$. In Estimate, $\tilde{r}_{i,j} \leq \min\left(\frac{A_{i,j}}{R(1-2^{-j})^{\sigma'}}, 1\right)$. Also, $\mathbf{E}[A_{i,j}] \geq RL/(128e)$ since temp is set and $\mathcal{F}$ occurs. Recall that $\mathbf{E}[A_{i,j}] = R(1-2^{-j})^{\sigma} r_{i,j}$. Therefore,

$$(1-L)(1-2^{-j})^{\gamma\sigma} r_{i,j} \leq \tilde{r}_{i,j} \leq (1+L)(1-2^{-j})^{-\gamma\sigma} r_{i,j}.$$

Since temp is set and $\gamma < 1/3$, we have $2^j \geq 4\sigma' \geq 2\sigma$. Using standard bounds (see the appendix of [12]), one can show that

$$\begin{aligned}
(1-L)(1-2^{-j})^{\gamma\sigma} r_{i,j} &\geq (1-L)(1-\gamma)r_{i,j} \\
&\geq (1-L-\gamma)r_{i,j}
\end{aligned}$$

and similarly that

$$(1+L)(1-2^{-j})^{-\gamma\sigma} r_{i,j} \leq (1+L)(1+\gamma)r_{i,j} \leq (1+2L+\gamma)r_{i,j}.$$

$\square$

COROLLARY 10. *Suppose for some i and some $0 < \gamma < 1/3$, $|\sum_{l>i} s_l - \sum_{l>i}\tilde{s}_l| \leq \gamma \sum_{l>i} s_l$. Then $0 \leq \tilde{s}_i \leq s_i + (\gamma + O(L))s_i$.*

PROOF. If either temp or $\tilde{s}_i$ is not set, then $\tilde{s}_i = 0$ and we're done. Otherwise, $2^j \geq \tilde{s}_i/L$ in step 4. If $\tilde{s}_i < s_i$, since the output of Estimate is nonnegative, $0 \leq \tilde{s}_i < s_i$. Otherwise, $2^j \geq \tilde{s}_i/L \geq s_i/L > 0$, where the last inequality follows from the fact that temp is set. Since $|\tilde{r}_{i,j} - r_{i,j}| \leq (1+\gamma+2L)r_{i,j}$ by Lemma 9, applying Proposition 8 gives $\tilde{s}_i \leq s_i + (\gamma+O(L))s_i$. $\square$

LEMMA 11. *Let $0 < \gamma < 1/3$, $|\sum_{l>i}\tilde{s}_l - \sum_{l>i} s_l| \leq \gamma\sum_{l>i} s_l$, and suppose that S_i contributes. Then $|\tilde{s}_i - s_i| = (\gamma + O(L))s_i$.*

PROOF. Suppose S_i contributes, and put $\sigma = \sum_{l>i} s_l$. Consider j for which $8\sigma \leq 2^j < 16\sigma$. Then, recalling that $\mathbf{E}[A_{i,j}] = Rp_{i,j}$, one can bound $\mathbf{E}[A_{i,j}]$ below by $R(1-\sigma 2^{-j})s_i/2^{j+1}$ since $(1-(1-2^{-j})^{s_i}) \geq s_i/2^{j+1}$ for $2^j \geq s_i$. Note that $2^{j+1} < 32\sigma$. Moreover, $s_i \geq L\sigma$ since S_i contributes. Thus $\mathbf{E}[A_{i,j}]$ is at least $R(1-\sigma 2^{-j})L/32 \geq (7/8)(RL/32)$. Since $L < 3/7$, we have $A_{i,j} \geq (1-L)\mathbf{E}[A_{i,j}] \geq RL/64$. Since $8\sigma \geq 4\sigma'$, temp is set, so $|\tilde{r}_{i,j} - r_{i,j}| \leq (1+\gamma+2L)r_{i,j}$ by Lemma 9.

Now consider j for which $2s_i/L \leq 2^j < 4s_i/L$. The claim is that $\mathbf{E}[A_{i,j}] \geq \frac{RL}{8e}$. Note that if we show the claim, then it follows that $A_{i,j} \geq (1-L)RL/(8e) \geq RL/64$ since $L < 1/2$. To show the claim, by dividing by R, by using the upper bound on 2^j, and by using that $(1-(1-2^{-j})^{s_i}) \geq s_i/2^{j+1}$ for $2^j \geq s_i$, it is enough to show that $(1-2^{-j})^{\sigma} \geq e^{-1}$. If $\sigma = 0$, this is immediate. Otherwise, by Lemma 1, $s_i \geq L\sigma$, and thus $2^j \geq 2\sigma$. So it remains to show that $\left(1-\frac{1}{2\sigma}\right)^{\sigma} \geq e^{-1}$. But $\left(1-\frac{1}{2\sigma}\right)^{\sigma} \geq e^{-1/2}(1-\frac{1}{2\sigma})^{1/2} \geq e^{-1}$, as needed (again, see the bounds in the appendix of [12]).

Now we know for the j found in step 4 of the algorithm, we have $2^j \geq s_i/L > 0$. Proposition 8 implies $|\text{temp} - s_i| \leq (\gamma + O(L))s_i$. Finally, since $\gamma < 1/3$, it holds that for large enough n, $s_i \geq \text{temp}/2$, and thus, $2^j \geq 2s_i/L \geq \text{temp}/L$. In particular, $\tilde{s}_i$ is set. $\square$

Now define $\beta_i = (\log n + 1 - i)O(L)$ for $i = \log n + 1, \ldots, 0$. We show that in step 4 of the algorithm, when processing S_i, we have $|\sum_{l>i}\tilde{s}_i - \sum_{l>i} s_i| \leq \beta_{i+1}\sum_{l>i} s_i$ for all i. The only worry is that we may underestimate many of the s_i for those ranges for which either temp or $\tilde{s}_i$ is not set. However, these ranges cannot contain too many elements.

LEMMA 12. *For $i = \log n, \ldots, -1$,*

$$\left|\sum_{l>i}\tilde{s}_i - \sum_{l>i} s_i\right| \leq \beta_{i+1}\sum_{l>i} s_i.$$

PROOF. We induct downwards on i. The base case, $i = \log n$, is trivial. We show it for some $i < \log n$ assuming it holds for $i+1$. If S_i contributes, then $|\tilde{s}_i - s_i| \le (\beta_{i+1} + O(L))s_i \le \beta_i s_i$ by Lemma 11. Thus,

$$\begin{aligned} \left|\sum_{l\ge i}\tilde{s}_l - \sum_{l\ge i} s_l\right| &\le \left|\sum_{l>i}\tilde{s}_l - \sum_{l>i} s_l\right| + |\tilde{s}_i - s_i| \\ &\le \beta_{i+1}\sum_{l>i} s_l + \beta_i s_i \\ &\le \beta_i \sum_{l\ge i} s_l, \end{aligned}$$

proving the inductive step. If S_i does not contribute, then $s_i \le L\sum_{l>i} s_l$, and so

$$\begin{aligned} \left|\sum_{l\ge i}\tilde{s}_l - \sum_{l\ge i} s_l\right| &\le \left|\sum_{l>i}\tilde{s}_l - \sum_{l>i} s_l\right| + s_i \\ &\le \beta_{i+1}\sum_{l>i} s_l + L\sum_{l>i} s_l \\ &\le \beta_i \sum_{l\ge i} s_l, \end{aligned}$$

where we use that $|\tilde{s}_i - s_i|$ is maximized for $\tilde{s}_i = 0$, since, by Corollary 10, $\tilde{s}_i \le s_i + (\gamma + O(L))s_i$. □

Noting that $(\log n + 1)O(L) \le \epsilon/(k+2)$, Claim 4, Corollary 10, Lemma 11, and Lemma 12 imply that event $\mathcal{E}$ occurs.

3.3 A 2-pass Algorithm

We instantiate Assumption 3 with the CountSketch algorithm of [5].

THEOREM 13. *(**[5]**) Let $0 < \eta < 1$. Given a stream $\mathcal{S}$ and a parameter B, there is an algorithm* CountSketch *that with probability at least $1-\eta$ returns all items x for which $f_x^2 \ge F_2/B$. Further, the space complexity of* CountSketch *is $O(B(\ln n \ln 1/\eta + \ln m))$.*

For completeness, we briefly sketch the algorithm. It maintains $t = O(B)$ counters, $c[0]\ldots c[t-1]$, initially set to 0. It also maintains pseudorandom hash function $g : [m] \to [t]$ and pseudorandom variables Y_x, $x \in [m]$. Given a new element x, the i-th counter performs the following: if $g(x) = i$, then $c[x] = c[x] + Y_a$.

We modify F_k-Approximator as follows. In step 3 we invoke CountSketch, obtaining lists L_j^r of candidate maxima for $\mathcal{S}_j^r$. In parallel, we compute a 2-approximation $\tilde{F}_2(\mathcal{S}_j^r)$ to $F_2(\mathcal{S}_j^r)$ for each j and r. Then, before invoking steps 4 and 5, we make a second pass over $\mathcal{S}$ and compute the true frequency of each element of each L_j^r. Since the L_j^r are relatively small, this is efficient. We then prune the lists L_j^r by removing all items with frequency less than $2\tilde{F}_2(\mathcal{S}_j^r)$. We set M_j^r to be the maximum frequency amongst the remaining items in L_j^r, if there is at least one remaining item. Otherwise we set $M_j^r = 0$. At the end of the 2nd pass, we proceed with steps 4 and 5 as before.

We note that since both the CountSketch algorithm and the F_2 algorithm have $\tilde{O}(1)$ update time and can handle deletions, our algorithm also has these properties.

We start by conditioning on the following event:

$$\mathcal{G}_1 \stackrel{\text{def}}{=} \forall j, r,\ \textsf{CountSketch succeeds.}$$

The next lemma follows by a union bound.

LEMMA 14. *If we set $\eta = O(1/(bR))$ in Theorem 13, then* $\Pr[\mathcal{G}_1] \ge 8/9$.

To further simplify matters, we condition on the event:

$$\mathcal{G}_2 \stackrel{\text{def}}{=} \forall j, r,\ F_2(\mathcal{S}_j^r) \le \frac{9bR \cdot F_2(\mathcal{S})}{2^j}.$$

LEMMA 15. $\Pr[\mathcal{G}_2] \ge 8/9$.

PROOF. We have $\mathbf{E}[F_2(\mathcal{S}_j^r)] = F_2(\mathcal{S})/2^j$, so

$$\Pr\left[F_2(\mathcal{S}_j^r) \ge 9bRF_2(\mathcal{S})/2^j\right] \le 1/(9bR)$$

by Markov's inequality. By a union bound over all j, r, $\Pr\left[\exists j, r \mid F_2(\mathcal{S}_j^r) \ge 9bRF_2(\mathcal{S})/2^j\right] \le 1/9$. □

Finally, define the event

$$\mathcal{G}_3 \stackrel{\text{def}}{=} \forall j, r,\ \tilde{F}_2(\mathcal{S}_j^r) \le 2F_2(\mathcal{S}_j^r).$$

LEMMA 16. $\Pr[\mathcal{G}_3] \ge 8/9$.

PROOF. We may run the F_2-approximator of [1] or [14] in space $O((\ln m + \ln n)\ln 1/\eta)$, where $\eta = 1/(9bR)$ is the failure probability. A union bound gives the lemma. □

We set the CountSketch parameter $B = O(bRm^{1-2/k})$.

Consequences of $\mathcal{F}, \mathcal{G}_1, \mathcal{G}_2$, and $\mathcal{G}_3$: We start with a technical claim.

CLAIM 17. *For all j, r, either M_j^r is set to the maximum frequency in $\mathcal{S}_j^r$, or to 0.*

PROOF. Suppose for some j, r the pruned list L_j^r contains at least one element x of $\mathcal{S}_j^r$, but does not contain the most frequent element y of $\mathcal{S}_j^r$. Then, since $\mathcal{G}_1$ occurs, $f_x^2 < f_y^2 < F_2(\mathcal{S}_j^r)/B$. Since $\mathcal{G}_3$ occurs, we have $f_x^2 < F_2(\mathcal{S}_j^r)/B \le 2\tilde{F}_2(\mathcal{S}_j^r)/B$, which contradicts the fact that x wasn't pruned. □

COROLLARY 18. *For all i, j, $\mathbf{E}[A_{i,j}] \le Rp_{i,j}$.*

PROOF. Follows from the fact that $\Pr[M_j^r] = 1$ can only decrease. □

We have the following adaptation of Lemma 9 and Corollary 10.

LEMMA 19. *Suppose for some i and some $0 < \gamma < 1/3$, $|\sum_{l>i} s_l - \sum_{l>i}\tilde{s}_l| \le \gamma\sum_{l>i} s_l$. Then $0 \le \tilde{s}_i \le s_i + (\gamma + O(L))s_i$.*

PROOF. We may assume temp is set for S_i, otherwise $\tilde{s}_i = 0$. The main difference from Lemma 9 is that now $\mathbf{E}[A_{i,j}] \le R(1-2^{-j})^{\sigma} r_{i,j}$, by Corollary 18. We use half of the proof of Lemma 9 to conclude that $\tilde{r}_{i,j} \le (1 + 2L + \gamma)r_{i,j}$. If $\tilde{s}_i$ is not set or if $\tilde{s}_i \le s_i$, we are done. Otherwise $2^j \ge \tilde{s}_i/L \ge s_i/L > 0$. and the lemma follows by Proposition 8. □

Now we show that Lemma 11 still holds.

LEMMA 20. *Let $0 < \gamma < 1/3$, $|\sum_{l>i}\tilde{s}_l - \sum_{l>i} s_l| \le \gamma\sum_{l>i} s_l$, and suppose S_i contributes. Then $|\tilde{s}_i - s_i| = (\gamma + O(L))s_i$.*

PROOF. Since $\mathcal{F}$ occurs, from the proof of Lemma 11, it is enough to show that for $2^j \geq 2s_i/L$, the values M_j^r are distributed just as under Assumption 3. That is, they are i.i.d. Bernoulli($p_{i,j}$). Indeed, if it happens that $2s_i/L \geq 8\sigma$, then temp will be set as before. If instead $2s_i/L \leq 8\sigma$, then in particular, the values M_j^r are identically distributed for $2^j \geq 8\sigma$, and temp will be set as in the proof of Lemma 11.

Since $\mathcal{G}_1$ and $\mathcal{G}_3$ occur, it suffices to show that for all $x \in S_i$ and all r, $f_x^2 \geq 4F_2(\mathcal{S}_j^r)/B \geq 2\tilde{F}_2(\mathcal{S}_j^r)/B$ (thus for $y \in S_{i'}, i' > i$, $f_y^2 \geq 2\tilde{F}_2(\mathcal{S}_j^r)/B$). Applying Hölder's inequality, $F_2 = \sum_{i=1}^m f_i^2 \cdot 1 \leq \left(\sum_{i=1}^m f_i^k\right)^{2/k} \left(\sum_{i=1}^m 1\right)^{1-2/k} = F_k^{2/k} m^{1-2/k}$. Using that $f_x^k s_i \geq \alpha^{ik} s_i \geq LF_k$, we have

$$\alpha^{2i} s_i^{2/k} \geq \frac{F_2 L^{2/k}}{m^{1-2/k}} \geq \frac{2^j \cdot F_2(\mathcal{S}_j^r) L^{2/k}}{9bR \cdot m^{1-2/k}} \geq \frac{2^{j+1} F_2(\mathcal{S}_j^r) L^{2/k}}{B},$$

since $\mathcal{G}_2$ occurs. Since $k \geq 2$, we have $2^j/s_i^{2/k} \geq 2^j/s_i$. If $2^j \geq 2s_i/L$, since $L^{2/k}/L \geq 1$ for $k \geq 2$, we have $f_x^2 \geq 4F_2(\mathcal{S}_j^r)/B$, as needed. □

As the proof of Lemma 12 is unchanged, we conclude,

THEOREM 21. *There exists a 2-pass algorithm which (ϵ, δ)-approximates F_k in space $\tilde{O}(m^{1-2/k})$.*

PROOF. The lemmas above show that if $\mathcal{F} \wedge \mathcal{G}_1 \wedge \mathcal{G}_2 \wedge \mathcal{G}_3$ occurs, then so does $\mathcal{E}$. By a union bound, $\Pr[\mathcal{F} \wedge \mathcal{G}_1 \wedge \mathcal{G}_2 \wedge \mathcal{G}_3] \geq 5/9$. Taking the median of $O(\ln 1/\delta)$ independent repetitions shows the output is an (ϵ, δ)-approximation to F_k.

Clearly the algorithm can be implemented with 2 passes. Ignoring $\ln(\ln mn/\epsilon)$ factors, Theorem 13 implies the total space is (recall, $\log n = \log_{1+\alpha} n = O\left(\frac{\ln n}{\epsilon}\right)$)

$$\begin{aligned} & O\left(bR \cdot B\left(\ln n + \ln m\right) \ln \frac{1}{\delta}\right) \\ = \quad & O\left(b^2 R^2 m^{1-2/k}\left(\ln n + \ln m\right) \ln \frac{1}{\delta}\right) \\ = \quad & O\left(\frac{\ln^2 m \ln^6 n}{\epsilon^{12}} m^{1-2/k}\left(\ln n + \ln m\right)\right). \end{aligned}$$

□

3.4 The 1-pass Algorithm

In this section we show how to remove the second pass from the algorithm in the previous section, and obtain a 1-pass algorithm. For convenience, let $f(i)$ denote the frequency of item i.

Recall that, in the previous section, the algorithm assumed an oracle that we refer to as Partial Max. For a certain value of a threshold T, the oracle reported the element $i^* \in [m]$ with the maximum value of $f(i^*)$, but if and only if $f(i^*) \geq T$. The second pass was needed in order to compute the exact frequencies of the candidate maxima, and check if (a) any of them was greater than T and (b) find the element with the maximum frequency.

We reduce the need of the second pass by transforming the algorithm in such a way that, if we replace each frequency $f(i)$ by its estimation $\tilde{f}(i)$ provided by Countsketch, the behavior of the transformed algorithm is, with high probability, the same as in the original algorithm.

We assume the following event holds for the items reported by CountSketch:

$$f(i) \leq \tilde{f}(i) \leq (1+\kappa) f(i)$$

Note that this event holds with probability $1 - o(1)$ if the space used by the algorithm is multiplied by a factor polynomial in $1/\kappa$

The transformations are as follows.

Shifted boundaries. We modify the algorithm so that the thresholds T passed to the Partial Max are multiplied by some value $y \in [1, \alpha)$, and the frequency boundaries α^i are multiplied (consistently) by some value $x \in [1/\alpha, 1)$. The algorithm and its analysis of correctness can be easily adapted to this case. This requires natural adjustments, such as replacing each term α^i in the estimator by $(x\alpha)^i$, etc.

The reason for this modification is that, if we choose x, y independently at random from a near-uniform distribution, then, for *fixed* i, the outcome of comparisons , say, $f(i) \geq yT$ and $\tilde{f}(i) \geq yT$, is likely to be the same, as long as $f(i)$ and $\tilde{f}(i)$ differ by a small multiplicative factor.

Class reporting. We replace the Partial Max oracle by another oracle, called Rounded Partial Max, which does the following: for a permutation $\pi : [m] \rightarrow [m]$, it reports i with the smallest value of $\pi[i]$ such that $f(i)$ is in the same frequency class as $f(i^*)$, but only if $f(i^*) \geq yT$. The algorithm and its analysis remains unchanged, since it only performs comparisons of $f(i)$ with values $x\alpha^i$ and yT.

In the following we assume π is choosen uniformly at random from the set of all permutations of $[m]$. Later, we show how to reduce the required randomness by choosing π from a family of 2-approximate min-wise independent functions [11].

Approximate frequencies. Now we consider the following key modification to Rounded Partial Max. The modification replaces the use of the exact frequencies $f(i)$ by their approximations $\tilde{f}(i)$. Specifically, we replace each comparison $f(i) \geq v$ by a comparison $\tilde{f}(i) \geq v(1+\kappa)$. Note that $\tilde{f}(i) \geq v(1+\kappa)$ implies $f(i) \geq v$. Call the resulting oracle Approximate Max.

Let i' be such that $\pi(i')$ is the smallest value of $\pi(i)$ over all i for which $f(i)$ is in the same frequency class as $f(i^*)$, and let $[x\alpha^{k'}, x\alpha^{k'+1})$ be the frequency class containing $f(i^*)$. If we invoke Rounded Partial Max with parameters j, r, we denote the values i', i^*, k', T by $i'(j,r)$, $i^*(j,r)$, $k'(j,r)$ and $T(j,r)$. Consider the following event $\mathcal{B}(j,r)$:

1. $f(i'(j,r)) \geq x\alpha^{k'(j,r)}(1+\kappa)$, and
2. $f(i^*(j,r)) \geq yT(j,r) \Rightarrow f(i^*(j,r)) \geq yT(j,r)(1+\kappa)$

The following claim can be verified by tracing through the steps of the algorithm.

CLAIM 22. *Fix the random bits of the algorithm. If $\mathcal{B}(j,r)$ holds for all j, r, then the behaviors of all invocations of* Rounded Partial Max *and* Approximate Max*, respectively, are exactly the same. Therefore, the output of the algorithms using either oracle is the same.*

Now it suffices to show that each $\mathcal{B}(j,r)$ holds with good probability. We show that this holds even if π is chosen from a family of 2-approximate min-wise permutations i.e., such that for any $A \subset [m], a \in [m] - A$, we have $\Pr_\pi[\pi(a) < \min_{b \in A} \pi(b)] \leq \frac{2}{|A|+1}$. Such families exist and are constructible from only $O(\log^2 m)$ random bits [11].

LEMMA 23. *There is a distribution of x so that, for any $0 < \zeta < 1$, if $0 < \kappa < 1 < \alpha = 1 + \epsilon' < 2$, then for a fixed* *pair (j,r) the probability of*

$$f(i'(j,r)) < x\alpha^{k'(j,r)}(1+\kappa)$$

is at most $O(\kappa/\epsilon' \cdot \log m \cdot 1/\zeta + \zeta)$. Moreover, this fact holds even if π is chosen at random from a family of 2-wise functions.

PROOF. For simplicity, we are going to omit the pair (j,r) in the notation below.

For a parameter β, define $I'(\beta) = |\{i : \beta \leq f(i)\}|$, and $I'' = |\{i : \beta \leq f(i) < \beta(1+\kappa)\}|$. Consider $\beta = x\alpha^{k'}$. Observe that the event we are concerned in this lemma occurs if $i' \in I''(\beta)$.

We choose $x = (1+\kappa)^s/\alpha$, where s is chosen uniformly at random from $\{0,\ldots,\log_{1+\kappa}\alpha\}$. Note that $\log_{1+\kappa}\alpha = \Theta(\epsilon'/\kappa)$.

Observe that the value of β ranges in $[f(i^*)/\alpha \ldots f(i^*)]$. Also, observe that each value in that interval is assumed at most once (that is, for only one value of x).

CLAIM 24. *For any $0 < \zeta < 1$, the number of different values of β such that $I''(\beta)/I'(\beta) \geq \zeta$ is at most $\log_{1+\zeta}(m+1)+1 = O(\log m/\zeta)$.*

PROOF. Assume this is not the case, and let $\beta_1 \ldots \beta_t$, $t \geq \log_{1+\zeta}(m+1)+1$ be the different values of β such that $I''(\beta)/I'(\beta) \geq \zeta$, in decreasing order.

Since $I'(\beta) = I''(\beta) + I'(\beta(1+\kappa))$, we have that for each β_i, $I'(\beta_i) \geq \frac{1}{1-\zeta} I'(\beta_i(1+\kappa)) \geq (1+\zeta)I'(\beta(1+\kappa))$. Moreover, the value of $I'(\beta)$ does not decrease as β decreases. It follows that $I'(\beta_t) \geq (1+\zeta)^{t-1} > m$, which is a contradiction. $\square$

Thus, for the value β induced by a random choice of x, the probability that $I''(\beta)/I'(\beta) \geq \zeta$ is at most $O(\kappa/\epsilon' \log_{1+\zeta} m)$. The probability that i' belongs to $I''(\beta)$ is at most ζ (if π is a truly random permutation) or at most 2ζ (if π is chosen from 2-approximate min-wise independent family).

The other part of the event $\mathcal{B}(j,r)$ can be handled in a similar way.

By setting $\zeta = \sqrt{\kappa/\epsilon' \cdot \log m}$ we get

LEMMA 25. *The probability that* some *event $\mathcal{B}(j,r)$ does not hold is at most*

$$O(Rb\sqrt{\kappa/\epsilon' \cdot \log m})$$

which is $o(1)$ for small enough $\kappa = 1/(1/\epsilon' + \log m)^{O(1)}$.

3.5 Reducing the randomness

It remains to show that the functions h_j^r used by our algorithm can be generated in small space. To this end, we will use Nisan's generator, as in [10]. Specifically, observe that the state maintained by algorithm consists of several counters c (as in the CountSketch algorithm). Each counter is identified by indices j, r and i. Given a new element x, the counter performs the following operation: if $h_j^r(x) = 1$ and $g(x) = i$, then $c = c + Y_x$. Therefore, we can use Lemma 3 of [10], to show that the random numbers $h_j^r(0), \ldots h_j^r(m-1)$ can be generated by a PRG using only $O(\log^2(nm))$ truly random bits as a seed. Thus, the total number of random bits we need to store is bounded by the total storage used by our 1-pass algorithm times $O(\log(nm))$.

4. REFERENCES

[1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the 28th Annual ACM Symposum on the Theory of Computing (STOC)*, pages 20-29, 1996. Also; J. Comp. Sys. Sci. 58, pages 137–147, 1999.

[2] Z. Bar Yossef, T.S. Jayram, R. Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. In *Proceedings of the 43rd Symposium on Foundations of Computer Science (FOCS)*, pages 209–218, 2002.

[3] Z. Bar Yossef, T.S. Jayram, R. Kumar, D. Sivakumar, and Luca Trevisan. Counting distinct elements in a data stream. *RANDOM 2002, 6th. International Workshop on Randomization and Approximation Techniques in Computer Science*, p. 1-10, 2002.

[4] A. Chakrabarti, S. Khot, and X. Sun. Near-optimal lower bounds on the multiparty communication complexity of set-disjointness. In *Proceedings of the 18th IEEE Conference on Computational Complexity (CCC)*, pages 107–117, 2003.

[5] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Theor. Comput. Sci.* **312(1)**, pages 3-15, 2004 (Extended Abstract appeared in *Proceedings of the 29th International Colloquium on Automata Languages and Programming (ICALP)*, 2002).

[6] D. Coppersmith and R. Kumar, An improved data stream algorithm for frequency moments. In *Proc of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 151-156, 2004.

[7] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. An Approximate L1-Difference Algorithm for Massive Data Streams. In *SIAM Journal on Computing* **32**, pages 131-151, 2002 (Extended Abstract appeared in *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS), 1999*).

[8] S. Ganguly. Estimating Frequency Moments of Data Streams using Random Linear Combinations. In *RANDOM*, 2004.

[9] S. Ganguly. A Hybrid Technique for Estimating Frequency Moments over Data Streams. Unpublished Manuscript, 2004.

[10] P. Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. In *Journal of the ACM*, to appear. Preliminary version in *Proceedings of the 41st Symposium on Foundations of Computer Science (FOCS)*, pages 187-197, 2000.

[11] P. Indyk. A Small Approximately Min-Wise Independent Family of Hash Functions. In *Journal of Algorithms* **38**, 2001.

[12] R. Motwani and P. Raghavan. Randomized Algorithms. Cambridge University Press, 1995.

[13] M. Saks and X. Sun. Space lower bounds for distance approximation in the data stream model. In *Proc of the 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 360-369, 2002.

[14] M. Thorup and Y. Zhang. Tabulation based 4-universal hashing with applications to second moment estimation. In *Proc of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*

(SODA), pages 615-624, 2004.