University of Pennsylvania
ScholarlyCommons

Departmental Papers (CIS) | Department of Computer & Information Science

January 2009

Tight results for clustering and summarizing data streams

Sudipto Guha
University of Pennsylvania, sudipto@cis.upenn.edu

Follow this and additional works at: http://repository.upenn.edu/cis_papers

Recommended Citation
Sudipto Guha, "Tight results for clustering and summarizing data streams", . January 2009.

This paper is posted at ScholarlyCommons. http://repository.upenn.edu/cis_papers/394
For more information, please contact libraryrepository@pobox.upenn.edu.

Tight results for clustering and summarizing data streams

Abstract
In this paper we investigate algorithms and lower bounds for summarization problems over a single pass data stream. In particular we focus on histogram construction and K-center clustering. We provide a simple framework that improves upon all previous algorithms on these problems in either the space bound, the approximation factor or the running time. The framework uses a notion of ``streamstrapping'' where summaries created for the initial prefixes of the data are used to develop better approximation algorithms. We also prove the first non-trivial lower bounds for these problems. We show that the stricter requirement that if an algorithm accurately approximates the error of every bucket or every cluster produced by it, then these upper bounds are almost the best possible. This property of accurate estimation is true of all known upper bounds on these problems.

Keywords
data streams, clustering

This conference paper is available at ScholarlyCommons: http://repository.upenn.edu/cis_papers/394

Tight results for clustering and summarizing data streams

Sudipto Guha*

Abstract

In this paper we investigate algorithms and lower bounds for summarization problems over a single pass data stream. In particular we focus on histogram construction and K-center clustering. We provide a simple framework that improves upon all previous algorithms on these problems in either the space bound, the approximation factor or the running time. The framework uses a notion of "streamstrapping" where summaries created for the initial prefixes of the data are used to develop better approximation algorithms. We also prove the first non-trivial lower bounds for these problems. We show that the stricter requirement that if an algorithm accurately approximates the error of every bucket or every cluster produced by it, then these upper bounds are almost the best possible. This property of accurate estimation is true of all known upper bounds on these problems.

1 Introduction

In the single pass data stream model any input data which is not explicitly stored cannot be accessed again. For a variety of these problems there exist small space, offline algorithms with optimal or good approximation. These algorithms typically find an appropriate granularity at which it inspects the data. In a streaming setting the problem is that by the time we have found the correct granularity, we have already seen a significant portion of the stream and unlike the offline algorithms we cannot revisit the stream. This manifests in the case of several clustering and summarization problems. A typical way of addressing this challenge has been to run the algorithm for a number of eventualities and to pick the best solution at the end of input. This results in space bound of these algorithms to depend on (logarithm of) the magnitude of the optimum solution $\mathcal{E}^*$ or the inverse of the smallest nonzero number that can be represented (machine precision) M. This raises the main question we address in this paper:

Question 1. *Is it possible to design clustering and summarization algorithms for data streams whose space requirements do not depend on $n, \mathcal{E}^*, M$? What is the best achievable approximation ratio under this restriction on space?*

The above question is motivated both by theory and practice. From a theoretical point of view, the question of minimum space is a natural one and the question of a space bound which is independent of n (and other input parameters) harks back to the celebrated results on ϵ-nets

*Department of Computer and Information Sciences. Philadelphia 19104. Email: sudipto@cis.upenn.edu. Research supported in part by an Alfred P. Sloan Research Fellowship and NSF awards CCF-0430376 and CCF-0644119. ©ACM, 2009. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version will be published in Proceedings of ICDT, (2009)

[22], which are independent of the size of the input. Such input parameter independent bounds are extremely useful building blocks for other algorithms. Also as $n, \mathcal{E}^*, M$ increase, it seems there is less information in any B term approximation of the signal – using more space when the information decreases is absolutely counter-intuitive! However note that we are seeking algorithms that considers all input points, and not just a large subsample. From an implementation perspective, if the space used depends on $n, \mathcal{E}^*, M$ then several messy complications, including growing an initial memory allocation, are introduced. Further, reducing the space below the cache size speeds up streaming algorithms significantly.

In the main result in this paper we show that for clustering and summarization problems which satisfy some simple criteria we can achieve streaming approximation algorithms whose space bounds are independent of $n, M\epsilon^*$ and are almost the best possible. We show that there exists an opportunity to bootstrap or "streamstrap" streaming algorithms where we use We can use the summaries of the prefixes of the data to inform us of the correct level of detail we need to be investigating the data. As a consequence we get summarization algorithms whose space bounds are independent of $n, M, \mathcal{E}^*$. We focus on two summarization problems in this paper – the k-center problem and the maximum error histogram construction problem. We also show that the ideas extend to more complicated minsum objective functions.

Clustering is one of the most extensively used summarization technique. In this paper we focus on K-center clustering in arbitrary metric spaces, in a model which is known as the *Oracle Distance Model*. In this model, given two points, p_1, p_2 we have an oracle that uses small additional space and determines their distance. The goal, given n points $P = p_1, \ldots, p_n$ is to identify K centers $p_{i_1}, \ldots, p_{i_k}$ such that $\max_{x \in P} \min_{j \leq K} d(x, p_{i_j})$ is minimized. In other words, we are asked to find the smallest radius $\mathcal{E}^*$ such that if disks of radius $\mathcal{E}^*$ are placed on the chosen centers then every input point is covered. The minsum variant of this problem is the well known k-median clustering problem where we seek to minimize $\sum_{x \in P} \min_{j \leq K} d(x, p_{i_j})$.

The oracle distance model allows us to consider complicated metric spaces which are difficult to embed in known and simpler metric spaces (for example, euclidean, hamming). With the growth of richer web applications, analysis of blog posts, this model of clustering will only grow in relevance. However a downside of the oracle distance model is that unless p_1, p_2 are stored, their distance can only be imputed based on other information stored. In an early result, Charikar et. al [6] gave a single pass streaming 8 approximation algorithm which uses $O(K)$ space. Note that based on the NP-Hardness of deciding if a dominating set of size K exists, achieving an approximation ratio better than 2 for the K-center problem is NP-Hard. It is possible to achieve a $2(1+\epsilon)$ approximation with a space bound of $O(\frac{K}{\epsilon} \log(M\mathcal{E}^*)$, in a streaming setting using geomteric discretization of the distances.

The **histogram construction** problem is defined as: given a sequence of n numbers, $x_1, \ldots, x_n$ representing a vector $X \in \mathcal{R}^n$, construct a piecewise constant representation H with at most B pieces such that a suitable objective function $f(X, H)$ is minimized. For example, the VOPT histogram problem seeks to minimize $\|X - H\|_2^2$, the maximum error histogram seeks to optimize $\|X - H\|_\infty$. These have recently been used in approximate query answering [1], time series mining [5], curve simplification [3]. In query optimization, after the frequencies have been aggregated, the serial histograms considered by Ioannidis [18] correspond to piecewise constant representation. This initiated a lot of research leading up to the dynamic programming algorithms provided by Jagadish et. al [19]. Since the end use of histograms is approximation, it was natural to consider approximation algorithms for histograms which was addressed in [12, 13]. These works also pro-

vided streaming algorithms for histogram construction, namely when $\ldots, x_i, \ldots$ were provided one at a time in increasing order of i and the algorithms are restricted to use sublinear space. Since then a large number of algorithms have been proposed, for many different measures and in particular the maximum error, many of which extend to streaming algorithms [16, 4, 20]. However for *every* algorithm proposed till date, for any error measure, the space bound depends either on $\log n, \log \mathcal{E}^*$ or $\log M$. As in the K-center problem, these streaming algorithms depend on geometric discretization.

Our Contribution: We focus on single pass insertion only (no deletion or updates) streaming algorithms. We begin with the results for specific problems:

1. For the model where $\ldots, x_i, \ldots$ are presented in increasing order of i, we provide a $(1+\epsilon)$ approximation algorithm for the maximum error and VOPT error histogram construction with space requirements $\frac{B}{\epsilon}\log\frac{1}{\epsilon}$ and $\frac{B^2}{\epsilon}\log\frac{1}{\epsilon}$ respectively, which are independent of $n, \mathcal{E}^*, M$. The running time of both algorithms are $O(n)$ plus smaller order terms. For the VOPT error this improves the previous best space bound of an algorithm with $O(n)$ running time by a factor B. For the maximum error, when $\epsilon \leq 1/(40B)$, we show that an algorithm must use use $\Omega(\frac{B}{\epsilon \log \frac{B}{\epsilon}})$ space if it simultaneously achieves a (i) a $(1+\epsilon)$ approximation and (ii) for each of the buckets produced in the solution approximates the error of that bucket to additively within ϵ times optimum of the actual error of that bucket in the solution. Observe that the second requirement is natural for any good summarization algorithm – and all previous algorithms as well as the two new ones we propose obey this property. This is the first lower bound for any histogram construction algorithm which is stronger than $\Omega(B)$. We note that the difficulty of proving a lower bound lies in the fact that the $\ldots, x_i, \ldots$ are presented in increasing order of i, which does not conform to known lower bound techniques for data streams where the arbitrary order of input is critical for lower bounds.

2. For the K-center problem, in the oracle distance model, we provide the first $2(1+\epsilon)$ approximation using space $O(\frac{K}{\epsilon}\log\frac{1}{\epsilon})$ which is independent of $n, M, \mathcal{E}^*$. Our setup easily extends to near optimal results for weighted K-centers. We show that this method improves the approximation ratios for the streaming k-median clustering; however it does not improve previous space bounds which depend on $\log^2 n$. For $\epsilon \leq 1/10K$, we also show that if a deterministic algorithm simultaneously provides $2+\epsilon$ approximation as well as approximates the radius of the clusters it produces to additively within ϵ times the optimum, then the algorithm must store $\Omega(\frac{K}{\epsilon}) = \Omega(K^2)$ points. As in histograms, this requirement means that the clustering produced is sufficiently tight for every cluster.

From a point of view of *techniques*, all the upper bounds follow the same framework. We use three main ideas: (i) we use the notion of a "thresholded approximation" where the goal is to minimize the error assuming we know the optimum error [1], (ii) we run multiple copies (but controlled in number) of the algorithm corresponding to different estimates of the final error and, (iii) we use a "streamstrapping" procedure to use partially completed summarization for a certain estimate to create summarization for a different estimate of error. The first two ideas have been explicitly used in the context of summarization before, see [4, 9, 10, 20, 11] among many others. We are unaware of the use of the third idea in any previous work and we believe that this notion will be

[1]The thresholded approximation is similar to, but not the same as, approximating the "dual" problem of minimizing size subject to a fixed error.

useful in a variety of different problems. Interestingly, the formalization of the general framework also provides results superior to all known algorithms for several summarization problems.

In terms of lower bounds, we provide the first non-trivial lower bounds for these problems. Further, we use the fact that summarization typically entails a tight guarantee (per point, per bucket or per cluster) to develop novel and strengthened lower bounds in this paper. While several of our results are almost tight (upto factors of $\log \frac{B}{\epsilon}$) many interesting open questions remain.

Roadmap: We present the upper bounds in Section 2. We then prove the lower bound for histograms in Section 3 and the lower bounds for the K-center problem in Section 4.

2 Upper bounds

In this section we provide a framework that simultaneously handles a variety of summarization problems. Let $\mathcal{P}$ be a summarization problem with space constraint B with input X. As easy running examples, consider $\mathcal{P}$ to be the maximum error histogram construction problem or the K-center problem.

2.1 The setup: Requirements

Consider summarization scenarios where the following conditions apply:

- *Thresholded small space approximations exist.* For a problem $\mathcal{P}$ a "thresholded approximation" is defined to be an algorithm which simultaneously guarantees that (i) if there is a solution with summarization size B' and error $\mathcal{E}$ (and we know $\mathcal{E}$), then in small space we can construct a summary of size at most B' such that the error of our summary is at most $\alpha\mathcal{E}$ for some $\alpha \geq 1$ and (ii) otherwise declare that no solution with error $\mathcal{E}$ exists.

- *The error measure is a Metric error.* Let $\mathcal{E}(Y, H)$ be the summarization error of Y if the summary is H. Let $X \circ Y$ denote concatenation of the input X followed by Y and let $X(H)$ be the input where every input $X \in X$ is replaced by the corresponding element $\hat{x}$ generated from H which best represents x. The $\mathcal{E}$ is defined to be a *Metric error* if for any X, Y, H, H' we have:

$$\mathcal{E}(X(H) \circ Y, H') - \mathcal{E}(X, H) \leq \mathcal{E}(X \circ Y, H') \leq \mathcal{E}(X(H) \circ Y, H') + \mathcal{E}(X, H)$$

For the K-center problem: Hochbaum and Shmoys [17] gave a thresholded approximation algorithm using $O(K)$ space with $\alpha = 2$. Given a threshold $\mathcal{E}$ the algorithm maintains a set S such that all points are within distance $2\mathcal{E}$ from at least one member of S, and every point that violates the condition is added to S. The space required is the size of S which is at most K (or the estimate $\mathcal{E}$ is wrong). To see that the clustering radius defines a metric error – consider a clustering given by H and and replace a point x by its closest center in H. The fact that the underlying distances form a metric space and satisfy triangle inequality completes the argument that the metric error property holds.

Thresholded approximation has been used in the context of histograms before, in the context of "dual" problems where the summary size is minimized to achieve a predetermined error [4, 20, 11]. Concretely, recall that the maximum error histogram construction problem is: given a set

of numbers $X = x_1, x_2, \ldots, x_n$ construct a piecewise constant representation H with at most B pieces such that $\|X - H\|_\infty$ (both of these are vectors in $\mathcal{R}^n$) is minimized. Now, $\mathcal{E}(X \circ Y, H') = \|X \circ Y, H'\|_\infty$ and

$$\|X(H) \circ Y, H'\|_\infty - \|X \circ Y, X(H) \circ Y\|_\infty \leq \|X \circ Y, H'\|_\infty \leq \|X(H) \circ Y, H'\|_\infty + \|X \circ Y, X(H) \circ Y\|_\infty$$

Now $\mathcal{E}(X(H) \circ Y, H') = \|X(H) \circ Y, H'\|_\infty$ and $\mathcal{E}(X, H) = \|X \circ Y, X(H) \circ Y\|_\infty$, and thus the error measure for the maximum error histogram problem is a metric error. This property also holds for the *square root* of the VOPT error, which is the ℓ_2 norm.

A thresholded optimum algorithm for the maximum error problem is as follows (see also [15]): observe that if we are to approximate a set of numbers then the best representation is $(max+min)/2$ and the error is $(max - min)/2$. So a simple implementation reads the numbers in the input and keep a running max and min. If $max - min > 2\mathcal{E}$ at some point (the knowledge of $\mathcal{E}$ is used here) then the numbers read so far are declared to be in one bucket and a new bucket is started. This is a greedy algorithm and it is easy to prove by induction over B' that the greedy algorithm will never use more than B' buckets. To complete the algorithm, we observe that the min, max are defined by the set of $\binom{n}{2}$ intervals and can be found by binary search. Thus for maximum error histograms we have an approximation with $\alpha = 1$. The space requirement is $O(B')$.

2.2 The solution: The StreamStrap Algorithm

Consider the algorithm given if Figure 1.

Algorithm StreamStrap:

1. Read the first B items in the input. This should have summarization error 0 for any reasonable measure since the entire input is stored. Keep reading the input as long as the error is 0.

2. Suppose we see the first input which causes non-zero error. The error has to be at least $1/M$ where M is the largest number possible to represent in the machine. Let this error be $\mathcal{E}_0$.

3. Initialize and run the thresholded algorithm for $\mathcal{E} = \mathcal{E}_0, (1+\epsilon)\mathcal{E}_0, \ldots, (1+\epsilon)^J\mathcal{E}_0$. We set J such that $(1+\epsilon)^J > \alpha/\epsilon$. The number of different algorithms run is $O(\frac{1}{\epsilon}\log\frac{\alpha}{\epsilon})$.

4. At some point the thresholded algorithm will declare "fail" for some $\mathcal{E}$. Then we know that $\mathcal{E}^* > \mathcal{E}$ for the (recursively) modified instance. We terminate the algorithm for all $\mathcal{E}' \leq \mathcal{E}$ and start running a thresholded algorithm for $(1+\epsilon)^J\mathcal{E}'$ *using the summarization of* $\mathcal{E}'$ *as the initial input.* Note that we always maintain the same number of copies of the thresholded algorithm but the error estimates change.

5. We repeat the above step until we see the end of input. We now declare the answer for the lowest estimate for which a thresholded algorithm is still running.

Figure 1: The StreamStrap Algorithm

Theorem 1. *If a thresholded approximation exists for a summarization problem whose error objective is a metric error then for any* $\epsilon \leq 1/10$ *the StreamStrap algorithm provides a* $\alpha/(1-3\epsilon)^2$ *approximation. The running time is the time to run* $O(\frac{1}{\epsilon}\log\frac{\alpha}{\epsilon})$ *copies of the thresholded algorithm plus* $O(\frac{1}{\epsilon}\log(\alpha\mathcal{E}^*M))$ *initializations.*

Proof: Consider the lowest value of the estimate $\mathcal{E}$ for which we have an algorithm running currently. Suppose that we had raised the estimate j times before settling on this estimate for this copy of the algorithm $\mathcal{A}(\mathcal{E})$. Let X_i denote the the prefix of the input *just before* the estimate was

raised for the i^{th} time over the history of $\mathcal{A}(\mathcal{E})$. Let H_i be the corresponding summary maintained for X_i. Denote the entire input as $X_j \circ Y$ and define Y_j as $X_j \setminus X_{j-1}$, that is, $X_j = X_{j-1} \circ Y_j$. Suppose the final summary is H. By the metric error property,

$$\mathcal{E}(X_j(H_j) \circ Y, H) - \mathcal{E}(X_j, H_j) \leq \mathcal{E}(X_j \circ Y, H) \qquad \leq \mathcal{E}(X_j(H_j) \circ Y, H) + \mathcal{E}(X_j, H_j) \tag{1}$$

Now $\mathcal{E}(X_j, H_j) \leq \mathcal{E}(X(H_{j-1}) \circ Y_j, H_j) + \mathcal{E}(X_{j-1}, H_{j-1})$. We observe that $\mathcal{E}(X(H_{j-1}) \circ Y_j, H_j)$ was run for an estimate $\epsilon\mathcal{E}/\alpha$, and thus $\mathcal{E}(X(H_{j-1}) \circ Y_j, H_j) \leq \alpha\frac{\epsilon\mathcal{E}}{\alpha}$ and further for all $i < j$, we have $\mathcal{E}(X_{i-1}(H_{i-1}) \circ Y_i, H_i) \leq \epsilon\mathcal{E}(X_i(H_i) \circ Y_{i+1}, H_{i+1})$. Using telescoping and observing $\mathcal{E}(X_1, H_1) = 0$, we get that

$$\mathcal{E}(X_j, H_j) \leq \sum_{1<i\leq j} \mathcal{E}(X_{i-1}(H_{i-1}) \circ Y_i, H_i) \leq \frac{\epsilon}{1-\epsilon}\mathcal{E} \tag{2}$$

Therefore the error of the algorithm is $\alpha\mathcal{E} + \mathcal{E}\epsilon/(1-\epsilon)$ which is less than $\mathcal{E}\alpha/(1-3\epsilon)$ since $\alpha \geq 1$. At the same time, if H^* is the optimum summary for $X_j \circ Y$, then by Equations 1 and 2,

$$\mathcal{E}(X_j(H_j) \circ Y, H^*) - \frac{\epsilon}{1-\epsilon}\mathcal{E} \leq \mathcal{E}(X_j \circ Y, H^*)$$

But since the algorithm failed for $\mathcal{E}/(1+\epsilon)$ we know that $\mathcal{E}(X_j(H_j) \circ Y, H^*)$ is at least $\mathcal{E}/(1+\epsilon)$. Therefore, $\mathcal{E}(X_j \circ Y, H^*)$ is at least $\mathcal{E}\left(\frac{1}{1+\epsilon} - \frac{\epsilon}{1-\epsilon}\right) \geq \mathcal{E}(1-3\epsilon)$ for the range of ϵ considered. Thus the approximation ratio follows. The number of initializations correspond to $\log_{1+\epsilon}(\alpha\mathcal{E}M)$ which is $O(\frac{1}{\epsilon}\log(\alpha\mathcal{E}^*M))$. □

2.3 Applications I: MinMax objectives

The important aspect of a minmax guarantee is that it applies to all the data points, and thus thresholded algorithms are very natural for this problem.

Theorem 2 (K-Center)**.** *We have a single pass $2+\epsilon$ approximation for K center using $O(\frac{K}{\epsilon}\log\frac{1}{\epsilon})$ space and $O(\frac{Kn}{\epsilon}\log\frac{1}{\epsilon} + \frac{K}{\epsilon}\log M\mathcal{E}^*)$ time when the points are input in an arbitrary order. Note that the radius of any cluster computed by the algorithm is additively within $\epsilon\mathcal{E}^*$ of the true radius of that cluster using that center.*

Proof: The main claim follows from applying Theorem 1 with $\epsilon' = \epsilon/20$ and $\alpha = 2$. The per cluster guarantee comes from the following facts: first, the error is a min-max objective and applies to every point in the input, secondly the $\sum_{i<j}\mathcal{E}(X(H_{i-1}) \circ Y_i, H_i)$ term evaluates to $\epsilon'\mathcal{E}$ which is at most $\epsilon\mathcal{E}^*$. □

Using the 3 approximation algorithm in [17] for K center with costs (where each node has a cost and we are restricted to sum of the cost of the centers to be less than C in addition to bound on K) we immediately get a $3+\epsilon$ approximation in $O(\frac{K}{\epsilon}\log\frac{1}{\epsilon})$ space. Achieving an approximation better than 3 is NP hard for this problem [8].

Theorem 3 (Maximum Error Histograms)**.** *We have a single pass $1+\epsilon$ streaming approximation for B bucket histogram construction using $O(\frac{B}{\epsilon}\log\frac{1}{\epsilon})$ space and $O(n+\frac{B}{\epsilon}(\log^2\frac{B}{\epsilon})\log M\mathcal{E}^*)$ time when the input $\ldots, x_i, \ldots$ is presented in increasing order of i. Again the error of any bucket found by the algorithm is additively within $\epsilon\mathcal{E}^*$ of the true error of that bucket.*

Proof: The space bound follows from the theorem. To see the time bound, consider, instead of running the thresholded algorithm on one input, to batch $t = O(\frac{B}{\epsilon}\log\frac{1}{\epsilon})$ inputs. On these t values we define a complete binary tree and recursively compute the max and min values of each interval defined by the tree in $O(t)$ time. Over the entire input, the time taken would be $\frac{n}{t}O(t)$ which is $O(n)$ for this part. Using the array we can compute the max and min of any interval in $O(\log t)$ time. Now every thresholded algorithm only needs to (repeatedly) find the maximal right extension of its current bucket (interval) such that $max - min \leq 2\mathcal{E}$. If this condition is violated in the t then call the search "terminating". Note that a non-terminating search can be decided in $O(1)$ time using the max, min of the entire t numbers. Observe that in that case the particular thresholded algorithm will continue to run. Thus over the entire life of the entire algorithm we would spend $O(n/t)$ times the number of thresholded algorithms being run (which is $O(\frac{1}{\epsilon}\log\frac{1}{\epsilon})$) for this step. But the product is $O(n/B)$ and is dominated by the $O(n)$ term.

For a terminating search, we can find the exact extension using another binary search in $O(\log t)$ time. Thus every bucket terminates using $O(\log^2 t)$ time. Thus over all the B buckets, for each algorithm the time is $O(B\log^2 t)$. The initiation time for each algorithm is $O(B)$ (the summary contains only B numbers which defines piecewise constant intervals). The number of thresholded algorithms tried is at most the number of algorithms being initiated. Since $\log t = O(\log\frac{B}{\epsilon})$, the result follows. □

2.4 Applications II: MinSum Objectives

The minsum variants of the summarization problems seek to minimize a sum over all the points. The well known VOPT histogram is the ℓ_2 variant of the maximum error objective and is a minsum (of squares) variant. The K-median problem is the minsum objective corresponding to the K-center problem.

We now focus on the VOPT histogram which seeks an H which minimizes $\|X-H\|_2^2$. The square root of the VOPT error is the ℓ_2 metric and satisfies the metric error property. Further a $(1+\epsilon)$ approximation of the square root gives a $(1+\epsilon)^2 < 1+3\epsilon$ approximation for small ϵ. A summary which gives a $(1+\epsilon)$ approximation of the VOPT error also provides a $(1+\epsilon)$ approximation of the square root.

We note that the algorithm AHIST-B as detailed in Section 3.5 in [13] is a streaming $(1+\epsilon)$ approximation for the VOPT error. We will run *two* such algorithm as our thresholded algorithm assuming that the error is between $[\mathcal{E}, B\mathcal{E}/\epsilon)$ and $[B\mathcal{E}/\epsilon, B^2\mathcal{E}/\epsilon^2)$. Once the first fails, we use the summary of that to initiate a thresholded algorithm for $[B^2\mathcal{E}/\epsilon^2, B^3\mathcal{E}/\epsilon^3)$. This geometric factor of B/ϵ suffices for the telescoping sum in proof of Theorem 1. In the algorithms studied in [13], there was no upper bound to the error, but here we have an upper bound in the thresholding algorithm which limits the parameter τ as descried in the AHIST-B algorithm[2] to be $B/\epsilon + \log_{1+\epsilon/B}\frac{B}{\epsilon} =$

[2]These details are available in [13]. The analysis is however sharper than that of [13] since we will separate the terminating and non-terminating searches as in the analysis of maximum error which is novel in this paper.

$O(\frac{B}{\epsilon}\log\frac{B}{\epsilon})$. This algorithm simultaneously tries to maintain approximate j-bucket histograms for $j < B$. For each j it finds τ "breakpoints" which determines the error of a bucket for the $j+1$-bucket histogram.

We would set $t = O(\frac{B^2}{\epsilon}\log\frac{B}{\epsilon})$ which is the target space bound, and corresponds to the number of items read in a batch. Again using $O(t)$ space we compute the running sum and sum of squares for the intervals corresponding to the complete binary tree.

An easy analysis of the cost of terminating searches is $O(\log t)$ evaluations of the error in a bucket is required to add a bucket. Each evaluation requires $O(\tau)$ time. So over the $B-1$ values of j the time taken is $O(B\tau(\tau+\log t)\log t) = O(\frac{B^3}{\epsilon^2}\log^3\frac{B}{\epsilon})$ for each thresholded algorithm. We have to run at most $\log_{\frac{B}{\epsilon}} M\mathcal{E}^*$ such algorithms. This gives a running time of $O(\frac{B^3}{\epsilon^2}(\log^2\frac{B}{\epsilon})\log M\mathcal{E}^*)$.

The cost of non-terminating searches is $O(1)$ time (after the sum and sum of squares arrays are set up) for each last bucket for each j– which translates to $O(B\frac{n}{t}) = O(\epsilon n/B)$ and is again dominated by the $O(n)$ time to create the n/t arrays of sum and sum of squares. Thus,

Theorem 4 (VOPT error)**.** *We can compute a $1+\epsilon$ approximation to the best B-bucket histogram for VOPT error using $O(\frac{B^2}{\epsilon}\log\frac{B}{\epsilon})$ space and $O(n + \frac{B^3}{\epsilon^2}(\log^2\frac{B}{\epsilon})\log M\mathcal{E}^*)$ time when the input $\ldots, x_i, \ldots$ is presented in increasing order of i.*

We now consider the K-median problem. Recall that the goal in this problem, given n points $P = p_1, \ldots, p_n$, is to identify K medians $p_{i_1}, \ldots, p_{i_k}$ such that $\sum_{x\in P}\min_{j\leq K} d(x, p_{i_j})$ is minimized.

The first $O(1)$ approximation for the K-median problem for data streams using sublinear space was given by Guha et. al [14]. Based on Meyerson's [23] online facility location algorithm, Charikar et. al [7] gave a randomized $O(1)$ approximation using $O(K\log^2 n)$ space which succeeded with probability $1 - 1/n^{\Omega(1)}$. However the algorithm in [7] borrows the "doubling" argument from [6], and the approximation ratio is $\beta + 2c(1+\beta)$ where β is large. In fact β satisfies $\beta\gamma \geq 4 + 16\beta + 17\gamma$ and $\beta \geq 2c(1+\gamma)+\gamma$ where c is the best approximation algorithm for the K-median problem and γ can be chosen to satisfy the two conditions. Based on the result of Arya et. al [2] $c = 3+\epsilon$. Inspection shows that $\gamma > 16$ and minimizing β over the choices of γ gives $\beta \geq 130$. 130.

Using the framework here we were able to improve the 8 approximation in [6] to a $2+\epsilon$ approximation. Applying the same ideas to the K-median problem reduces the parameter β to $4+\epsilon$, as we show below. We note that this result is immediate if we lose a further factor of $\frac{1}{\epsilon}\log(M\mathcal{E}^*)$ in space. The goal is to avoid dependence on $M, \mathcal{E}^*$ (unfortunately the algorithm will depend on $\log^2 n$).

First, we observe that the K-median objective function satisfies the metric error property. We note that based on Lemma 1 in [7], Markov inequality and the union bound it follows that: claim:

Lemma 1. *There exists a simple randomized algorithm such that with probability at least ϵ, we produce a r-median solution whose objective is $(1+2\epsilon)((4\mathcal{E}^*+L)$ where $r \leq \frac{k}{\epsilon}(1+\log n)(1+4\mathcal{E}^*/L)$ and $\mathcal{E}^*$ is the value of the best k-median solution. This algorithm uses $O(r)$ space.*

Suppose we run $O(\frac{1}{\epsilon}\log n)$ copies of the above procedure for $L = \epsilon\mathcal{E}$ for an estimate $\mathcal{E}$. An individual copy fails if the number of medians exceed r or if the solution exceeds $4(1+\epsilon)\mathcal{E}$. Then if $\mathcal{E} \leq \mathcal{E}^*/(1+\epsilon)$ then the probability of declaring failure is at most $1/n^{\Omega(1)}$. We can now run the StreamStrap algorithm (which will run $O(\frac{1}{\epsilon}\log\frac{1}{\epsilon})$ copies of this) and we achieve:

Lemma 2. *There exists a randomized algorithm such that the expected value of a r-median solution produced by the algorithm is $4(1+\epsilon)\mathcal{E}^*$ where $r \leq 4\frac{k}{\epsilon^2}\log n$ and $\mathcal{E}^*$ is the value of the best k-median solution. This algorithm uses $O(\frac{k}{\epsilon^3}(\log^2 n)\log\frac{1}{\epsilon})$ space and succeeds with probability $1 - 1/n^{\Omega(1)}$.*

In essence, we show that we can achieve a β arbitrarily close to 4, similar to the statement we showed for the K-center problem. Note that the space bound matches [7] for any constant ϵ, but the approximation factor is greatly improved, which was our goal.

Theorem 5 (K-median)**.** *There exists a randomized* $34+\epsilon$ *approximation for the* K*-median problem in the oracle distance model in a data stream setting using* $O(\frac{k}{\epsilon^3}(\log^2 n)\log\frac{1}{\epsilon})$ *space which succeeds with probability* $1 - 1/n^{\Omega(1)}$*.*

3 Lower bound for Maximum Error Histograms

We begin with the definition of the Indexing problem in communication complexity. Alice has a string $\sigma \in \{0,1\}^n$ and Bob has an index $1 \leq j \leq n$. The goal is for Alice to send a single message to Bob such that Bob can compute the j^{th} bit σ_j. It is known that this requires Alice to send $\Omega(n)$ bits [21]. We would reduce the Indexing problem to constructing a histogram – Alice would interpret her string as some numbers and start a histogram construction algorithm. At the end of her input she will send her memory state to Bob and Bob will continue the computation. A good approximation to the histogram problem will solve the indexing problem. Thus the memory state sent by Alice must be $\Omega(n)$ bits, which gives us a lower bound of the space complexity of any streaming algorithm. Since the lower bound of indexing holds for randomized algorithms, the same proofs will translate to a lower bound for randomized algorithms. We start with a simple reduction.

Theorem 6. *Any* $(1+\epsilon)$ *approximation for* $B = 2$ *bucket histogram for maximum error, even when the input* $\ldots, x'_i, \ldots$ *is presented in increasing order of* i'*, must use* $\Omega(1/\epsilon)$ *bits of space.*

Proof: Suppose we have a histogram algorithm which requires s space. Alice starts the histogram algorithm with the input 0. Then starting from $i = 1$ if $\sigma_i = 1$ she adds the number $n + i$ to the stream. If $\sigma_i = 0$ she does not add anything. In both cases she proceeds to the next i'. Note that the i and i' are different – then $x_{i'}$ input corresponds to the i'-th bit which has value 1. At the end of $i = n$ she sends the contents of her memory to Bob. Bob adds the number $2(n + j)$.

If $\sigma_j = 1$ then the three numbers $0, n + j, 2n + 2j$ have to be covered by two buckets and the error is at least $\frac{1}{2}(n + j)$. If however $\sigma_j = 0$ then the error is no more than $\frac{1}{2}(n + j - 1)$ which corresponds to covering all numbers less or equal $n+j-1$ and all numbers greater or equal $n+j+1$ by the two buckets. Suppose $\epsilon = 1/(4n)$. Then a $(1+\epsilon)$ approximation separates the two cases since $j \leq n$,

$$(1+\frac{1}{4n})\frac{1}{2}(n+j-1) \leq \frac{1}{2}(n+j) - \frac{1}{2}(1+\frac{1}{4n}) + \frac{1}{4n}\frac{1}{2}2n < \frac{1}{2}(n+j)$$

Thus a $(1+\epsilon)$ approximation will reveal σ_j and therefore s must be at least $\Omega(n) = \Omega(\frac{1}{\epsilon})$. □

The above leaves open the possibility that there is an algorithm possible with space $O(B + \frac{1}{\epsilon})$. This is ruled out by the next lower bound. However we use the natural requirement of summarization that each bucket be approximated to additive ϵ times the optimum error. All upper bound algorithms satisfy this criterion.

Theorem 7. *For all $\epsilon \leq 1/(40B)$, any $(1+\epsilon)$ approximation for B bucket histogram for maximum error, which also approximates the error of each bucket within additive ϵ times the optimum error must use $\Omega(\frac{B}{\epsilon \log \frac{B}{\epsilon}})$ bits of space, even when the input $\ldots, x'_i, \ldots$ is presented in increasing order of i'.*

Proof: Let t, r be integers such that $t > 2r$. Let $S_i = \{a(t+i) | a(t+i) < 2rt$ and a is a positive integer$\}$. Observe that for $i, i' < t$, such that $t+i, t+i'$ are coprime (do not share a common factor), the sets $S_i, S_{i'}$ are disjoint and $2r > |S_i| \geq r$. Now, using the prime number theorem, there are $\Theta(t/\log t)$ primes between t and $2t$ for large t. Thus $S = \cup_{0 \leq i < t} S_i$ is of size $n = \Omega(rt/\log t)$. Let the numbers in S be denoted by $T_1, T_2, \ldots, T_n$.

Again we reduce indexing to the histogram construction. Assume we have a good histogram algorithm. Alice with her string σ, now starts with 1/4 and adds $T_i - \frac{1}{2}, T_i + \frac{1}{2}$ if $\sigma_i = 0$ and adds $T_i - \frac{1}{4}, T_i + \frac{1}{4}$. She sends the memory state to Bob. Bob computes an i_0 such that T_j belongs to S_{i_0}. Bob can also compute T_n. For each element u of the form $a(t+i_0)$ such that $T_n < u < 2r(t+i_0)$ he adds $u - \frac{1}{4}, u + \frac{1}{4}$. He finally adds $2r(t+i_0) - 1/4$. The input is interpreted as a sequence $\ldots, x_{i'}, \ldots$ in increasing order of i'.

Set $B = 2r$. Then there exists a $2r$ bucket histogram which uses the buckets $[1/4, t + i_0 - 1/4], [t + i_0 + \frac{1}{4}, 2(t+i_0) - \frac{1}{4}], \ldots, [(2r-1)(t+i_0) + \frac{1}{4}, 2r(t+i_0) - \frac{1}{4}]$. We are willfully ignoring the 0/1 settings for this case. The error therefore is at most $\frac{1}{2}(t + i_0 - \frac{1}{2})$. Any other histogram either contains a bucket spanning two multiple of $t + i_0$ or contains $t + i_0$ in the interval corresponding to the first bucket or contains $(2r-1)(t+i_0)$ in the interval corresponding to the last bucket. Thus the the error will be at least $\frac{1}{2}(t + i_0 - \frac{1}{4})$.

Now if we are guaranteed a $1+\epsilon$ approximation with $\epsilon = 1/(20t)$ (which ensures the range of ϵ in the theorem statement) then

$$(1+\epsilon)\frac{1}{2}(t + i_0 - \frac{1}{2}) < \frac{1}{2}(t + i_0 - \frac{1}{4}) + \frac{1}{2}\epsilon(t + i_0) - \frac{1}{8} < \frac{1}{2}(t + i_0 - \frac{1}{4}) + \frac{1}{2}\frac{1}{20t}2t - \frac{1}{8} < \frac{1}{2}(t + i_0 - \frac{1}{4})$$

Therefore any $(1+\epsilon)$ forces the bucket boundaries to begin or end around multiples of $t + i_0$.

We each bucket has to be approximated well. Therefore if we use $T_j - \frac{1}{4}$ instead of $T_j - \frac{1}{2}$ or $T_j + \frac{1}{4}$ instead of $T_j + \frac{1}{2}$ then the error would be at least $\frac{1}{2}\frac{1}{4} = \frac{1}{8}$ But the allowed error is $\epsilon\frac{1}{2}(t + i_0 - \frac{1}{2}) < \frac{1}{20t}\frac{1}{2}2t < \frac{1}{8}$. Therefore the approximation of the particular bucket which contains the endpoint corresponding to j will reveal σ_j, and solve the indexing problem. Therefore the space requirement is $\Omega(n) = \Omega(\frac{B}{\epsilon \log \frac{B}{\epsilon}})$. □

4 Lower Bounds for K-center in the Oracle Distance Model

Let us recall the oracle distance model we are considering in this paper. There is a distance evaluation function or an "oracle", which when provided with two point p, q return only the distance $d(p, q)$. The oracle model has to store the individual points in their entirety, to be able to invoke the oracle. Thus the measure of space used by any algorithm will be the number of points stored. *The fundamental assumption in this model is that the algorithm cannot "create" any point which is not in the input.* This separates arbitrary metric spaces from geometric spaces, and we will see a direct effect of this soon. In the remainder of the section we will provide lower bounds for single pass deterministic algorithms.

The roadmap: We will first provide the algorithm with a lot of input points and the algorithm will be forced to forget a majority of these points. Based on these forgotten points we will adversarially (this is where we use the fact that the algorithm is deterministic) choose a further set of points and force the algorithm to remember all these new points.

Theorem 8. *A single pass deterministic streaming algorithm in the oracle distance model for $\epsilon = \Theta(K)$ that simultaneously provides a $2+\epsilon$ approximation for the K-center problem as well as a bound on the radius of each cluster within an additive ϵ times the optimum radius must store $\Omega(K^2)$ points for some input.*

Proof: Let $K = t + r$, $\epsilon = 1/(8t)$ and $t = 8r$. Consider a set of points $P = \{p_{uv} | 1 \leq u \leq t \text{ and } 1 \leq v \leq r\}$.

We first provide the points P_0 corresponding to $1 \leq u \leq t/2$. The distance between p_{uv}, p_{gh} is defined as (assume wlog $u \geq g$) : If $u = g$ (and $h \neq v$) the distance is $\frac{3}{2}$. Otherwise if $h = v$ then the distance is $\frac{3}{2} + \frac{u}{2t}$. Otherwise the distance is $\frac{9}{4}$. We can verify that this is a metric.

Suppose the algorithm remembers a set T_1 of points, $T_1 \leq tr/100$. Define a column to be "sparse" if at least $2r$ points from this column has been forgotten. A column is "dense" otherwise. Now $t - 2r \geq \frac{3t}{4}$. The number of dense columns is therefore at most $r/75$. Therefore $74r/75$ columns are sparse. Let this set of columns be S.

We now provide the points $P_1 = \{p_{uv} | t/2 < u \leq t \text{ and } v \in S\}$. *Note that we do not have to specify the distances between the points in P_1 and $P_0 \setminus T_1$.* Otherwise for $p_{gh}, p_{uv} \in T_1 \cup P_1$ the distances are given by the same set of conditions that determine the distance between P_0 above. At the end of this phase the algorithm remembers a point set T_2.

We now choose a $j : t/2 < j \leq t$. We add r special points $\{a_i\}$ such that distance from a_i to any p_{uv} is $\frac{9}{4}$ if $i \neq v$. If $v \notin S$ the distance of a_v to all p_{uv} (note $u \leq t/2$) is $\frac{3}{4} + \frac{j}{4t}$. If $v \in S$ then the distance of a_v, p_{uv} is: if $u \leq j$ then it is $\frac{3}{4} + \frac{j}{4t}$ else it is $\frac{3}{2} + \frac{u}{2t}$.

We next introduce $t - j$ special points $\{b_g | g > j\}$ such that distance from b_g to any p_{uv} where $u \neq g$ is $\frac{9}{4}$. If $g = u$ (then $v \in S$) then it is $\frac{3}{4}$. Finally we introduce j "faraway" points which are at distance 10 from every other point.

Supposing a $p_{gh} \in P_0 \setminus T_1$ behaved exactly the same as a $p_{uv} \in T_1$, in its distance to a_i, and in particular a_h. Then there is a clustering with centers $\{a_i\} \cup \{b_g\}$ and the faraway points, with radius $R = \frac{3}{4} + \frac{j}{4t}$. *The algorithm does not know this, but cannot rule this possibility out – hence it must provide a solution with radius $(2+\epsilon)R < \frac{9}{4}$.* But the distance between a $p_{gh} \in P_0 \setminus T_1$ and a_h can also be $\frac{9}{4}$ (without conflicting the metric property since $\frac{9}{4} \leq \frac{3}{4} + (\cdots) + \frac{3}{2} + (\cdots)$ for the shortest path from a_h). Therefore none of the a_h can be used as centers in the solution of the algorithm.

Further the points in row $g \geq j + 1$ are at a distance $\frac{3}{2} + \frac{j+1}{4t}$ from points in any other row – and this is larger than $(2+\epsilon)(\frac{3}{4} + \frac{j}{4t})$. Thus every such row must have a different center (which can be at b_g). Also the faraway points must have a center by themselves. This leaves exactly r centers. Consider a_v, a_h; there is no point which is within distance $(2+\epsilon)R$ from both. Therefore each a_v must be covered by a separate center which is either a_v or some p_{uv}. But we have already shown that none of the a_v can be used as a center by the algorithm. Therefore the algorithm must use r centers corresponding to some p_{uv}.

Let us now focus on $v \in S$. *The algorithm cannot use a center p_{uv} which is in $P_0 \setminus T_1$.* If it did, it would not account for the possibility that the distance to a point forgotten in this column is $\frac{9}{4}$ (shortest path from p_{uv} through any other p_{gv} will be more than $\frac{3}{2} + (\cdots) + \frac{3}{2} + (\cdots)$). A

point in $P_0 \setminus T_1$ may be covered by a center in the same row but – since a sparse column has at least $2r$ forgotten points and we only have r centers which are free, at most r of these points can be covered by a center in the same row. Therefore for all $v \in S$ the center must be a point p_{uv} with $u \leq t/2$. *But then the algorithm must remember p_{jv} because p_{jv} is the farthest point from p_{uv} in this cluster and p_{jv} cannot be covered in any other way and the next farthest point is at least ϵR distance away.* That means $p_{jv} \in T_2$.

We now make the final observation that in the above, j was fixed after T_2 was fixed. Therefore unless T_2 contained all p_{jv} for $t \geq j > t/2, v \in S$ we can always find a j which breaks the clustering guarantee of the algorithm. Thus we arrive at a contradiction that we stored less than $tr/100$ points in T_1. This shows that $\Omega(K^2)$ points are needed. □

The above shows that the $O(\frac{K}{\epsilon} \log \frac{1}{\epsilon})$ is almost the best possible space *general* bound which holds for *all* K, ϵ. We believe that Theorem 8 generalizes to all ϵ, K and leave that question open. Another important open question is the status of randomized algorithms, namely, is it possible to have a 2 approximation for the K-center problem using $o(n)$ space? Although we know that it is NP-Hard to approximate the K-center problem better than factor 2, we can show a stronger result in the space bounded scenario.

Theorem 9 (Randomized K-Center)**.** *Any randomized algorithm that provides an approximation ratio better than* 2 *for the* 1*-center problem in the oracle distance model must use $\Omega(n)$ space.*

Proof: The Indexing problem (see previous section) can be reduced to this problem. Given a $\sigma \in \{0,1\}^n$, if $\sigma_i = 1$ Alice adds a point p_i to the stream otherwise she does nothing. The oracle answers the distance between any two pair of points to be 1. She runs the K center algorithm and sends the content of the memory to Bob. Bob adds a point p'_j which is at distance 2 from all p_i where $i \neq j$ and is at a distance from 1 from p_j. If $\sigma_j = 1$ then there exists a clustering of radius 1 choosing p_j as a center. If $\sigma_j = 0$ then the minimum radius is 2. Therefore an algorithm than distinguishes these cases must use $\Omega(n)$ bits of space. □

Acknowledgments

We are grateful to Piotr Indyk, Samir Khuller and Andrew McGregor for a number of stimulating discussions.

References

[1] S. Acharya, P. Gibbons, V. Poosala, and S. Ramaswamy. The Aqua Approximate Query Answering System. *Proc. of ACM SIGMOD*, pages 574–576, 1999.

[2] V. Arya, N. Garg, R. Khandekar, and V. Pandit. Local search heuristics for k–median and facility location problems. In *Proc. STOC*, 2001 (to appear).

[3] M. Bertolotto and M. J. Egenhofer. Progressive vector transmission. Proc. of the 7th ACM symposium on Advances in Geographical Information Systems, pages 152–157, 1999.

[4] C. Buragohain, N. Shrivastava, and S. Suri. Space efficient streaming algorithms for the maximum error histogram. *Proc of. ICDE*, pages 1026–1035, 2007.

[5] K. Chakrabarti, E. J. Keogh, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM TODS*, 27(2):188–228, 2002.

[6] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. In *Proc. STOC*, pages 626–635, 1997.

[7] M. Charikar, L. O'Callaghan, and R. Panigrahy. Better streaming algorithms for clustering problems. *Proc. of STOC*, pages 30–39, 2003.

[8] J. Chuzhoy, S. Guha, E. Halperin, S. Khanna, G. Kortsarz, R. Krauthgamer, and J. Naor. Asymmetric k-center is $\log^* n$-hard to approximate. *J. ACM*, 52(4):538–551, 2005.

[9] M. N. Garofalakis and P. B. Gibbons. Wavelet synopses with error guarantees. *Proc. of ACM SIGMOD*, pages 476–487, 2002.

[10] S. Guha. Space efficiency in synopsis construction problems. *Proc. of VLDB Conference*, pages 409–420, 2005.

[11] S. Guha and B. Harb. Approximation algorithms for wavelet transform coding of data streams. *IEEE Trans. of Info. Theory*, 2008.

[12] S. Guha, N. Koudas, and K. Shim. Data-streams and histograms. In *Proc. STOC*, pages 471–475, 2001.

[13] S. Guha, N. Koudas, and K. Shim. Approximation and streaming algorithms for histogram construction problems. *ACM TODS*, 31(1), 2006.

[14] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams: Theory and practice. *IEEE Trans. Knowl. Data Eng.*, 15(3):515–528, 2003.

[15] S. Guha and K. Shim. A note on linear time algorithms for the maximum error problem. *IEEE Trans. Knowl. Data Eng.*, 19(7):993–997, 2007.

[16] S. Guha, K. Shim, and J. Woo. REHIST: Relative error histogram construction algorithms. *Proc. VLDB Conference*, pages 300–311, 2004.

[17] D. S. Hochbaum and D. B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *J. ACM*, 33(3):533–550, 1986.

[18] Y. E. Ioannidis. Universality of serial histograms. *Proc. of the VLDB Conference*, pages 256–267, 1993.

[19] H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. C. Sevcik, and T. Suel. Optimal Histograms with Quality Guarantees. *Proceedings of VLDB*, pages 275–286, Aug. 1998.

[20] P. Karras, D. Sacharidis, and N. Mamoulis. Exploiting duality in summarization with deterministic guarantees. *Proc. of KDD*, 2007.

[21] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[22] J. Matousek. *Lectures on Discrete Geometry*. Springer, GTM series, 2002.

[23] A. Meyerson. Online facility location. *Proc. FOCS*, 2001.