Optimal Quantile Approximation in Streams

Zohar Karnin
Yahoo Research
New York, NY
zkarnin@yahoo-inc.com

Kevin Lang
Yahoo Research
Sunnyvale, CA
langk@yahoo-inc.com

Edo Liberty
Amazon
New York, NY
libertye@amazon.com

***Abstract*—This paper resolves one of the longest standing basic problems in the streaming computational model. Namely, optimal construction of quantile sketches. An ε approximate quantile sketch receives a stream of items $x_1, \ldots, x_n$ and allows one to approximate the rank of any query item up to additive error εn with probability at least $1 - \delta$. The rank of a query x is the number of stream items such that $x_i \leq x$. The minimal sketch size required for this task is trivially at least $1/\varepsilon$. Felber and Ostrovsky obtain a $O((1/\varepsilon)\log(1/\varepsilon))$ space sketch for a fixed δ. Without restrictions on the nature of the stream or the ratio between ε and n, no better upper or lower bounds were known to date. This paper obtains an $O((1/\varepsilon)\log\log(1/\delta))$ space sketch and a matching lower bound. This resolves the open problem and proves a qualitative gap between randomized and deterministic quantile sketching for which an $\Omega((1/\varepsilon)\log(1/\varepsilon))$ lower bound is known. One of our contributions is a novel representation and modification of the widely used merge-and-reduce construction. This modification allows for an analysis which is both tight and extremely simple. The same technique was reported, in private communications, to be useful for improving other sketching objectives and geometric coreset constructions.**

***Keywords*-quantiles; streaming quantiles; streaming median**

I. Introduction

Given a set of items $x_1, \ldots, x_n$, the quantile of an item x is the fraction of items in the stream such that $x_i \leq x$. It is convenient to define the rank of x, $R(x)$, as the *number* of items such that $x_i \leq x$. An additive error εn for $R(x)$ is an ε approximation of its rank. The literature distinguishes between several different definitions of this problem. In this manuscript we distinguish between the single quantile approximation problem and the all quantiles approximation problem.

Definition 1. *The single quantile approximation problem: Given $x_1, \ldots, x_n$ in a streaming fashion in arbitrary order, construct a data structure for computing $\tilde{R}(x)$. By the end of the stream, receive a single element x and compute $\tilde{R}(x)$ such that $|\tilde{R}(x) - R(x)| \leq \varepsilon n$ with probability $1 - \delta$.*

There are variations of this problem in which, the algorithm is not given x (as a query) but rather a rank r. It should be able to provide an element x_i from the stream such that $|R(x_i) - r| \leq \varepsilon n$. There are also variants that make the value r, or r/n known to the algorithm in advance. For example, one could a priori choose to search for an approximate median. There are easy reductions between the different variants that maintain the failure probability and error up to a constant. The solution we propose solves all the above variants.

Definition 2. *The all quantiles approximation problem: Given $x_1, \ldots, x_n$ in a streaming fashion in arbitrary order, construct a data structure for computing $\tilde{R}(x)$. By the end of the stream, with probability $1 - \delta$, for all values of x simultaneously it should hold that $|\tilde{R}(x) - R(x)| \leq \varepsilon n$.*

Observe that approximating a set of $O(1/\varepsilon)$ single queries well suffices for solving the all quantiles approximation problem. Therefore, solving the single quantiles approximation problem with failure probability at most $\varepsilon\delta$ constitutes a valid solution for the all quantiles approximation problem simply by invoking the union bound. For this reason, in this manuscript we deal almost solely with the single quantile approximation problem.

A. Related Work

Two recent surveys [1][2] on this problem give ample motivation and explain the state of the art in terms of algorithms and theory in a very accessible way.[1] In what follows, we shortly review some of the prior work that is most relevant in the context of this manuscript. For readability, space complexities of randomized algorithms apply for a constant success probability unless otherwise stated.

Manku, Rajagopalan and Lindsay [3] built on the work of Munro and Paterson [4] and gave a randomized solution which uses at most $O((1/\varepsilon)\log^2(n\varepsilon))$ space. A simple deterministic version of their algorithm achieves the same bounds. This was pointed out, for example, by [1]. We refer to their algorithm as MRL. Greenwald and Khanna [5] created an intricate deterministic algorithm that requires $O((1/\varepsilon)\log(n\varepsilon))$ space. This is the best known deterministic algorithm for this problem. We refer to their algorithm as GK.

Allowing randomness enables sampling. A uniform sample of size $n' = O(\log(1/\varepsilon)/\varepsilon^2)$ from the stream suffices

[1]Manuscript [2] was authored in 2007 as a book chapter. It does not contain recent results but is an excellent survey nonetheless.

to produce an all quantiles sketch. Feeding the sampled elements into a GK sketch yields an $O((1/\varepsilon)\log(1/\varepsilon))$ solution. However, to produce such samples, one must know n (at least approximately) in advance. This observation was already made by Manku et al. [3]. Since n is not known in advance it is not a trivial task to combine sampling with GK sketches. Recently, Felber and Ostrovsky [6] managed to do exactly that. They achieved space complexity of $O((1/\varepsilon)\log(1/\varepsilon))$ by using sampling and several GK sketches in conjunction in a non trivial way. To the best of our knowledge, this is the best known space complexity result to date.

Space complexity: Throughout this manuscript we define the size of a sketch S as $|S|$ if its memory footprint is at most $|S|$ times the size of storing a single stream item and a single word of memory.

Mergeability: An important property of sketches is called mergeability [7]. Informally, this property allows one to sketch different sections of the stream independently and then combine the resulting sketches. The combined sketch should be as accurate as a single sketch one would have computed had the entire stream been consumed by a single sketcher. This is formally stated in Definition 3.

Definition 3. *Let S denote a sketching algorithm mapping a stream N to a summary $S(N)$, and denote by $\mathrm{err}(S, n)$ the worst-case error associated with the summary of S on a stream of length n. Similarly, define $\mathrm{size}(S, n)$ to be maximal size of the the sketch.*

Let M be a merge algorithm defined by $M(N) = M(S(N_1), S(N_2))$ where N_1 and N_2 constitute a partition of N. Define $\mathrm{err}(M, S, n)$ and $\mathrm{size}(M, S, n)$ similarly to the above. A sketching algorithm S is mergeable if there exists a merge algorithm M such that

$$\mathrm{err}(M, S, n) \le \mathrm{err}(S, n) \quad \textit{and} \quad \mathrm{size}(M, S, n) \le \mathrm{size}(S, n)$$

and furthermore, this sketch has the same guarantees of $S(N)$ w.r.t. future merge operations.

This property is extremely important in practice since large datasets are often distributed across many machines. Agarwal et al [7] conjecture that the GK sketch is not mergeable. They describe a mergeable sketch of space complexity $(1/\varepsilon)\log^{3/2}(1/\varepsilon)$. It is worth noting that this results predates that of [6].

The Comparison Model: Different sketching algorithms perform different kinds of operations on the elements in the stream. The most restricted model is the comparison model. In this model, there is a strong order imposed on the elements and the algorithm can only compare two items to decide which is larger. This is the case, for example, for lexicographic ordering of strings. All the cited works above operate in this model. Another model assumes the total size of the universe is bounded by $|U|$. An $O((1/\varepsilon)\log(|U|))$ space algorithm was suggested by [8] in that model.

Lower bounds: For any algorithm, there exists a (trivial) space lower bound of $\Omega(1/\varepsilon)$. Hung and Ting [9] showed that any *deterministic* comparison based algorithm for the single quantile approximation problem must store $\Omega((1/\varepsilon)\log(1/\varepsilon))$ items. Felber and Ostrovsky [6] suggest, as an open problem, that the $\Omega((1/\varepsilon)\log(1/\varepsilon))$ lower bound could potentially hold for randomized algorithms as well. Prior to this work, it was very reasonable to believe this conjecture is true.

Randomly Ordered Streams: If the stream is guarantied to be presented in random order, obtaining an $O((1/\varepsilon)\log(1/\varepsilon))$ solution is easy. Namely, save the first $O((1/\varepsilon)\log(1/\varepsilon))$ items and, from that point on, only count how many items fall between two consecutive samples. Due to coupon collector, at least one sample will fall in each stretch of εn items which makes the solution trivially correct. Guha and McGregor [10] describe an algorithm with success probability $1 - \delta$ which achieves $\varepsilon = O(\mathrm{polylog}(n/\delta)/\sqrt{n})$. For any single quantile it requires $O(1)$ memory and therefore $O(1/\varepsilon)$ space for all quantiles. If the items are numbers, for example, one could also compute averages or perform gradient descent like algorithms such as [11], [12].

B. Main Contribution

We begin by re-explaining the work of [7] and the deterministic version of [3] from a slightly different view point. The basic building block for these algorithms is a single compactor. A compactor can store k items all with the same weight w. It can also compact its k elements into $k/2$ elements of weight $2w$ as follows. First, the items are sorted. Then, either the even or the odd elements in the sequence are chosen. The unchosen items are discarded. The weight of the chosen elements is doubled (set to $2w$). Consider a single query x. Its rank estimation before and after the compaction defers by at most w regardless of k. This is illustrated in Figure 1.

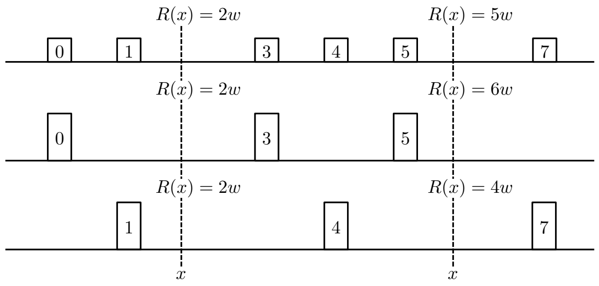


Figure 1. An illustration of a single compactor with 6 items performing a single compaction operation. The rank of a query remains unchanged if its rank within the compactor is even. If it is odd, its rank is increased or decreased by w with equal probability by the compaction operation.

This already gives a deterministic algorithm. Assume we use such a compactor. When it outputs items, we feed them into another compactor and so on. Since each compactor halves the number of items in the sequence, there could be at most $H \le \lceil \log(n/k) \rceil$ compactors chained together. Let h denote the height of a compactor where the last one created has height $h = H$ and the first one has height $h = 1$. Let $w_h = 2^{h-1}$ be the weight of items compactor h gets. Then, the number of compact operations it performs is at most $m_h = n/kw_h$. Summing up all the errors in the system $\sum_{h=1}^{H} m_h w_h = \sum_{h=1}^{H} n/k = Hn/k \le n \log(n/k)/k$. Since we have H compactors, the space usage is $kH \le k \log(n/k)$. Setting $k = O((1/\varepsilon) \log(\varepsilon n))$ yields error of εn with space $O((1/\varepsilon) \log^2(\varepsilon n))$.

One conceptual contribution of Agarwal et al. [7] is to have each compactor delete the odd or even items with equal probability. This has the benefit that the expected error is zero. It also lets one use standard concentration results to bound the maximal error. This eliminates one log factor from the worst case analysis. However, the dependence on the failure probability adds a factor of $\sqrt{\log(1/\delta)}$. In the all quantiles problem this translates into an additional $\sqrt{\log(1/\varepsilon)}$ factor for constant failure probability. Intuitively Agarwal et al. [7] also show that when $n \ge \mathrm{poly}(1/\varepsilon)$ one can sample items from the stream before feeding them to the sketch. This gives total space usage of $O\left((1/\varepsilon) \log(1/\varepsilon)\sqrt{\log(1/\delta)}\right)$ for the single quantile problem and $O\left((1/\varepsilon) \log(1/\varepsilon)\sqrt{\log(1/\varepsilon\delta)}\right)$ for the all quantiles problem.

The first improvement we provide to the algorithms above is to use different compactor capacities in different heights, denoted by k_h. We show that k_h can, for example, decrease exponentially $k_h \approx k_H(2/3)^{H-h}$. That is, compactors in lower levels in the hierarchy (smaller item weights) can operate with significantly less capacity. Surprisingly enough, this turns out to not effect the asymptotic statistical behavior of the error at all. Moreover, the space complexity is clearly improved.

The capacity of any functioning compactor must be at least 2. This could contribute $O(H) = O(\log(n/k))$ to the space complexity. To remove this dependence, we notice that a sequence of H'' compactors with capacity 2 essentially perform sampling. Out of every $2^{H''}$ elements they select one at random and output that element with weight $2^{H''}$. This is clearly very efficiently computable and does not truly require memory complexity of $O(H'')$ but rather of $O(1)$. The total capacity of all compactors whose capacity is more than 2 is bounded by $\sum_{h=H''+1}^{H} k(2/3)^{H-h} \le 3k$. This yields a total space complexity of $O(k)$. Setting $k = O((1/\varepsilon)\sqrt{\log(1/\delta)})$ gives an algorithm with space complexity $O((1/\varepsilon)\sqrt{\log(1/\delta)})$ for the single quantile problem and $O((1/\varepsilon)\sqrt{\log(1/\delta\varepsilon)})$ for the all quantiles problem, which constitutes our first result. Interestingly, our algorithm can be thought of a smooth interpolation between carful compaction of heavy items and efficient sampling for light items. We believe this idea would be potentially useful in other geometric coreset construction problems.

The next improvement comes from special handling of the top $\log\log(1/\delta)$ compactors. The number of compaction operations (and therefore random bits) in those levels is $O(\log(1/\delta))$ which means one could expect the worst case behavior with probability roughly δ. Therefore, optimizing for the worst case is better suited for those levels and opting for a fixed k_h is preferred to diminishing values of k_h. We suggest to set $k_h = k$ when $h \ge H - O(\log\log(1/\delta))$ and $k_h \approx k(2/3)^{H-h}$ otherwise. By analyzing the worst case error of the top $\log\log(1/\delta)$ compactors separately from the bottom $H - \log\log(1/\delta)$ we improve our analysis to $O((1/\varepsilon) \log^2 \log(1/\delta))$. This sketch is fully mergeable. Interestingly, the worst case analysis of the top $\log\log(1/\delta)$ compactors is identical to the analysis of the MRL sketch above.

This last observation leads us to our third and final improvement. If one replaces the top $\log\log(1/\delta)$ compactors with a GK sketch, the space complexity can be shown to reduce to $O((1/\varepsilon) \log\log(1/\delta))$. However, this prevents the sketch from being mergeable because the GK sketch is not known to have this property.

Another way to view this algorithm is as a concatenation of three sketches. The first, receiving the stream of elements is a sampler that simulates all the compactors of capacity 2. Its output is fed into a sketch composed of a sequence of compactors of increasing sizes, as described above. We refer to such a sketch as a *KLL* sketch. This sketch outputs $O((1/\varepsilon) \,\mathrm{poly}\log(1/\delta))$ items that are fed into an instance of GK. This idea is illustrated in Figure 2.

Our final contribution (Section V) is that of improving the lower-bound from the trivial $\Omega(1/\varepsilon)$ to $\Omega((1/\varepsilon) \log\log(1/\delta))$. We do this by leveraging the lower bound of $\Omega((1/\varepsilon) \log(1/\varepsilon))$ of [9] for deterministic algorithms. This proves that streaming quantile approximation is a rare case in which the correct space complexity of a sketch is doubly logarithmic in the failure probability.

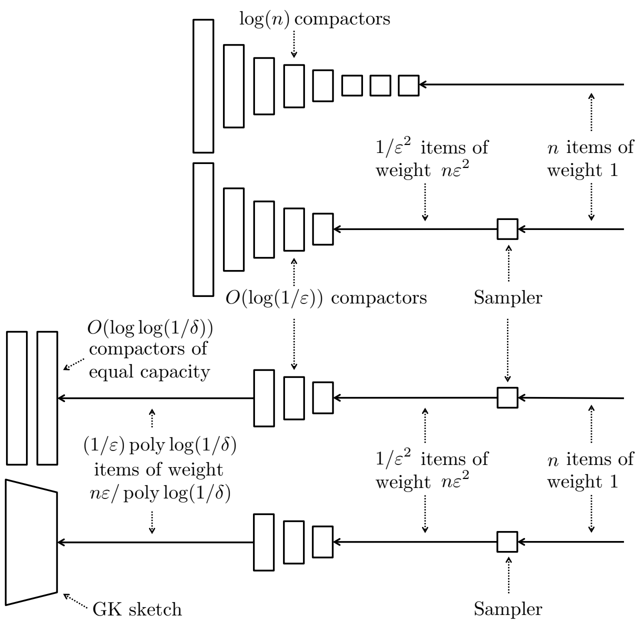
II. Algorithm and Analysis

As mentioned above, our sketching algorithm (KLL) includes a hierarchy of compactors with varying capacities. Consider a run of the algorithm that terminates with H different compactors. The compactors are indexed by their hight $h \in 1, \ldots, H$. The weight of items at hight h is $w_h = 2^{h-1}$. Denote by k_h the smallest number of items that the compactor at height h contains during a compact operation. For brevity, denote $k = k_H$. For reasons that will become clear later assume that $k_h \ge kc^{H-h}$ for $c \in (0.5, 1)$.

Since the top compactor was created, we know that the second compactor from the top compacted its elements at least once. Therefore $n \ge k_{H-1}w_{H-1} = k_{H-1}2^{H-2}$ which

	Single quantile	All quantiles	Randomized	Mergeable
MRL [3]	$(1/\varepsilon)\log^2(\varepsilon n)$	$(1/\varepsilon)\log^2(\varepsilon n)$	No	Yes
GK [5]	$(1/\varepsilon)\log(\varepsilon n)$	$(1/\varepsilon)\log(\varepsilon n)$	No	No
ACHPWY [7]	$(1/\varepsilon)\log(1/\varepsilon)\sqrt{\log(1/\delta)}$	$(1/\varepsilon)\log(1/\varepsilon)\sqrt{\log(1/\varepsilon\delta)}$	Yes	Yes
FO [6] $\delta = e^{-(1/\varepsilon)^c}$	$(1/\varepsilon)\log(1/\varepsilon)$	$(1/\varepsilon)\log(1/\varepsilon)$	Yes	No
KLL [This paper]	$(1/\varepsilon)\log^2\log(1/\delta)$	$(1/\varepsilon)\log^2\log(1/\delta\varepsilon)$	Yes	Yes
KLL [This paper]	$(1/\varepsilon)\log\log(1/\delta)$	$(1/\varepsilon)\log\log(1/\delta\varepsilon)$	Yes	No

Table I
THE TABLE DESCRIBES THE SPACE COMPLEXITY OF SEVERAL STREAMING QUANTILES ALGORITHMS IN BIG-O NOTATION. ALL ALGORITHMS ARE REQUIRED TO SOLVE THE SINGLE QUANTILE PROBLEM AND OPERATE IN THE COMPARISON MODEL FOR ARBITRARILY ORDERED STREAMS. THE RANDOMIZED ALGORITHMS ARE REQUIRED TO SUCCEED WITH PROBABILITY $1-\delta$. THE ALGORITHM OF FELBER AND OSTROVSKY [6] USES A FIXED FAILURE PROBABILITY $\delta = e^{-(1/\varepsilon)^c}$. THE NON-MERGEABILITY OF SOME OF THESE ALGORITHMS IS DUE TO THEIR RELIANCE ON GK WHICH IS NOT KNOWN TO BE MERGEABLE.



$\log(n)$ compactors
$1/\varepsilon^2$ items of weight $n\varepsilon^2$
n items of weight 1
$O(\log(1/\varepsilon))$ compactors
Sampler
$O(\log\log(1/\delta))$ compactors of equal capacity
$(1/\varepsilon)\,\mathrm{poly}\log(1/\delta)$ items of weight $n\varepsilon/\,\mathrm{poly}\log(1/\delta)$
$1/\varepsilon^2$ items of weight $n\varepsilon^2$
n items of weight 1
GK sketch
Sampler

Figure 2. Sampling, varying capacity compactors, equal capacity compactors and GK sketches achieve different efficiencies at different stream lengths. The set of capacitated compactors used by KLL creates a smooth transition between sampling and GK sketching. The top figure corresponds to the first construction explained in Section II. The second from the top is explained in Section III. The two bottom figures correspond to our main contributions and are explained in Sections IV and IV-A respectively.

gives

$$H \le \log(n/k_{H-1}) + 2 \le \log(n/ck) + 2 \ .$$

Using the above we can bound the number of compact operations m_h at height h. Every compact procedure call is performed on at least k_h items and the items have a weight of $w_h = 2^{h-1}$. Therefore,

$$m_h \le \frac{n}{k_h w_h} \le \frac{2n}{k2^H}(2/c)^{H-h} \le (2/c)^{H-h-1} \ .$$

To analyze the total error produced by the sketch, we first consider the error generated in each individual level. Define by $R(x,h)$ the rank of x among the following weighted set of items. The items yielded by the compactor at height h and all the items stored in the compactors of heights $h' \le h$ at end of the stream. For convenience $R(x,0) = R(x)$ is the exact rank of x in the input stream. Define $\mathrm{err}(x,h) = R(x,h) - R(x,h-1)$ to be the total change in the approximated rank of x due to level h.

Note that each compaction operation in level h either leaves the rank of x unchanged or adds w_h or subtracts w_h with equal probability. To be more explicit, if x has an even rank among the item inside the compactor, the total mass to the left of it (its rank) is unchanged by the compaction operation. If its rank inside the compactor is odd however and the odd items are chosen, this mass increases by w_h. If its rank inside the compactor is odd and the even items are chosen, its mass decreases by w_h. Therefore, $\mathrm{err}(x,h) = \sum_{i=1}^{m_h} w_h X_{i,h}$ where $\mathbb{E}[X_{i,h}] = 0$ and $|X_{i,h}| \le 1$. The final discrepancy between the real rank of x and its approximation $\tilde{R}(x) = R(x,H)$ is

$$R(x,H) - R(x,0) = \sum_{h=1}^{H} R(x,h) - R(x,h-1) =$$

$$\sum_{h=1}^{H} \mathrm{err}(x,h) = \sum_{h=1}^{H}\sum_{i=1}^{m_h} w_h X_{i,h} \ .$$

Lemma 1 (Hoeffding). *Let $X_1,\ldots,X_m$ be independent random variables, each with an expected value of zero, taking values in the range $[-w_i, w_i]$. Then for any $t > 0$ we* *have*

$$\Pr\left[\left|\sum_{i=1}^{m} X_i\right| > t\right] \le 2\exp\left(-\frac{t^2}{2\sum_{i=1}^{m} w_i^2}\right)$$

with exp *being the natural exponent function.*

We now apply Hoeffding's inequality to bound the probability that the bottom $H' \le H$ compactors contribute more than εn to the total error. The reason for considering only the bottom levels and not all the levels will become apparent in Section III.

$$\Pr\left[|R(x,H') - R(x,0)| > \varepsilon n\right] = \Pr\left[\sum_{h=1}^{H'}\sum_{i=1}^{m_h} w_h X_{i,h} > \varepsilon n\right] \le 2\exp\left(-\frac{\varepsilon^2 n^2}{2\sum_{h=1}^{H'}\sum_{i=1}^{m_h} w_h^2}\right) \quad (1)$$

A straight forward computation shows that

$$\begin{aligned}\sum_{h=1}^{H'}\sum_{i=1}^{m_h} w_h^2 &= \sum_{h=1}^{H'} m_h w_h^2 \le \\ &\sum_{h=1}^{H'} (c/2)^{H'-h-1} 2^{2h-1} = \\ &\frac{(2/c)^{H'-1}}{4}\sum_{h=1}^{H'}(2c)^h \le \\ &\frac{(2/c)^{H'-1}}{4}\frac{(2c)^{H'}}{2c-1} \le \frac{c}{8(2c-1)}2^{2H'}\end{aligned}$$

Substituting $2^{2H'} = 2^{2H}/2^{2(H-H')}$ and recalling that $H \le \log(n/ck) + 2$ we get that

$$\sum_{h=1}^{H'}\sum_{i=1}^{m_h} w_h^2 \le \frac{n^2/k^2}{2c^2(2c-1)}\frac{1}{2^{2(H-H')}} \quad (2)$$

Substituting Equation 2 into Equation 1 and setting $C = c^2(2c-1)$ we get the following convenient form

$$\Pr\left[|R(x,H') - R(x)| \ge \varepsilon n\right] \le 2\exp(-C\varepsilon^2 k^2 2^{2(H-H')}) \quad (3)$$

Theorem 1. *There exists a streaming algorithm that computes an ε approximation for the rank of a single item with probability $1-\delta$ whose space complexity is $O((1/\varepsilon)\sqrt{\log(1/\delta)} + \log(\varepsilon n))$. This algorithm also produces mergeable summaries.*

Proof: Let $k_h = \lceil kc^{H-h}\rceil + 1$. Note that k_h changes throughout the run off the algorithm since H changes. Nevertheless, H can only increase and so k_h is monotonically decreasing with the length of the stream. This matches the requirement that $k_h \ge kc^{H-h}$ where H is the final number of compactors. Notice that k_h is at least 2.

Setting $H' = H$ in Equation 3 and requiring failure probability at most δ we conclude that it suffices to set $k = (C/\varepsilon)\sqrt{\log(2/\delta)}$. The space complexity of the algorithm is $O\left(\sum_{h=1}^{H} k_h\right)$.

$$\sum_{h=1}^{H} k_h \le \sum_{h=1}^{H}(kc^{H-h}+2) \le k/(1-c) + 2H = O(k + \log(n/k))$$

Setting $\delta = \Omega(\varepsilon)$ suffices to union bound over the failure probabilities of $O(1/\varepsilon)$ different quantiles. This provides a mergeable sketching algorithm for the all quantiles problem of space $O((1/\varepsilon)\sqrt{\log(1/\varepsilon)} + \log(\varepsilon n))$.

The KLL sketch provides a *mergeable summary*. In a merge operation, same height compactors are concatenated together. Then, each level that contains more than k_h elements is compacted. The value of k_h is based on the new maximal height H which is derived from the combined lengths of the two streams. In either of the two merged sketches, each compaction at level h involved at least k_h items which means the proof above still holds. ■

Note that this result already improves on the best known prior art in the parameter setting where $n = \exp(O(1/\varepsilon))$.

Theorem 2. *There exists a streaming algorithm that computes an ε approximation for the rank of a single item with probability $1-\delta$ whose space complexity is $O\left((1/\varepsilon)\sqrt{\log(1/\delta)}\right)$.*

Proof: The KLL sketch maintains a total of $H = \log(\varepsilon n)$ sketches. Note, however, that only $O(\log(k))$ compactors have capacity greater than 2. More accurately the bottom $H'' = H - \lceil\log(k)/\log(1/c)\rceil$ all have capacity exactly 2. Each of those receives two items at a time, performs a random match between them, and sends the winner of the match to the compactor of the next level. Hence, the compactor of level H'' simply selects one item uniformly at random from every $2^{H''}$ elements in the stream and passes that item with weight $2^{H''}$ to the compactor at hight $H''+1$. This is easily simulated using $O(1)$ space which replaces the bottom H'' compactors and reduces the space complexity of the algorithm. ■

There is, however, a drawback in replacing the bottom H'' compactors with a simple sampler. When merging two sketches, it is not clear how to merge the samplers in a correct way. The next section explains how to do exactly that.

III. Sampling and Keeping Mergeablity

In the new sketch we have, in addition to the compactors, a new object we call a sampler. The sampler supports an UPDATE method that introduces an item of weight w to the sketch. When observing items in a stream the weight is always set to 1. However, when merging two sketches we require supporting an update of arbitrary weights. At any time the sampler has an associated height h. It outputs items of weight 2^h as inputs to the compactor of level $h+1$. This associated height will increase over time to eventually being roughly $H - \log(1/\varepsilon)$. Apart for sampler of height h, the sketch maintains compactors at heights greater than h.

The sampler keeps a single item in storage along with a weight of at most $2^h - 1$. When merging two sketches, the sketch with the sampler of smaller height will feed its item with its appropriate weight to the sampler of the other sketch. Also, all compactors with height $\leq h$ in the 'smaller' sketch will feed the items in their buffers to the sampler of the 'larger sketch'.

The UPDATE operation is performed as follows. Denote by v the weight of the internal item stored in the sampler and by w the weight of the newly introduced item. If $v + w \leq 2^h$, the sampler replaces its stored item with the new one with probability $w/(v+w)$ as in Reservoir Sampling. If $v + w = 2^h$ the sampler outputs the stored item and sets the internal weight w to 0. If however $v + w > 2^h$ (notice this can only happen if $w > 1$) the sampler discards the heavier item, and keeps the lighter item with a weight of $\min\{w, v\}$. With probability $\max\{w, v\}/2^h$ it also outputs the heavier item with weight 2^h.

It is easy to verify that the above described sketching scheme corresponds to the following offline operations. The sampler outputs items by performing the action *sample*; it takes as input a sequence of items W items such that $2^{h-1} < W \leq 2^h$. With probability $W/2^h$ it outputs one of the observed items chosen at random. With probability $1 - W/2^h$ it outputs nothing. Before analyzing the error associated with a *sample* operation we mention that, if no merges are performed, it suffices to restrict the value of W to be exactly 2^h. However, in order to account for merges we must let W obtain values in the range $W \in (2^{h-1}, 2^h]$.

Lemma 2. *Let $R(x, h, i)$ be the rank of x after sample operation i of the sampler of height h, where the rank is computed based on the stream elements that did not undergo a sample operation, and the weighted items outputted by the sample operations 1 through i. Let $R(x, h, i) - R(x, h, i-1) = 2^h Y_{h,i}$. Then $\mathbb{E}[Y_{h,i}] = 0$ and $|Y_{h,i}| \leq 1$.*

Proof: Let r be the exact rank of x among the input items before the sampling operation. Denote by W the sum of weights of the input items. After the sampling, those input items are replaced with a single element. The rank of x after the sampling is 2^h with probability $\frac{W}{2^h} \cdot \frac{r}{W}$ and 0 otherwise. The first term is the probability of outputting anything. The second is of selecting an element smaller than x. Therefore, the expected value of the rank of x after the sample operation is $2^h \frac{W}{2^h} \cdot \frac{r}{W} = r$, hence the expected value of the difference mentioned in the claim is 0. Clearly, the maximal value of this difference is bounded by 2^h. ■

Let m_h be total number of times a *sample* operation can be performed at height h. Since the sampler at that height takes items with a total minimum weight of $W > 2^{h-1}$ and there are a total of n items (with overall weight n)

$$m_h \leq \frac{n}{2^{h-1}} \ .$$

It follows that the expression of the error accounted for samplers of heights up to[2] H'' can be expressed as

$$\mathrm{err}_{H''} = \sum_{h=1}^{H''} \sum_{i=1}^{m_h} [R(x,h,i) - R(x,h,i-1)] = \sum_{h=1}^{H''} \sum_{i=1}^{m_h} 2^h Y_{i,h}$$

with $Y_{i,h}$ being independent, $\mathbb{E}[Y_{i,h}] = 0$ and $|Y_{i,h}| \leq 1$. We compute the sum of weights appearing in Hoeffding's inequality (Lemma 1) in order to apply it.

$$\sum_{h=1}^{H''} \sum_{i=1}^{m_h} 2^{2h} \leq \sum_{h=1}^{H''} n 2^{h+1} \leq 4n 2^{H''} = \frac{4n 2^H}{2^{H-H''}} \leq \frac{16n^2}{ck 2^{H-H''}}$$

Hence,

$$\Pr\left[\mathrm{err}_{H''} > \varepsilon n\right] \leq 2 \exp\left(-c\varepsilon^2 k 2^{H-H''}/32\right) \ . \quad (4)$$

The following is immediate from Equations 3 and 4.

Theorem 3. *Assume we apply a KLL sketch that uses H levels of compactors, with capacity $k_h \geq kc^{H-h}$. Also, assume that an arbitrary subset of the stream is fed into samplers of heights 1 through H'', while the output of these samplers is fed to appropriate compactors. Then for any $H' > H''$ it holds that*

$$\Pr\left[\mathrm{err}_{H'} > 2\varepsilon n\right] <$$

$$2 \exp\left(-c\varepsilon^2 k 2^{H-H''}/32\right) + 2 \exp\left(-C\varepsilon^2 k^2 2^{2(H-H')}\right)$$

Here, $\mathrm{err}_{H'}$ denotes the error of the stream outputted by the compactors of level H', and $C = c^2(2c-1)$.

By taking $H' = H$, $H'' = H - O(\log(k))$, and $k = (1/\varepsilon)\sqrt{\log(1/\delta)}$ we obtain the following corollary.

Corollary 1. *There exists a streaming algorithm that computes an ε approximation for the rank of a single item with probability $1 - \delta$ whose space complexity is $O((1/\varepsilon)\sqrt{\log(1/\delta)})$. This algorithm also produced mergeable summaries.*

[2]Notice that unlike compactors, there is no hierarchy of samplers. However, due to the fact that the height of the sampler grows with time, the sample operations may be performed on different heights. The only guarantee is that any item appears in at most one sample operation and that the height of the sampler is always at most H''.

IV. Reducing the Failure Probability

In this section we take full advantage of Theorem 3 to obtain a streaming algorithm with asymptotically better space complexity. Notice that the lion share of the contribution to the error is due to the top compactors. For those, however, Hoeffding's bound is not tight. Let $s = O(\log\log(1/\delta))$ be a small number of top layers. For the bottom $H-s$ layers we use Theorem 3, applied on $H' = H - s$ and corresponding H'', to bound their error. For the top s we simply use a deterministic bound.

Theorem 4. *There exists a streaming algorithm that computes an ε approximation for the rank of a single item with probability $1 - \delta$ whose space complexity is $O((1/\varepsilon)\log^2\log(1/\delta))$. This algorithm also produces mergeable summaries.*

Proof: Using Theorem 3 we see that the bottom compactors of height at most $H' = H - s$ and the sampler, when set to be of height at most $H'' = H - 2s - \log_2(k)$, contribute at most εn to the error with probability $1 - \delta$ at long as $\varepsilon k 2^s \geq c'\sqrt{\log(2/\delta)}$ for sufficiently small constant c'. For the top s compactors, we set their capacities to $k_h = k$. That is, we do not let their capacity drop exponentially. Those levels contribute to the error at most $\sum_{h=H'+1}^{H} m_h w_h = \sum_{h=H'+1}^{H} n/k_h = sn/k$. Requiring that this contribution is at most εn as well we obtain the relation $s \leq k\varepsilon$. Setting $s = O(\log\log(1/\delta))$ and $k = O(\frac{1}{\varepsilon}\log\log(1/\delta))$ satisfies both conditions. The space complexity of this algorithm is dominated by maintaining the top s levels which is $O(ks) = O((1/\varepsilon)\log^2\log(1/\delta))$. ■

Interestingly, the analysis of the top s levels is identical to the equal capacity compactors used in the MRL sketch. In the next section we show that one could replace the top s levels with a different algorithm and reduce the dependence on δ even further.

A. Gaining Space Optimality; Potentially Losing Mergeability

The most space efficient version of our algorithm, with respect to the failure probability, operates as follows. For δ being the target error probability we set

$$s = \left\lceil \log_2\left(c'\sqrt{\log(2/\delta)}/(k\varepsilon)\right)\right\rceil = O(\log\log(1/\delta))$$

as in the section above but set $k = O(1/\varepsilon)$. At any time point we keep 2 different copies of the GK sketch, tuned for a relative error of ε. They are correspondingly associated with the compactors of heights $h_1 < h_2$ which are the two largest height values that are multiples of s. The GK sketch associated with height h receives as input the outputs of the compactor of layer $h-1$. For $h = 0$ the GK sketch associated with it receives as input the stream elements. When a new GK sketch is built due to a new compactor being formed the bottom one is discarded.

Theorem 5. *There exists a streaming algorithm that computes an ε approximation for the rank of a single item with probability $1 - \delta$ whose space complexity is $O((1/\varepsilon)\log\log(1/\delta))$.*

Proof: Notice that the height of h_1 is at least $H - 2s$. It follows that the total number of items that is ever fed into a single GK sketch at most $n_1 = k2^{2s}$. Applying Theorem 3 again, on $H' = H - s$, and $H'' = H - 2s - \log_2(k)$, the error w.r.t. to the output of the compactor feeding elements into the GK sketch matching h_1 is at most $O(\varepsilon n)$, with $k = O(1/\varepsilon)$. Therefore, the sum of errors is still $O(\varepsilon n)$. The memory required by the GK sketch with respect to its input is at most $O((1/\varepsilon)\log(\varepsilon n_1)) = O((1/\varepsilon)s) = O((1/\varepsilon)\log\log(1/\delta))$, which dominates the memory of our sketch with $k = O(1/\varepsilon)$. The claim follows. ■

We note that the GK sketch is not known to be fully mergeable and so that property of the sketch is lost by this construction. That being said, we point out that the GK sketch is *one-way mergeable*. One-way mergeability is a weaker form of mergeability that informally states that the following setting can work: The data is partitioned among several machine, each creates a summary of its own data, and a single process merges all of the summaries into a single one. For example, stated in terms of Definition 3, when the error $\varepsilon(S(N), N)$ is linear in the size of the sketch N, i.e., $\varepsilon(S(N), N) = \varepsilon_S|N|$, a merge operation resulting in an error of $\varepsilon_S|N_1| + 2\varepsilon_S|N_2|$ rather than $\varepsilon_S|N_1| + \varepsilon_S|N_2|$ is one-way mergeable, while it is not (fully) mergeable. It was pointed out by [2], [7] that any sketch for the all quantiles approximation problem is one-way mergeable.

V. Tightness of Our Result

In [9] a lower bound is given for deterministic algorithms. A more precise statement of their result, implicitly shown in the proof of their Theorem 2, is as follows

Lemma 3 (Implicit in [9], Theorem 2)**.** *Let $\mathcal{A}_{\mathcal{D}}$ be deterministic comparison based algorithm solving single quantile ε approximate for all streams of length at most $C(1/\varepsilon)^2\log(1/\varepsilon)^2$ for some sufficiently large universal constant C. Then $\mathcal{A}_{\mathcal{D}}$ must store at least $c(1/\varepsilon)\log(1/\varepsilon)$ elements from the stream for some sufficiently small constant c.*

Below we obtain a lower bound matching our result completely for the case of single quantile problem and almost completely for the case of ε approximation of all quantiles.

Theorem 6. *Let $\mathcal{A}_{\mathcal{R}}$ be a randomized comparison based algorithm solving the ε approximate single quantile problem with probability at least $1 - \delta$. Then $\mathcal{A}_{\mathcal{R}}$ must store at least $\Omega\left((1/\varepsilon)\log\log(1/\delta)\right)$ elements from the stream.*

Proof: Assume by contradiction that there exists a randomized algorithm $\mathcal{A}_{\mathcal{R}}$ that succeeds in computing a single quantile approximation up to error ε with probability $1-\delta$ while storing $o\left((1/\varepsilon)\log\log(1/\delta)\right)$ elements from the stream. Let n be the length of the stream and $\delta = 1/2n!$. With probability $1/2$ the randomized algorithm succeeds simultaneously for all $n!$ possible inputs. Let r denote a sequence of random bits used by $\mathcal{A}_{\mathcal{R}}$ in one of these instances. It is now possible to construct a deterministic algorithm $\mathcal{A}_D(r)$ which is identical to $\mathcal{A}_R$ but with r hardcoded into it. Note that $\mathcal{A}_D(r)$ deterministically succeeds for streams of length n. Let $n = C(1/\varepsilon)^2\log(1/\varepsilon)^2$ for the same C as in Lemma 3. We obtain that $\mathcal{A}_D(r)$ succeeds on all streams of length $C(1/\varepsilon)^2\log(1/\varepsilon)^2$ while storing $o((1/\varepsilon)\log(1/\varepsilon))$ elements from the stream. This contradicts Lemma 3 above. ■

The lower bound above perfectly matches the single quantile approximation result we achieve. For the all quantiles problem, using the union bound over a set of $O(1/\varepsilon)$ quantiles shows that $O((1/\varepsilon)\log\log(1/\varepsilon))$ elements suffice. This leaves a potential gap of $\log\log(1/\varepsilon)$ for that problem.

Acknowledgments

The authors want to thank Sanjeev Khanna, Rafail Ostrovsky, Jeff Phillips, Graham Cormode, Andrew McGregor and Piotr Indyk for very helpful discussions and pointers.

argument is your BibTeX string definitions and bibliography database(s)

References

[1] L. Wang, G. Luo, K. Yi, and G. Cormode, "Quantiles over data streams: An experimental study," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '13. New York, NY, USA: ACM, 2013, pp. 737–748. [Online]. Available: http://doi.acm.org/10.1145/2463676.2465312

[2] M. B. Greenwald and S. Khanna, "Quantiles and equidepth histograms over streams," in *In Data Stream Management: Processing High-Speed Data Streams*, J. G. M. Garofalakis and R. Rastogi, Eds. Springer, 2016.

[3] G. S. Manku, S. Rajagopalan, and B. G. Lindsay, "Random sampling techniques for space efficient online computation of order statistics of large datasets," *SIGMOD Rec.*, vol. 28, no. 2, pp. 251–262, Jun. 1999.

[4] J. Munro and M. Paterson, "Selection and sorting with limited storage," *Theoretical Computer Science*, vol. 12, no. 3, pp. 315 – 323, 1980. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0304397580900614

[5] M. Greenwald and S. Khanna, "Space-efficient online computation of quantile summaries," in *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '01. New York, NY, USA: ACM, 2001, pp. 58–66.

[6] D. Felber and R. Ostrovsky, "A randomized online quantile summary in $O((1/\varepsilon)\log(1/\varepsilon))$ words," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, 2015, pp. 775–785.

[7] P. K. Agarwal, G. Cormode, Z. Huang, J. Phillips, Z. Wei, and K. Yi, "Mergeable summaries," in *Proceedings of the 31st Symposium on Principles of Database Systems*, ser. PODS '12. New York, NY, USA: ACM, 2012, pp. 23–34. [Online]. Available: http://doi.acm.org/10.1145/2213556.2213562

[8] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri, "Medians and beyond: New aggregation techniques for sensor networks," in *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 239–249. [Online]. Available: http://doi.acm.org/10.1145/1031495.1031524

[9] R. Y. Hung and H. F. Ting, "A $(1/\varepsilon)\log(1/\varepsilon)$ space lower bound for finding ε-approximate quantiles in a data stream," in *Frontiers in Algorithmics*. Springer, 2010, pp. 89–100.

[10] S. Guha and A. McGregor, "Stream order and order statistics: Quantile estimation in random-order streams," *SIAM J. Comput.*, vol. 38, no. 5, pp. 2044–2059, 2009. [Online]. Available: http://dx.doi.org/10.1137/07069328X

[11] A. Brodnik, A. Lopez-Ortiz, V. Raman, and A. Viola, *Space-Efficient Data Structures, Streams, and Algorithms: Papers in Honor of J. Ian Munro, on the Occasion of His 66th Birthday*. Springer, 2013, vol. 8066.

[12] Q. Ma, S. Muthukrishnan, and M. Sandler, "Frugal streaming for estimating quantiles," in *Space-Efficient Data Structures, Streams, and Algorithms - Papers in Honor of J. Ian Munro on the Occasion of His 66th Birthday*, ser. Lecture Notes in Computer Science, A. Brodnik, A. López-Ortiz, V. Raman, and A. Viola, Eds., vol. 8066. Springer, 2013, pp. 77–96. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40273-9