

# Robust Sketching and Aggregation of Distributed Data Streams

Marios Hadjieleftheriou

John W. Byers

George Kollios

Computer Science Department  
Boston University  
111 Cummington St.  
Boston MA 02215  
{marioh,byers,gkollios}@cs.bu.edu

## Abstract

The data streaming model provides an attractive framework for one-pass summarization of massive data sets at a single observation point. However, in an environment where multiple data streams arrive at a set of *distributed* observation points, sketches must be computed remotely and then must be aggregated through a hierarchy before queries may be conducted. As a result, many sketch-based methods for the single stream case do not apply directly, as either the error introduced becomes large, or because the methods assume that the streams are non-overlapping. These limitations hinder the application of these techniques to practical problems in network traffic monitoring and aggregation in sensor networks. To address this, we develop a framework for evaluating and enabling robust computation of duplicate-sensitive aggregate functions (e.g., SUM and QUANTILE), over data produced by distributed sources. We instantiate our approach by augmenting the Count-Min and Quantile Digest sketches to apply in this distributed setting, and analyze their performance. We conclude with an experimental evaluation to validate our analysis.

## 1 Introduction

Recent work has intensively focused on key research problems involving the summarization of massive data streams to support historical queries. In applications ranging from wide-area network monitoring to data collection across large-scale sensor networks, the volume of data collected can overwhelm processing and storage resources, and thus queriable summaries, or *sketches*, of these datasets are widely used. For many

interesting applications, the data arrives as a *stream* of observations at a single monitoring point, and must then be processed into the sketch *online* and in *one pass*, e.g., the sketch must be able to handle incremental updates (and deletions, if the application demands it). While a wide variety of sketches are available for different queries, all focus first on optimizing the fundamental performance tradeoff of space vs. accuracy, as applications often seek to obtain the best accuracy within a given space bound [1, 20, 25].

A new research thrust has started to focus on the next set of questions that arise in application environments in which the data arrives in multiple distributed streams at a set of observation points. For example, the growing body of work in sensor databases, views the observations at each individual sensor as comprising a data stream that the user can query via a base station at the edge of the network. Queries in sensor databases often focus on computation of *aggregates*, such as the median temperature within a prescribed region of space-time [22, 23, 30, 31]. Ideally, users would like to be able to access all sensor readings and queries could be answered based upon a global view of the sensor field from a centralized location. In practice, the costs of such an approach are prohibitive, as we argue later in more detail.

A second example application involving distributed streams is traffic accounting and monitoring in wide area networks. Consider a number of specialized monitoring devices colocated with backbone routers that have the capability to sample incoming traffic, keep logs and maintain statistics. Network administrators would like to be able to run data analysis tools on the combined information provided by the monitoring routers from arbitrary locations on the network e.g., for detecting DDoS attacks or other anomalous activity [26]. Each monitoring device can be considered as a source that continuously observes a stream of traffic data, and an administrator performing the analysis on

the collective data acts as the aggregation point.

For these applications and many others, the ideal solution for conducting aggregation queries over such networks would be to transmit and combine all streaming data to an aggregation point (i.e., to get the union of the streams), and answer queries locally. Even though this strategy would provide exact query results, it suffers from three problems: (1) Storing the data at a single location is impossible given the maintenance cost and memory requirements imposed by streaming data rates even for a small number of streams; (2) The communication overhead for transmitting all the data to a single location would be excessive; (3) The aggregation point could be a central point of failure. For these reasons, we focus on an alternative hierarchical model, where the distributed observation points perform an initial sketch of their streams, and these sketches are then routed up through an aggregation hierarchy to one or more aggregation points. Our focus is to develop a framework for evaluating which sketching methodologies are amenable to this general approach, and assessing the associated costs both analytically and experimentally.

Our starting point is to understand what set of properties a distributed query model such as this demands from sketches. While it is clear that excellent space-accuracy trade-offs are a requirement, as in the single stream case, the distributed model places other demands on the sketches as well. In both applications, sketching before transmission to the aggregation point will introduce errors; quantification of appropriate bounds on the error is a contribution of a work. In a multi-level hierarchical application, such as in sensor networks, it is essential that the sketches can themselves be *aggregated* into a small space representation, so that the messaging cost of the computation does not overwhelm the network. At the same time, it is preferable that the sketches are *duplicate-insensitive* so that multiple copies of messages can be transmitted to improve fault tolerance without adversely impacting the final computation.<sup>1</sup> In applications involving wide-area networks, small space summaries are needed to avoid excessive consumption of bandwidth when the sketches of packet logs are routed to a centralized processing point. Here, reliable transmission of these relatively large sketches can be realized by TCP, so fault tolerance is not an issue. However, individual packets may often be observed at multiple monitoring points, and these observations should be counted only once in the final aggregate. Therefore, this distributed application requires robust sketching and aggregation methods that can correctly handle *duplicates in the data itself*.

---

<sup>1</sup>Higher level mechanisms for reliable transmission of small datagrams, such as the use of TCP, are widely viewed as too expensive in current sensor network architectures.

The examples above exemplify a set of common challenges in taking techniques designed for sketching a single stream into a variety of distributed settings that we address in this work. We start by formalizing a distributed data streaming model that captures the notions developed in the discussion above and set up definitions with which to evaluate sketching methodologies in this model, and connect our approach to the large body of prior work. We then describe our methods to allow two dissimilar sketching methodologies, Count-Min sketches [9] and Quantile Digests [29], to be used in the distributed setting, and also present cases where our methodology does not readily apply. Through analysis and experimental evaluation on large simulated datasets we demonstrate that the accuracy and space costs of applying our approach for these two methodologies are small (less than 10% increase in the variance of the estimates and a constant factor increase in the total space utilization). As a result, they can readily be applied to an important, broader class of applications than single-stream sketching methods (which can incur significant error if used in distributed domains).

## 2 Problem Setting

We now present the general problem setting for computing aggregate queries over multiple data streams that are distributed over a set of  $n$  remote observation points, or hosts. The hosts can communicate with each other using an underlying network infrastructure. We are given a set of *streams*  $\mathcal{D} = \{D_1, \dots, D_n\}$  and an *aggregation point*  $A$  where we would like to issue and answer aggregate queries over the data that appear on the union of streams  $D_U = \bigcup_{i=1}^n D_i$ . For simplicity, we assume that there is a one-to-one correspondence between data streams and hosts and use  $D_i$  to refer to the data stream at host  $i$ . Finally, let  $\mathcal{S}(D)$  denote a synopsis maintained for stream  $D$ .

If bandwidth is plentiful, an ideal synopsis of the union of streams can be computed at the aggregation point. This simple solution creates only one synopsis  $\mathcal{S}(D_U)$  based on a one-pass sketch of a combination of all streams at a predetermined aggregation point. We refer to this idealized sketch as the *master sketch*, and note that this master sketch hides the fact that the observation points are distributed, since the sketching is done only after the raw data has been combined. The messaging cost of this approach is measured by the data transmitted from observation points to aggregation points, i.e.,  $\sum_{i=1}^n |D_i|$ . In practice, it will be prohibitively expensive to stream this much data from observation points to aggregation points in full fidelity, so the results obtained by the master sketch will form a baseline that is used in comparison to practical methods which perform summarization at the observation points.

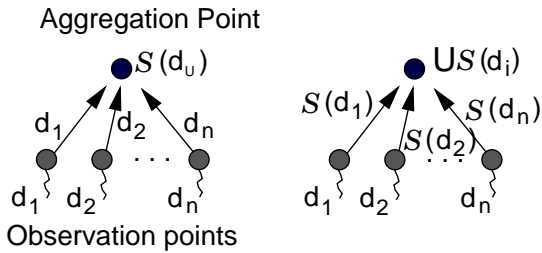


Figure 1: Master sketch approach vs. union of sketches

The family of practical solutions that we consider in this paper considers a two-level hierarchy, but which can readily be extended to a multi-level hierarchy. At each observation point we create one synopsis  $\mathcal{S}(D_i)$  for each data stream. These synopses are reliably routed to an aggregation point, which computes a combined synopsis consisting of a *union of sketches*:  $\mathcal{S}_\cup = \bigcup_{i=1}^n \mathcal{S}(D_i)$ . While this methodology is typically less accurate than the master sketch approach, it affords the advantage of significantly reduced messaging cost of data routed to the aggregation point. Here, the messaging cost is only a function of the sketch sizes, not the original streams:  $\sum_{i=1}^n |\mathcal{S}(D_i)|$ . A depiction of the master sketch approach against the more pragmatic union of sketches approach appears in Figure 1.

Our first objective is to consider the extent to which the union of sketches approach reflects the master sketch approach, noting that it is of course impossible to provide complete equivalence. We propose the following two definitions that provide different guarantees across the two methods.

**Definition 1** (Unbiasedness). Let  $\mathbb{E}\{\mathcal{S}(D)[q]\}$  denote the expected result of query  $q$  on sketch  $\mathcal{S}(D)$ , where the expectation is taken over random choices made in generating sketch  $\mathcal{S}$ . A union of sketches is said to be *unbiased* with respect to the master sketch if for all  $q$ ,  $\mathbb{E}\{\mathcal{S}(D_\cup)[q]\} = \mathbb{E}\{\mathcal{S}_\cup[q]\}$ .

**Definition 2** (Bounded Error). Suppose a sketching method must provide  $(\epsilon, \delta)$ -approximate query results over a set of queries  $\mathcal{Q}$ . A union of sketches is said to have *bounded error* with respect to the master sketch if for all  $q \in \mathcal{Q}$ , both sketches provide  $(\epsilon, \delta)$ -approximation guarantees for  $q$ .

In the first of the two notions, the union of sketches provides answers to queries that are unbiased with respect to the master sketch (but may have increased variance). In the second, there is no guarantee that the union of sketches is unbiased, but it is guaranteed that it returns  $(\epsilon, \delta)$ -approximate results. We shall demonstrate that Count-Min sketches satisfy the first notion, while Quantile Digests satisfy the latter.

Our second objective is to ensure that the aggregation methods we consider are robust, and in particular are order-insensitive and resilient to the presence of duplicates, as advocated in earlier work on sensor

aggregation [28, 7]. This is relevant to us more generally, as we wish to avoid double-counting duplicates across distributed streams, as we wish the results of union computations to be insensitive to the ordering of particular operations. We recap the basic definitions intuitively next; a formal definition is provided in [28].

**Definition 3** (Order Insensitivity). Consider a data stream  $D$  of items  $\alpha_1, \alpha_2, \dots, \alpha_m$ , and let  $\pi$  denote an arbitrary permutation on  $m$  items, and  $\pi(D)$  denote the stream of items after applying permutation  $\pi$ . A sketching algorithm is said to be *order-insensitive* if  $\forall \pi, \mathcal{S}(D) = \mathcal{S}(\pi(D))$ . Similarly, a sequence of union operations are *order-insensitive* if the result is not affected by reordering.

**Definition 4** (Duplicate Insensitivity). Consider a data stream  $D$  of distinct items  $\alpha_1, \alpha_2, \dots, \alpha_m$ . Now consider another stream  $D^+$  which consists of  $D$  as a subsequence but with arbitrary repetition of items. A sketching algorithm is said to be *duplicate-insensitive* if  $\forall D, D^+, \mathcal{S}(D) = \mathcal{S}(D^+)$ .

Our approach will be to establish the extent to which single-stream sketching methodologies can be modified to satisfy the conditions and properties enumerated above in our distributed streaming model.

## 2.1 Flajolet-Martin Sketches

One of the essential building blocks of our approach to realize the latter two properties of order- and duplicate-insensitivity is the use of Flajolet-Martin (FM) sketches [13]. As observed in previous work on aggregation in sensor databases, aggregation applications which employ integer counters are not duplicate-insensitive, as both duplicate occurrences of counted values and duplicate transmissions of counted values cause double-counting. An elegant fix is to apply an FM sketch to any integer value to render it duplicate-insensitive [7, 28], as we describe next. This method enables us to use both dispersity routing of aggregates as described in previous work, as well as providing a simple method for avoiding double-counting of identical items witnessed in multiple distributed streams.

We begin by describing the basic structure of the FM sketch. The FM sketch was introduced for estimating the number of distinct objects in a multi-set. FM requires a hash function  $h$  which takes as input an object id  $o$ , and outputs a pseudo-random integer  $h(o)$  with a geometric distribution, i.e.,  $\Pr[h(o) = v] = 2^{-v}$  for  $v \geq 1$ . A sketch consists of  $r$  bits, whose initial values are set to 0 (an appropriate choice of  $r$  is discussed later). For every object  $o$  in the multi-set, FM sets the  $h(o)$ -th bit of the sketch to 1. The key to duplicate-insensitivity of the FM sketch is that regardless of the number of times an identical object is inserted, the same bit in the sketch is always set. This also applies

---

**Algorithm 1** FM\_PCSA

---

```
1: Initialize  $m$  sketches  $FM_1, FM_2, \dots, FM_m$ , each with  $r$ 
   bits set to 0
2: for each object  $o$  in  $D$  do
3:   randomly pick a sketch  $FM_i (1 \leq i \leq m)$ 
4:    $FM[h(o)] = 1$ 
5:    $k = 0$ 
6:   for  $i = 1, \dots, m$  do
7:     for  $j = 1, \dots, r$  do
8:       if  $FM_i[j] = 0$  then
9:          $k = k + 1$ 
10:      break // go to the next sketch
11: return  $1.29m \cdot 2^{\frac{k}{m}}$ 
```

---

in distributed settings, provided that identical random hash functions are used.

To convert an FM sketch back to a value, the following approach is used. FM finds the first bit of the sketch that is still 0. Let the position of this bit be  $k$ ; then the number of distinct objects is estimated as  $n = 1.29 \cdot 2^k$ . This value is an unbiased estimate, i.e.,  $E\{n\} = 1.29 \cdot 2^{E\{k\}}$ . However, the variance of  $k$  is approximately equal to 1.12, and thus the estimate for  $n$  is frequently off by a factor of two or more. To remedy this deficiency, Flajolet and Martin proposed using  $m$  independent sketches, each with its own independent hash function, and averaging the resulting values. In order to keep processing costs expected  $O(1)$  instead of  $O(m)$ , Flajolet and Martin also proposed a technique called *Probabilistic Counting with Stochastic Averaging (PCSA)*. PCSA applies a second hash function to choose only one of the  $m$  sketches and performs the object insertion only to this sketch. As a result, each sketch is responsible for approximately  $\frac{n}{m}$  (distinct) objects, resulting in a new estimation formula:  $1.29m \cdot 2^{\frac{1}{m} \sum_{i=1}^m k_i}$ , with expected standard error  $O(m^{-\frac{1}{2}})$ . The pseudo-code of FM with PCSA is shown in Algorithm 1.

As shown in [13], an appropriate value for  $r$ , the sketch size, is  $O(\log_2 n)$ , where  $n$  is the number of distinct objects. The resulting space consumption of the FM algorithm is  $O(m \log_2 n)$  when  $m$  sketches are used. An extension to FM sketches appeared in [4, 15] that provides an  $(\epsilon, \delta)$ -approximation for the distinct counting problem. The space that is needed to guarantee a  $(1 + \epsilon)$  approximation, with probability at least  $1 - \delta$  is  $O(\frac{1}{\epsilon^2} \log(\frac{1}{\delta}) \log(n))$ . Later on, we will use this method to provide various error bounds for our duplicate insensitive sketches. A number of advanced methods based on FM sketches have also been proposed [1, 4, 17]; all of these could be used for our applications.

As shown in [7], FM sketches are both order- and duplicate-insensitive. We can also show that the union of FM sketches over a distributed environment is unbiasedness with respect to a master sketch, even in the presence of duplicate values and duplicate sketches, as shown in [28]. This property will prove useful for ar-

guing about the unbiasedness and the bounded error of more involved FM sketch constructions.

## 2.2 Other Related Work

Most prior work on data stream aggregation has concentrated on the single-stream, centralized case [1, 20, 18, 19, 25, 12, 2]. Only recently has the problem of computing aggregates on the union of distributed data streams received some interest. Gibbons and Tirthapura [17] show how to estimate simple bit level functions on the union of data streams using a sampling based technique. Das et al. [10] present a method to estimate the cardinality of set expressions on distributed streams that can be overlapping. Their method is based on the technique for distributed Top-k monitoring proposed by Babcock and Olston [3]. However, they don't discuss aggregation or sketch-based methods. Finally, a recent paper by Manjhi et al. [24], presents a method that finds frequent items in the union of distributed data streams, with small communication complexity. They utilize existing single stream sketches (based on Lossy Counting [25] or majority counting [11]) that are created at each site and periodically transmitted to an aggregation point. Although similar to ours, their problem is simpler, since they do not consider the case of duplicate values by assuming that the streams have no overlapping.

In sensor networks, in-network aggregation has been used in almost all sensor data management systems to reduce the power consumption of the network and improve its lifetime. The need for duplicate insensitivity has been recognized as an important issue since it allows for more robust multi-path or flooding based communication protocols. Methods to address this problem have been proposed in parallel by a number of researchers [7, 5, 28]. However, all these methods considered only single values produced by each sensor and not a stream as we consider in this paper. All of these proposed solutions build on FM sketches.

## 3 Distributed Count-Min Sketches

We now demonstrate how to augment the first of two single-stream sketches to make it suitable for use in our distributed model and analyze its performance. The Count-Min (CM) sketch was introduced by Cormode and Muthukrishnan [9] for accurately estimating the frequency of an element on a stream.

### 3.1 CM Sketch Basics

Let  $D = \{\alpha_1, \dots, \alpha_{|D|}\}$  denote a sequence of elements from the domain  $[M] = \{1, \dots, M\}$ . The CM sketch is a simple randomized data structure that can estimate the frequency of any element  $\alpha$  in  $D$ , and consists of

a  $k \times m$  matrix of counters:

$$CM = \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,m} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{k,1} & c_{k,2} & \cdots & c_{k,m} \end{bmatrix}$$

To estimate the frequency of a given element, the CM sketch uses a simple hashing scheme with a set of  $k$  independent hash functions  $h_1, \dots, h_k : [M] \rightarrow [m]$ , one hash function per row of the matrix. Each hash function maps a given element to a specific counter of the filter on a specific row. The matrix is initialized by setting all counters to zero. Then,  $h_i(\cdot)$  is evaluated for all  $\alpha \in D$ , and the counters  $CM_\alpha[i]$ ,  $1 \leq i \leq k$  (where  $CM_\alpha[i] = \{CM[i, j] : j = h_i(\alpha)\}$ ) are increased by 1. If the hash functions are independent, then the probability that two elements will hash to exactly the same set of counters is equal to  $\frac{1}{m^k}$ , which decreases rapidly with increasing numbers of  $k$  and  $m$ .

Given this structure, the estimated frequency of  $\alpha$  is given by the formula  $|\hat{\alpha}| = \min_i \{CM_\alpha[i]\}$ . Essentially, the minimum of any counter yields a frequency estimate that is the least affected by hash function collisions of other elements on the same counters. The CM sketch imposes specific constraints on  $k$  and  $m$ , in order to give tight guarantees on the estimation accuracy. In particular, it can be proved that for the CM sketch the following holds:

**Theorem 1.** *For a CM sketch with  $m = \lceil e/\epsilon \rceil$ ,  $k = \lceil \ln(1/\delta) \rceil$  and user defined constants  $\epsilon, \delta \in (0, 1]$ , let  $|\alpha|$  be the exact frequency of element  $\alpha \in D$ , and  $|\hat{\alpha}|$  be the estimated frequency computed using the sketch. Then,  $|\hat{\alpha}| \geq |\alpha|$ , and with probability  $1 - \delta : |\hat{\alpha}| \leq |\alpha| + \epsilon|D|$ .*

It can be shown that the CM sketch has the following properties:

**Property 1** (Union Property). The CM sketch of the union of two multi-sets is the simple matrix addition of their individual CM sketches. That is,  $CM(D_1 \cup D_2)[i, j] = CM(D_1)[i, j] + CM(D_2)[i, j]$ .

**Property 2** (Order Insensitivity). The CM sketch is order insensitive (Definition 3).  $CM(D)$  is entirely determined by the items of  $D$ . Ordering of insertions does not affect the structure.

### 3.2 CM-FM Sketches: Design and Theory

Apart from order-insensitivity, we would ideally like the CM sketch to be duplicate-insensitive in terms of the union operation. Notice that double-counting a certain CM sketch into a union operation many times, will yield errors in estimated frequencies that grow with the total number of duplicate additions that occurred. A similar effect occurs when double-counting

elements that have been inserted across different CM sketches.

We propose to make the CM sketch duplicate-insensitive by replacing every matrix counter with an FM sketch, and take advantage of the unbiasedness of these sketches. Essentially, we use the FM sketches to estimate the magnitude of each counter, i.e., to estimate how many elements have been added to the specific FM sketch. We name this construction the CM-FM sketch, for which the following hold:

**Property 3** (Union Property). The CM-FM sketch of the union of two multi-sets is the simple matrix addition of their individual CM-FM sketches, where each element addition is given by the bit-wise OR of the corresponding FM sketches. That is,  $CM - FM(D_1 \cup D_2)[i, j] = CM - FM(D_1)[i, j] \vee CM - FM(D_2)[i, j]$ .

**Property 4** (Duplicate Insensitivity). The CM-FM sketch is duplicate-insensitive (Definition 4).

**Property 5** (Unbiasedness). The CM-FM sketch is unbiased with respect to the master sketch (Definition 1).

*Proof.* (Sketch) It follows directly from the unbiasedness of the FM sketches, Properties 3 and 4 of CM-FM, and the fact that the estimates produced by the CM-FM are given using the simple *min* estimation function. Each FM counter — including the one with the minimum estimation — produces an unbiased estimation of the total frequency of the elements that have been hashed on it, irrespective of the insertion sequence and union operations performed on the CM-FM.  $\square$

The first important observation for CM-FM sketches is that we need to construct an appropriate function for estimating the frequency of value  $\alpha$  using the sketch. The second observation is that we need to theoretically derive the error bounds for the estimated frequencies, based on the individual errors introduced both by the FM sketch approximations and the CM sketch hash function collisions.

**Theorem 2.** *For a CM-FM sketch with  $m = \lceil (1 + \epsilon_f)e/\epsilon_c \rceil$ ,  $k = \lceil \ln(1/\delta_c) \rceil$  and user defined constants  $\epsilon_f, \epsilon_c, \delta_f, \delta_c$ , let  $|\alpha|$  be the exact frequency of element  $\alpha \in D$ , and  $|\hat{\alpha}|$  be the estimated frequency computed using the sketch. Then, with probability at least  $(1 - \delta_f)^k : |\hat{\alpha}| \geq (1 - \epsilon_f)|\alpha|$  and with probability at least  $(1 - \delta_f)^k(1 - \delta_c) : |\hat{\alpha}| \leq (1 + \epsilon_f)|\alpha| + \epsilon_c|D|$ .*

*Proof.* The lower bound can be proven as follows. Let  $D_\alpha[i]$  denote the set of elements that hash to exactly the same FM sketch as element  $\alpha$  on row  $i$ . Then,  $|D_\alpha[i]|$  quantifies the total number of insertions to FM sketch  $CM_\alpha[i]$ , which implies that by construction the FM sketch  $CM_\alpha[i]$  gives the estimate  $|\widehat{D}_\alpha[i]|$ . Also, let  $|\alpha|$  denote the total number of insertions of  $\alpha$  in

the sketch. Clearly,  $|D_\alpha[i]|$  accounts for the  $|\alpha|$  insertions plus the collisions of other elements to the same FM. Let the estimation accuracy of each individual FM sketch be bounded by:

$$(1 - \epsilon_f)|D_\alpha[i]| \leq |\widehat{D_\alpha[i]}| \leq (1 + \epsilon_f)|D_\alpha[i]|$$

with probability  $1 - \delta_f$  [15]. Then,

$$\begin{aligned} CM_\alpha[i] &\geq (1 - \epsilon_f)|D_\alpha[i]| \\ &\geq (1 - \epsilon_f)|\alpha| \end{aligned}$$

with probability  $1 - \delta_f$ . By using:

$$|\widehat{\alpha}| = \min_{i=1}^k \{CM_\alpha[i]\}$$

as the estimation function, the lower bound follows. Notice that the lower bound succeeds with probability  $(1 - \delta_f)^k$ , i.e., the probability that the estimates of all FM sketches will be correct.

To prove the upper bound, assuming perfect hash functions it holds that:

$$P = Pr[h_i(\alpha) = h_i(\beta)] \leq \frac{1}{m}$$

Let  $X_\alpha[i]$  denote the total number of collisions of elements other than  $\alpha$  on  $CM_\alpha[i]$ . Then:

$$E\{X_\alpha[i]\} = \sum_{\beta \neq \alpha} P \cdot |\beta| \Rightarrow E\{X_\alpha[i]\} \leq \frac{1}{m}|D|$$

Thus, by using  $Pr[A > B] \leq Pr[A' > B]$ , for  $A < A'$ , and Markov's inequality:

$$\begin{aligned} Pr[|\widehat{\alpha}| > (1 + \epsilon_f)|\alpha| + \epsilon_c|D|] &= \\ Pr[\forall i, CM_\alpha[i] > (1 + \epsilon_f)|\alpha| + \epsilon_c|D|] &= \\ Pr[\forall i, |\widehat{D_\alpha[i]}| > (1 + \epsilon_f)|\alpha| + \epsilon_c|D|] &\leq \\ Pr[\forall i, (1 + \epsilon_f)|D_\alpha[i]| > (1 + \epsilon_f)|\alpha| + \epsilon_c|D|] &\leq \\ Pr[\forall i, (1 + \epsilon_f)(|\alpha| + X_\alpha[i]) > (1 + \epsilon_f)|\alpha| + \epsilon_c m E\{X_\alpha[i]\}] &= \\ Pr[\forall i, X_\alpha[i] > \frac{\epsilon_c}{1 + \epsilon_f} m E\{X_\alpha[i]\}] &\leq \left(\frac{1 + \epsilon_f}{\epsilon_c m}\right)^k \end{aligned}$$

Now, by setting  $e = \frac{\epsilon_c m}{1 + \epsilon_f}$ , we get:

$$Pr[|\widehat{\alpha}| \leq (1 + \epsilon_f)|\alpha| + \epsilon_c|D|] > 1 - e^{-k}$$

Thus,  $\delta_c = e^{-k} \Rightarrow k = \ln(1/\delta_c)$ . The above holds only if none of the FM sketches fails, thus the estimation will succeed with probability  $(1 - \delta_f)^k(1 - \delta_c)$ .  $\square$

Since the success of the CM-FM sketch depends heavily on the success of the individual FM sketches, for the given analysis, it is essential to guarantee that the failure probability of FM sketches is low, in order

to limit the factor of the exponent  $k$ . In practice, for a failure probability  $\delta_c = 1\%$  the CM sketch requires  $k = \lceil \ln(1/0.01) \rceil = 5$  hash functions. If the FM sketches fail with probability  $\delta_f = 1\%$  as well, then the total probability of failure for the CM-FM sketch, from Theorem 2, is 5% and 6% for the lower and upper bound respectively. With a somewhat more complicated analysis we can give better guarantees for the estimates of the CM-FM sketch:

**Claim 3.** *We can guarantee that by using the  $\gamma$ -th order statistic of the  $k$  FM sketch estimations  $|\widehat{D_\alpha[i]}|$  instead of the min, where  $\gamma = k\delta_f + 4 \ln \frac{1}{\delta_c}$ , the FM estimation that we pick is within  $\pm \epsilon_f |D_\alpha[\gamma]|$  with probability at least  $\delta_c$ .*

*Proof.* Assume once more that an individual FM sketch can guarantee that with probability  $1 - \delta_f$ :

$$(1 - \epsilon_f)|D_\alpha[i]| \leq |\widehat{D_\alpha[i]}| \leq (1 + \epsilon_f)|D_\alpha[i]|$$

Denote with  $X_j$  the r.v.  $Pr[|\widehat{D_\alpha[i]}| - |D_\alpha[i]| > \epsilon_f |D_\alpha[i]|]$ , which will occur with probability of success  $p = \delta_f$ . Let  $X = \sum_j X_j$  and consider the  $k$  Bernoulli trials, over the  $k$  FM sketches. Using Chernoff bounds:

$$Pr[X > \gamma] \leq e^{-\frac{\lambda^2 \mu}{4}}$$

for  $\gamma = (1 + \lambda)\mu$ . We would like the probability of  $X$  succeeding more than  $\gamma$  times to be smaller than  $\delta_c$ . Then, by picking the  $\gamma$ -th order statistic of the sorted sequence of  $|\widehat{D_\alpha[i]}|$ , we will know that it must be a successful FM estimation within interval  $\pm \epsilon_f |D_\alpha[\gamma]|$ , since at least  $k - \gamma$  FMs are within these bounds with probability higher than  $\delta_c$ . Thus:

$$e^{-\frac{\lambda^2 \mu}{4}} = \delta_c \Rightarrow \lambda = \pm \frac{4}{\mu} \ln \frac{1}{\delta_c}$$

Now, since  $\mu = kp = k\delta_f$ , we get:

$$\gamma = (1 + \lambda)\mu = \left(1 + \frac{4}{\mu} \ln \frac{1}{\delta_c}\right) \mu = k\delta_f + 4 \ln \frac{1}{\delta_c} \quad (1)$$

subject to  $\gamma \leq k$ .

Hence, for given  $\delta_f, \delta_c$  and  $k$ , from equation (1) we can compute the  $\gamma$ -th order statistic of the sorted sequence of the  $k$  FM sketch estimations that we need to pick, in order to guarantee that the probability of the  $\gamma$  smallest of them failing all at the same time is less than  $\delta_c$ .  $\square$

Given Claim 3, we can prove the following:

**Theorem 4.** *For a CM-FM sketch with given  $m, k, \delta_f, \delta_c, \epsilon_f, \epsilon_c$ , we can guarantee that by using the  $\gamma$ -th order statistic as the estimation function, where  $\gamma = k\delta_f + 4 \ln 1/\delta_c, \gamma < k$ , with probability at least*

$1 - \delta_c : |\hat{\alpha}| \geq (1 - \epsilon_f)|\alpha|$  and with probability at least  $(1 - \delta_c)^2 : |\hat{\alpha}| \leq (1 + \epsilon_f)|\alpha| + \epsilon_c|D|$ .

*Proof.* The proof follows a similar reasoning with the proof of Theorem 2 and is omitted.  $\square$

## 4 Distributed Quantile Digests

The Quantile Digest (QD) [29] is an efficient  $\epsilon$ -approximation technique for estimating arbitrary quantiles over data produced by distributed sources. The idea is to maintain a small summary of the data at each source, and to be able to combine individual summaries in order to answer quantile queries on the union of the data available to the corresponding sources, with relative error at most  $\epsilon$ .

### 4.1 Basics

Given a multi-set with values in the range  $[m]$  and a fraction  $q \in (0, 1)$  the  $q$ -th quantile of  $D$  is the value with rank  $q|D|$  in the sorted sequence of values in  $D$  (the MEDIAN is a special quantile, with  $q = 0.5$ ). If  $D$  is known in full, then quantile computation can be done very efficiently; e.g., the  $O(|D|)$  algorithm of Floyd and Rivest [14] can be used. On the other hand, if  $D$  is distributed over a number of sources, or if the size of  $D$  is too large to be retained in full (e.g., in streaming applications), then exact computation of quantiles is not an easy problem. In the first case, special distributed algorithms need to be used [21]. In the second, exact quantile computation is not feasible at all [27]. For these reasons, various approximation algorithms that work by summarizing the data appropriately have been proposed for quantile computation. We say that an algorithm computes an  $\epsilon$ -approximate quantile, if the estimated answer has relative error  $\epsilon = |\hat{q} - q|/|D|$ , where  $\hat{q}$  is the estimated rank of the quantile, and  $q$  is the actual rank in  $D$ .

QD uses a predefined binary tree structure to capture the distribution of values in  $D$  over the domain  $[1, m]$  (where  $D$  from now on denotes the multi-set of data available at a specific source). Every node of the tree is associated with a tuple  $\langle N, R_N, C_N \rangle$ , where  $N$  is a unique node identifier,  $R_N$  is a range of values from  $[m]$  (where  $R_N^{min}, R_N^{max}$  will be used to denote the upper and lower bound of the range respectively) and  $C_N$  is a tally of the number of elements of  $D$  with values in  $R_N$ . Values are assigned to leaves in a sorted order from left to right. Nodes in higher levels of the tree correspond to dyadic ranges over the values of the lower levels. The root of the tree corresponds to the whole domain. In addition, node identifiers are assigned by following a sequence that is equivalent with a post-order traversal of the tree. An example is shown in Figure 2(a), for the domain of values  $[1, 8]$ . The identifiers of the nodes and the frequency of each value are also shown.

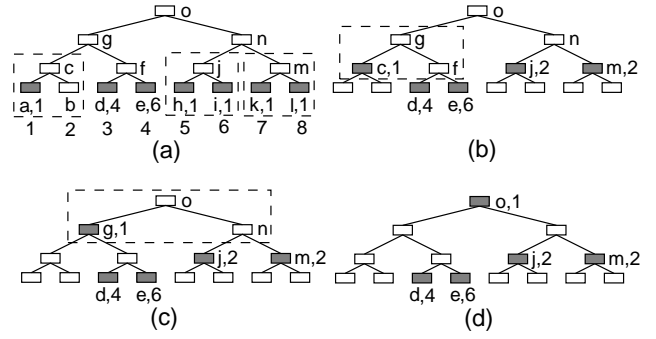


Figure 2: Building a quantile digest. The complete conceptual tree is shown here. Gray nodes correspond to the actual sketch.

This binary tree is *conceptually* maintained locally by all the sources. In practice, every source needs to instantiate only a subset of the tree nodes at a time, and transmit this subset to the destination. The subset constitutes the actual QD sketch. To decrease the size of the sketch we need to choose which nodes to instantiate and include in the QD. This is governed by the following two rules:

$$C_N \leq \lfloor |D|/k \rfloor, \text{ if } N \text{ is not a leaf} \quad (2)$$

$$C_N + C_{N_p} + C_{N_s} > \lfloor |D|/k \rfloor \quad (3)$$

where  $N, N_p$  and  $N_s$  are a node, the node's parent and its sibling, respectively, and  $k$  is a user defined parameter that controls the accuracy and the compression ratio of the sketch. Equation (2) can be used to guarantee special error bounds on QD. Equation (3) implies that if two adjacent leaves have a low count, they are compressed to save space by preserving only their parent.

A QD is constructed as follows. First, we compute the frequency  $C_N$  of every leaf. We do not instantiate any index nodes, but we assume that for these nodes  $C_N = 0$ . Next, we propagate values up the tree as follows: if a leaf and its sibling violate equation (3) we merge the two leaves, instantiate their parent, set  $C_{N_p} = C_N + C_{N_s}$ , and discard  $N, N_s$ . We continue the same process left to right and bottom to top, until no more nodes can be merged. At the end, all nodes that have not been discarded constitute the QD. An example is shown in Figures 2(a)-(d). The quantile digest at each step of the compression algorithm consists of the gray nodes. The final QD sketch is the following:  $\langle d, 4 \rangle, \langle e, 6 \rangle, \langle j, 2 \rangle, \langle m, 2 \rangle, \langle o, 1 \rangle$ .

Merging QDs is straightforward. Starting from the complete conceptual binary tree with all counters initialized to 0, incoming QD nodes from other sources are added to their respective counters in the tree. Then, equations (2) and (3) are applied on the resulting tree, where  $|D| = |D_1| + |D_2| + \dots$  is the cardinality of the union of the sets corresponding to the

incoming QDs.<sup>2</sup> Answering quantile queries on any QD requires first sorting the nodes in increasing identifier order. Notice that since identifiers are assigned in a post-order fashion, sorting the nodes by identifier corresponds to sorting them in increasing order of the upper bound of their range, breaking ties with smaller ranges first. Now, given  $q \in (0, 1)$ , we start adding node counters from left to right in the sorted order, until the sum becomes larger than  $q|D|$ , after some node  $N$ , at which point we know that at least  $q|D|$  values are included in the range  $[1, R_N^{max}]$ . We report as the  $q$ -th quantile of  $D$  the value  $R_N^{max}$ . In the previous example the computation yields node  $e$  with value 4 as the estimated MEDIAN.

The following hold for quantile digests [29]:

**Lemma 5.** *For a QD with compression factor  $k$ , the maximum error in  $C_N$  for any node is  $\frac{\log_2(m)}{k}|D|$ .*

**Theorem 6.** *A QD  $Q$  can answer any quantile query with error  $\epsilon_{qd}$  such that  $\epsilon_{qd} \leq \frac{3 \log_2(m)}{|Q|}$ , where  $|Q|$ , the number of nodes in the digest, satisfies  $|Q| < 3k$ , given compression factor  $k$ .*

**Property 6** (Union Property). Given  $n$  QDs  $QD_1, \dots, QD_n$  built on streams of values  $D_1, \dots, D_n$ , each with maximum relative error  $\epsilon_{qd}$ , the merging algorithm combines them into a QD for  $D_U = D_1 \cup \dots \cup D_n$  with the same relative error.

## 4.2 Insensitivity and Bounding the Error

The QD sketch is clearly not duplicate-insensitive. For example, merging a  $QD(D)$  with itself produces a new QD for the larger multi-set  $D \cup D$ . However, we can again utilize FM sketches to make a combined QD-FM sketch that is duplicate-insensitive. In this case, first we build a normal QD as described above. Then, we replace all tallies  $C_N$  in the retained nodes, with an equivalent FM sketch that estimates the magnitude of  $C_N$ . Now if duplicate elements are inserted in the FM sketches across sources, they will be counted only once. In addition, merging a QD-FM sketch with itself produces the original QD-FM sketch. Therefore we have:

**Property 7** (Duplicate Insensitivity). The QD-FM sketch is duplicate-insensitive (Definition 4).

Let a QD-FM sketch  $\{\langle A, FM_A \rangle, \langle B, FM_B \rangle, \dots\}$ , where each  $FM_N$  is an estimator for  $C_N$ , with  $(1 - \epsilon_f)C_N \leq \widehat{C}_N \leq (1 + \epsilon_f)C_N$ . To find the  $q$ -th quantile of  $D$ , we iteratively combine FM sketches from left to right (in the sorted identifier order) using the OR operation on the bit vectors, until  $\widehat{C}_{A \cup \dots \cup N} \geq q|D|$  (where

<sup>2</sup>A detail of QD sketches is that during the merging operation we also need to know the cardinality of the set on which each sketch was built, in order to obtain the cardinality for the combined sketch. This value needs to be transmitted with the rest of the sketch.

$\widehat{C}_A \cup \dots \cup N$  is the estimation produced by the combined FM sketch  $FM_A \cup \dots \cup FM_N$ ). We report as the  $q$ -th quantile of  $D$ , the value  $R_N^{max}$ .

Having established the quantile estimation function using QD-FM sketches, we now turn our attention to the error introduced during the merging operation, that will quantify the total error per node counter for the QD-FM sketch, as in Lemma 5. Then, we will show how to bound the error on the quantile estimation. During the merge operation, we re-compress the conceptual binary tree, after combining individual FMs per node, using compression factor  $k$ . In order to account for the estimation errors introduced by the FMs we need to modify Equations (2) and (3) as follows:

$$C_N \leq \lfloor |D|/k \rfloor (1 + \epsilon_f), \text{ if } N \text{ is not a leaf} \quad (4)$$

$$C_N + C_{N_p} + C_{N_s} > \lfloor |D|/k \rfloor (1 + \epsilon_f) \quad (5)$$

We can show that by using this pessimistic compression during merging, the total error introduced by the FMs in the worst case is offset by a somewhat larger sketch, larger by a factor of  $1 + \epsilon_f$ . Using the new compression formulas we can show that the total error in the count of every node is at most the one reported by Lemma 5. The proof follows a similar reasoning with the proof appearing in [29]. Thus, the FM sketch approximations do not introduce any error, after merging a number of QD-FM sketches. We can also claim that the QD-FM sketch has the Union Property (Property 6), as well. Since the QD sketch is not order insensitive, we cannot claim that QD-FM has is unbiased with respect to a master sketch. Nevertheless, given Lemma 5 and Property 6 (both recast for QD-FM), the following holds:

**Property 8** (Bounded Error). The QD-FM has bounded error with respect to the master sketch (Definition 2).

Given the above, we can claim the following:

**Theorem 7.** *Given a QD-FM sketch on  $D$  and user defined parameters  $\epsilon_{qd}, \epsilon_f, \delta_f$ , let  $q$  be the exact rank of the  $q$ -th quantile in  $D$  and  $\widehat{q}$  be the estimated rank using the sketch. Then, with probability at least  $1 - \delta_f$  :  $\frac{q}{1 + \epsilon_f} \leq \widehat{q} \leq \frac{q + \epsilon_{qd}}{1 - \epsilon_f}$ .*

*Proof.* To prove the lower bound, by construction of the estimation function  $\widehat{C}_N \geq q|D|$  for some node  $N$ , where  $\widehat{C}_N$  is the FM estimation for the actual sum  $C_N$ , corresponding to node  $N$  in the original QD. By definition  $\widehat{C}_N \leq (1 + \epsilon_f)C_N$  with probability  $1 - \delta_f$ , which yields  $(1 + \epsilon_f)C_N \geq q|D| \Rightarrow C_N \geq \frac{q}{1 + \epsilon_f}|D|$ . Hence, in the worst case we know that at least  $\frac{q}{1 + \epsilon_f}|D|$  values lie in the range  $[1, R_N^{max}]$ , which implies that  $R_N^{max}$  is at least the  $\frac{q}{1 + \epsilon_f}$ -th quantile.

To prove the upper bound, we know that in the worst case, the total error of the sum up to any node

is at most  $\epsilon_{qd}|D|$  (i.e., the number of values lying in the range  $[1, R_N^{max}]$  that have not been accounted for from ancestors of  $N$  not included in the sum, as Theorem 6 implies [29]). Given that  $\widehat{C}_N \leq (1 + \epsilon_f)C_N$ , with probability  $1 - \delta_f$  and  $\widehat{C}_N \leq q|D| + \epsilon_{qd}|D|$ , we get  $C_N \leq \frac{q + \epsilon_{qd}}{1 - \epsilon_f}|D|$ , meaning that  $R_N^{max}$  is at most the  $\frac{q + \epsilon_{qd}}{1 - \epsilon_f}$ -th quantile.  $\square$

## 5 Discussion

### 5.1 CM-FM and QD-FM Sketches in Practice

Depending on the application, various construction schemes for duplicate insensitive sketches can be employed. Below, we give two simple examples to illustrate the generality of the proposed framework, using the CM-FM. Similar claims hold for the QD-FM sketch.

In the first example we would like to answer *frequency estimation queries* on a sensor network. That is, given a sensor field with sensors that produce streams of values from a known domain, we would like to be able to know at the base station the total number of times that any value  $\alpha$  has been reported by all the sensors (a single value might be produced multiple times by the same sensor, and this should be accounted for in the frequency of that value on the union of streams). Every sensor maintains locally a sketch of its values. First, a frequency query is flooded on the network, then all sensors transmit their sketches to the base station. After the sketches are merged, we can estimate the frequency of any value using the combined sketch. Since multi-path routing protocols might be in use, we need to make sure that the sketches we use are duplicate-insensitive to double counting of duplicate sketch instances. This can be accomplished using CM-FM sketches as follows: Let a CM-FM matrix of a specific sensor. In order to estimate the total frequency of value  $\alpha$  on that sensor, we need to treat subsequent insertions of  $\alpha$  into its corresponding FM sketch uniquely, no matter if the element has been inserted before. As such, we produce a unique identifier per insertion of  $\alpha$ , based on its identifier and the identifier of the sensor.<sup>3</sup> That way, the FM sketches produced by a specific sensor are unique and, in addition, they estimate the total number of times that each element  $\alpha$  has appeared locally. Merging FM sketches from the same sensor has no effect, while merging FM sketches from distinct sensors produces a sketch that can estimate the frequency of  $\alpha$  in the union of the sensors' data. The resulting CM-FM sketch has all the desired properties.

<sup>3</sup>For example, we can maintain a global counter  $C$  of the total number of insertions so far per sensor, and use as the hashing key the tuple  $\langle sid, \alpha, C \rangle$ . This tuple is guaranteed to be unique for every insertion of value  $\alpha$ , across all FM sketches and across all sensors.

As a second example, consider a number of core routers on the Internet, that are used to monitor heavy-hitting flows. In this application, we would like to be able to *estimate the size of a flow* by combining information from multiple routers. The duplicates problem that arises in this case is not on a per sketch basis as before, but per IP packet. That is, we can guarantee that every sketch is transmitted only once, although we cannot guarantee that a specific packet has been observed only by a single router. Essentially, packets that went through a large number of monitoring routers will be double-counted if the information from all routers is combined without care (e.g., by using simple CM sketches). In order to use the CM-FM sketch in this scenario, given a packet with a unique identifier that belongs to a specific OD flow, first we hash on the CM matrix using the flow identifier (i.e., every flow hashes to a specific set of FM sketches), then, we hash on the FM sketches using the packet's unique identifier. This insertion policy guarantees that all OD flows hash to the same FM sketches across routers, and that each packet is hashed to its corresponding FM sketch with the same identifier across all CM-FM sketches. When the FM sketches of a specific flow are merged, we know that packet double-counting has been eliminated due to the duplicate-insensitivity property of the FM sketches.

### 5.2 Generalizations to Other Techniques

The applications of the duplicate-insensitive FM estimator to CM and QD sketches are basic examples that illustrate the applicability of our framework. The same reasoning can be applied to other applications and sketching techniques as well. Examples include: counting sampling [18], Bloom filters [6], and group testing [8] among others. Nevertheless, there are other techniques that are not amenable to our framework due to certain limitations. One such example is the well known AMS sketch [1]. The main problem is that when we construct the AMS sketch, we not only add but also *subtract* values from a set of counters. So, the counters are not always positive, but they can also be negative or zero. One, may think of using two FM sketches for each counter of the original AMS sketch; one to approximate the number of additions and the other to approximate the number of subtractions. However, even though each individual FM sketch is a good estimator, the *difference* of the estimated values of these FM sketches is *not guaranteed* to be a good estimate of the original value. The relative error in that case can be arbitrarily large. This means that if we use the above method the error of the AMS sketch estimator cannot be bounded. Other techniques have similar problems, but further analysis is beyond the scope of this article and is left as future work.

## 6 Empirical Studies

In this section we present the results of an empirical study conducted on our CM-FM and QD-FM sketches. We study their average case behavior over a number of synthetic data streams. We compare the techniques with their original counterparts (CM and QD sketches, respectively), to quantify the effect of FM sketch approximations on estimation accuracy. In addition, we evaluate the union of sketches approach against the master sketch approach, as defined in Section 2, to illustrate the unbiasedness of the CM-FM sketch, and the bounded error of the QD-FM sketch. The results substantiate our claims that with limited space increase (compared to their original counterparts) various sketching techniques can apply in a distributed setting and can be made order- and duplicate-insensitive while still providing accurate answers.

### 6.1 Experimental Testbed

In our experiments we use synthetic data streams with skewed distributions. The values are drawn from a domain of  $10^6$  integers, with Zipf parameter  $z = 1.5$ . Each data stream consists of 500,000 values. To quantify the accuracy of our sketches, we utilize the conventional relative error metric: given an estimate  $\hat{f}$  and the real answer  $f$ , the error of the estimate is defined as the ratio  $\frac{|\hat{f}-f|}{f}$ . To build the union of sketches we generate  $n$  streams and  $n$  individual sketches are build. Then, these sketches are merged in order to produce the union of sketches. The master sketch approach builds a single sketch on the complete data stream, after the individual  $n$  streams have been merged. Finally, we evaluate both approaches on the same queries, using the exact answers obtained from a histogram to test the accuracy of the techniques.

### 6.2 CM-FM Sketches

For the CM-FM sketches (which are used for frequency estimation queries), using an exact histogram, we first identify the top 10% of the most frequent values. Then, we query the CM and CM-FM sketches, retrieve the frequency estimates for these values, and compute the average frequency error. Finally, we run the same experiment 10 times, with different random seeds, and report in our plots the average relative error over all runs. During each run we maintain a CM sketch, a CM-FM master sketch, a CM-FM union of sketches, and the exact histogram, incrementally. For simplicity in our experiments, we use the *min* estimator for the CM and CM-FM sketches (see Section 3), rather than using quantiles as we needed to do to push through the analysis. In practice, we did not observe significant differences between these approaches. The CM-FM master sketch (CM-FM-M in the figures) is useful for evaluating the unbiasedness of the union of sketches (CM-FM-U). In practice, data sources need

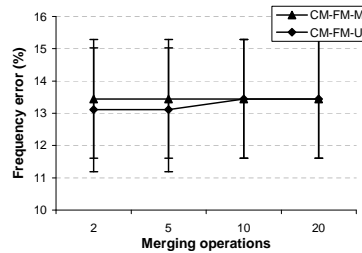


Figure 3: CM and CM-FM unbiasedness vs. Number of merging operations ( $k = 3, m = 200, B = 16, BV = 30$ )

only maintain normal CM sketches locally, and convert them to CM-FM sketches before sketch transmission.

To test the accuracy of the techniques we repeat the experiments multiple times, and report the estimation errors as a function of space required by the sketches. The size of the CM sketch is affected only by parameters  $k$  (the number of hash functions, i.e., the number of rows in the CM matrix) and  $m$  (the range of values per hash function, i.e., the number of columns in the CM matrix). For the CM-FM sketch, two more parameters affect the size of the sketch: the number of bits  $B$  and bit vectors  $BV$  per FM sketch.

In Figure 3 we empirically validate the unbiasedness of the CM-FM sketches. In every run, we generate a number of streams, build the individual sketches, and merge those sketches at the aggregation point. The larger the number of streams, the more the merging operations that need to be performed. To build the master sketch, first we get the union of streams and build one sketch on the whole data. The number of merges for the union of sketches does not affect the error guarantees of the results, and thus, we empirically observe the unbiasedness of CM-FM sketches relative to the master sketch. The small variations in the results can be attributed to different hashing keys used during each run for performing insertions in the FM sketches (every sub-stream is assigned a unique identifier and in order to make FM insertions unique throughout all sketches, we utilize these identifiers).

In Figure 4 we plot the frequency error as a function of the number of hash functions  $k$ . For this experiment we use a  $k \times 200$  matrix (i.e., each row has 200 counters for CM or 200 FM sketches for CM-FM). For the FM sketch estimators we use  $BV = 30$  bit vectors, and  $B = 16$  bits. The total size of the sketches can be computed easily, as the number of rows times the number of columns times the size of the individual counters (or FM sketches). Finally, for the union of sketches we divide the stream into 5 disjoint sub-streams (the same default value will be used in the rest of the experiments as well).

Straightforwardly, for CM and CM-FM sketches the probability of low frequent values colliding with highly frequent values on all hash function decreases as the number of hash functions increases. Hence, we expect an improvement in estimation accuracy, espe-

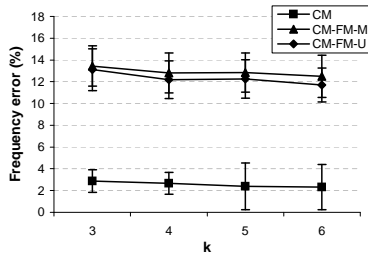


Figure 4: CM and CM-FM frequency error vs. Number of hash functions ( $m = 200, B = 16, BV = 30$ )

cially for large value domains where the sketches become very densely populated. For our datasets, from this graph we observe that as few as 3 hash functions provides 2.8% estimation accuracy for CM and 12% accuracy for CM-FM sketches. Increasing the number of hash functions slightly improves accuracy for the CM sketch, and has no impact for CM-FM sketches. The reason is that the estimation error introduced by the FM approximations subsume the gain obtained by the larger matrix.

In general, the FM sketch approximations increase the final estimation error by at most three times, for a CM-FM sketch that is 10 times larger than the equivalent CM sketch, taking into account the  $1/3$  compression ratio of FM sketches, achieved with the technique described in [7]. Notice here that the FM sketch algorithm introduces by itself, on average, 10% to 20% error in the count estimates. Hence, a three-fold increase in the error of the CM sketch is an excellent compromise when using FM sketches to provide duplicate-insensitivity, especially when no other alternatives have been proposed yet.

The next experiment tests the improvement in estimation accuracy as a function of  $m$ , the number of counters (or FM sketches) per row. Figure 5 plots the results. Once again, we expect accuracy to improve as the number of counters increases due to a smaller number of collisions. The results are qualitatively similar to the previous experiment. Indeed, the CM sketch demonstrates substantial improvement. The CM-FM sketch benefits significantly from an initial increase in the number of FM sketches per hash function, but for larger numbers, the FM approximation errors again subsume the improvements. Finally, the master sketch and union of sketches provide largely equivalent results.

Figure 6 plots the frequency error of the CM-FM sketch with a varying number of bit vectors per FM sketch. Larger FM sketches (and hence increased space requirements) will yield improved estimation accuracy. Clearly, we can see that the error drops from 20% with 10 bitvectors to 12% with 40 bitvectors. The trend of the plot implies that a larger number of bit vectors will not provide any further improvements; with 40 bit vectors we reach the maximum estimation accuracy offered by the FM sketch.

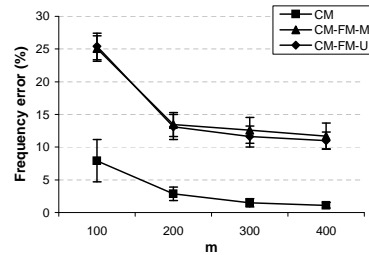


Figure 5: CM and CM-FM frequency error vs. Number of counters per hash function ( $k = 3, B = 16, BV = 30, z = 1.5$ )

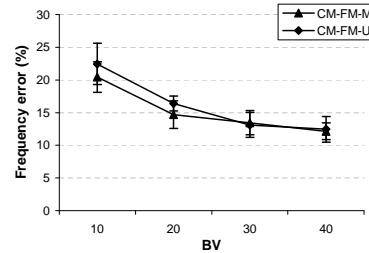


Figure 6: CM-FM frequency error vs. Number of bit vectors per FM sketch ( $k = 3, m = 200, B = 16, z = 1.5$ )

It is clear that the theoretical analysis of CM-FM sketches is pessimistic compared to the actual accuracy achieved for skewed datasets. In general, the CM-FM sketch achieved the empirical best approximation possible, between 10-20% error for FM sketches, as has been reported in related literature [13, 16, 7]. Our evaluation shows empirically the unbiasedness of the CM-FM sketch to the union operation, a useful property for aggregation.

### 6.3 QD-FM Sketches

For the QD-FM sketches (which are used for quantile estimation queries), using an exact histogram we first identify the MEDIAN of the union of data streams. After building and merging the individual QD-FM sketches we also compute the estimated MEDIAN and calculate the rank error: Given the true rank of the MEDIAN  $n/2$  and the estimated rank  $\hat{r}$ , the relative error is defined as the ratio  $\frac{|\hat{r}-n/2|}{n}$  [29]. We perform 10 independent runs, with different random seeds, and report in our graphs the average MEDIAN rank error. We use skewed datasets with a default Zipf parameter  $z = 1.5$ . For every run, we also maintain a master QD-FM sketch on the union of the streams, in order to evaluate the error bounds of the QD-FM union sketch.

To test the accuracy of the techniques we run the experiments multiple times with various sketch sizes. The size of the QD sketch depends only on compression factor  $k$ . The size of the QD-FM sketch depends on the number of bits  $B$  and bit vectors  $BV$  used per FM sketch, as well. The actual size of the sketches, in number of nodes that need to be retained after compression, can be estimated to be at most  $3 \cdot k(1 + \epsilon_f)$

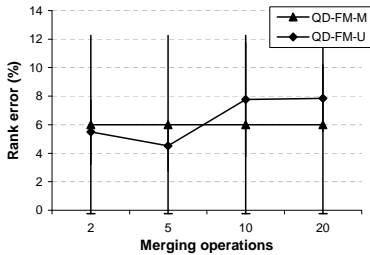


Figure 7: QD-FM sketch rank error vs. Number of merging operations ( $k = 6, BV = 30, B = 16$ ).

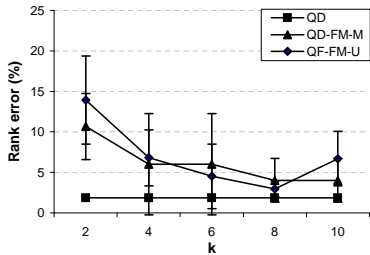


Figure 8: QD-FM sketch rank error vs. Compression factor  $k$  ( $BV = 30, B = 16$ ).

in the worst case. The size of one node is equal to two integers for the QD sketch (a node identifier and a count) and one integer plus the size of the FM estimator for the QD-FM sketch. For fairness, we utilize the same space constraints for both the QD and the QD-FM sketch, ignoring for now the compression diminishing factor  $1 + \epsilon_f$ . We will see that in practice, even if we fix the space requirements of the QD-FM sketch, the error bounds will still be comparable to the master sketch.

In Figure 7 we test the error bound property of the QD-FM sketch. We compute the rank error for the MEDIAN of a set of streams, where we build individual QD-FM sketches for each stream and get the union sketch at the end. The master sketch is obtained by first getting the union of streams. The graph empirically validates the theoretical results. The error of the QD-FM sketch is affected slightly by the total number of merging operations. Nevertheless, it always remains within the variance of the master sketch.

In Figure 8 we report the rank error (percentage) as a function of compression factor  $k$ . Larger values for  $k$  correspond to looser space constraints, and smaller compression ratios with respect to the total size of the binary tree. For a compression factor of 2 (at most 6 nodes in the worst case), the error of the QD sketch is as low as 2%, while the QD-FM sketch reports errors as high as 13%. Nevertheless, for larger  $k$  the error of QD sketch is unaffected, while the QD-FM sketch benefits significantly reaching the lowest reported error of 3% in the best case. In general, for a five-fold increase in size in the worst case, we pay a three-fold increase, on average, in MEDIAN rank errors, in order to achieve duplicate-insensitive functionality for quantile estimation.

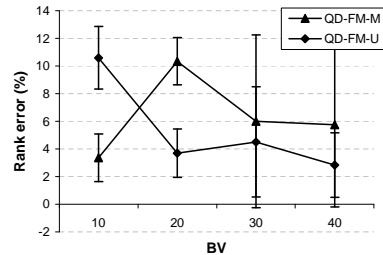


Figure 9: QD-FM sketch rank error vs. Number of bit vectors per FM sketch ( $k = 6, B = 16$ ).

Our last experiment shows the effect on rank errors as a function of FM sketch sizes and  $k = 6$ . Figure 9 plots the results. The general trend is that larger numbers of bit vectors benefit the QD-FM sketch. Although we still observe outliers, attributed to the inherent high variance of FM sketches. FM sketches generally do not significantly add to the error.

To conclude, empirical results for the QD-FM sketch corroborate our analysis. In addition, the utility of the QD-FM sketch is considerable, since it can answer quantile queries, inverse quantile queries, range frequency queries, and consensus queries, as discussed in [29].

## 7 Conclusion

The large body of recent work on one-pass sketching of data streams provides a powerful set of useful techniques for summarization of massive data sets. But in order to harness the full power of these methods in distributed settings involving multiple data streams, further investigation is needed to determine whether these methods are amenable to distributed aggregation. In this work, we specify a simple distributed model in which we assess hierarchical aggregation of sketched streams. Our work demonstrates that with some effort, basic sketching methodologies such as Count-Min sketches and Quantile Digests can be augmented so that they are suitable for use in distributed settings. Our evaluation is based along the standard dimension of space-accuracy trade-offs, the more recent benchmark of order- and duplicate-insensitivity, and in our new formulations of competitiveness against a so-called *master sketch*.

Our future work is centered on identification of sketch invariants and translation mechanisms that enable us to “automatically” generate versions suitable for distributed use. We are also interested in those sketching methodologies for which we do not see a natural translation to our distributed setting. Some of these methodologies run into difficulties when issues of order- and duplicate-insensitivity come into play; for others it is difficult to bound errors that grow during aggregation. Characterizing this boundary between usable and unusable in distributed settings is our ultimate objective.

## References

- [1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 20–29, 1996.
- [2] B. Babcock, M. Datar, and R. Motwani. Load shedding for aggregation queries over data streams. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 350–361, 2004.
- [3] B. Babcock and C. Olston. Distributed top-k monitoring. In *Proc. of ACM Management of Data (SIGMOD)*, pages 28–39, 2003.
- [4] Z. Bar-Yossef, T. S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *Proc. of International Workshop on Randomization and Approximation Techniques (RANDOM)*, pages 1–10, 2002.
- [5] M. Bawa, A. Gionis, H. Garcia-Molina, and R. Motwani. The price of validity in dynamic networks. In *Proc. of ACM Management of Data (SIGMOD)*, pages 515–526, 2004.
- [6] A. Broder and M. Mitzenmacher. Network applications of bloom filters: A survey. In *Annual Allerton Conference on Communication, Control, and Computing*, 2002.
- [7] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 449–461, 2004.
- [8] G. Cormode and S. Muthukrishnan. What’s hot and what’s not: tracking most frequent items dynamically. In *Proc. of ACM Symposium on Principles of Database Systems (PODS)*, pages 296–306, 2003.
- [9] G. Cormode and S. Muthukrishnan. Improved data stream summaries: The count-min sketch and its applications. *Journal of Algorithms*, 2004.
- [10] A. Das, S. Ganguly, M. Garofalakis, and R. Rastogi. Distributed set expression cardinality estimation. In *Proc. of Very Large Data Bases (VLDB)*, pages 312–323, 2004.
- [11] E. D. Demaine, A. López-Ortiz, and J. I. Munro. Frequency estimation of internet packet streams with limited space. In *Proceedings of the 10th Annual European Symposium on Algorithms*, pages 348–360, 2003.
- [12] A. Dobra, M. Garofalakis, J. Gehrke, and R. Rastogi. Processing complex aggregate queries over data streams. In *Proc. of ACM Management of Data (SIGMOD)*, pages 61–72, 2002.
- [13] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, 1985.
- [14] R. W. Floyd and R. L. Rivest. Expected time bounds for selection. *Communications of the ACM (CACM)*, 18(3):165–172, 1975.
- [15] S. Ganguly, M. Garofalakis, and R. Rastogi. Processing set expressions over continuous update streams. In *Proc. of ACM Management of Data (SIGMOD)*, pages 265–276, 2003.
- [16] S. Ganguly, M. Garofalakis, and R. Rastogi. Tracking set-expression cardinalities over continuous update streams. *The VLDB Journal*, 13(4):354–369, 2004.
- [17] P. Gibbons and S. Tirthapura. Estimating simple functions on the union of data streams. In *Proc. of ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 281–291, 2001.
- [18] P. B. Gibbons and Y. Matias. New sampling-based summary statistics for improving approximate query answers. In *Proc. of ACM Management of Data (SIGMOD)*, pages 331–342, 1998.
- [19] P. B. Gibbons and Y. Matias. Synopsis data structures for massive data sets. In *Proc. of Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 909–910, 1999.
- [20] A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss. Surfing wavelets on streams: One-pass summaries for approximate aggregate queries. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 79–88, 2001.
- [21] M. Greenwald and S. Khanna. Power-conserving computation of order-statistics over sensor networks. In *Proc. of ACM Symposium on Principles of Database Systems (PODS)*, pages 275–285, 2004.
- [22] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. *ACM SIGOPS Operating Systems Review*, 36(SI):131–146, 2002.
- [23] S. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *Proc. of ACM Management of Data (SIGMOD)*, pages 491–502, 2003.
- [24] A. Manjhi, V. Shkapenyuk, K. Dhamdhere, and C. Olston. Finding (recently) frequent items in distributed data streams. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 767–778, 2005.
- [25] G. S. Manku and R. Motwani. Approximate Frequency Counts over Data Streams. In *Proc. of Very Large Data Bases (VLDB)*, pages 346–357, 2002.
- [26] D. Moore, G. Voelker, and S. Savage. Inferring internet denial-of-service activity. In *Usenix Security Symposium*, 2001.
- [27] J. I. Munro and M. Paterson. Selection and sorting with limited storage. *Theoretical Computer Science*, 12:315–323, 1980.
- [28] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *SenSys ’04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 250–262, 2004.
- [29] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri. Medians and beyond: new aggregation techniques for sensor networks. In *Proc. of International Conference on Embedded Networked Sensor Systems (SenSys)*, pages 239–249, 2004.
- [30] Y. Yao and J. Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Record*, 3(31):9–18, 2002.

- [31] Y. Yao and J. Gehrke. Query processing in sensor networks. In *Proc. of Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.