Algorithms for Distributed Functional Monitoring

Graham Cormode
AT&T Labs
and
S. Muthukrishnan
Google Inc.
and
Ke Yi
Hong Kong University of Science and Technology

We study what we call *functional monitoring* problems. We have k players each receiving a stream of items, and communicating with a central coordinator. Let the multiset of items received by player i up until time t be $A_i(t)$. The coordinator's task is to monitor a given function f computed over the union of the inputs $\cup_i A_i(t)$, *continuously* at all times t. The goal is to minimize the number of bits communicated between the players and the coordinator. Of interest is the approximate version where the coordinator outputs 1 if $f \geq \tau$ and 0 if $f \leq (1-\epsilon)\tau$. This defines the (k, f, τ, ϵ) distributed functional monitoring problem. Functional monitoring problems are fundamental in distributed systems, in particular sensor networks, where we must minimize communication; they also connect to the well studied streaming model and communication complexity. Yet few formal bounds are known for functional monitoring.

We give upper and lower bounds for the (k, f, τ, ϵ) problem for some of the basic f's. In particular, we study the frequency moments F_p for $p = 0, 1, 2$. For F_0 and F_1, we obtain monitoring algorithms with costs almost the same as their one-shot computation algorithms. However, for F_2 the monitoring problem seems much harder. We give a carefully constructed multi-round algorithm that uses "sketch summaries" at multiple levels of details and solves the (k, F_2, τ, ϵ) problem with communication $\tilde{O}(k^2/\epsilon + k^{3/2}/\epsilon^3)$. Our algorithmic techniques are likely to be useful for other functional monitoring problems as well.

Categories and Subject Descriptors: F.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms: Algorithms, theory

Additional Key Words and Phrases: Distributed computing, functional monitoring

Ke Yi was supported in part by Hong Kong Direct Allocation Grant (DAG07/08).
A preliminary version of the paper appeared in the ACM-SIAM Symposium on Discrete Algorithms (SODA), 2008.
Authors' addresses: Graham Cormode, AT&T Labs, Florham Park, NJ; email: graham@research.att.com. S. Muthukrishnan, Google Inc., New York, NY; email: muthu@google.com. Ke Yi, HKUST, Clear Water Bay, Hong Kong, China; email: yike@cse.ust.hk

1. INTRODUCTION

We introduce the *distributed functional monitoring* problem with a simple example, SUM. Suppose we have two observers, Alice and Bob, who each see arrivals of items over time. At time t, Alice has a multiset $A(t)$ of items and Bob has $B(t)$. Both Alice and Bob have an individual two-way communication channel with Carole so that Carole can monitor $C(t) = |A(t)| + |B(t)|$. Our goal is to minimize the total communication with Carole; Alice and Bob do not communicate with each other, but up to a factor of 2, that is not a limitation. As stated, it is easy to see that all Alice or Bob can do is to send a bit whenever they each see a new item, and hence, communicating a total of $|A(t)| + |B(t)|$ bits trivially. Of interests is a relaxed version of the problem where, given ϵ, Carole's new task is to output 0 whenever $C(t) \leq (1-\epsilon)\tau$ and must output 1 when $C(t) \geq \tau$ for a threshold τ. Now the problem is nontrivial. For example, here are some possible communication procedures:

—[Coin Toss] Alice and Bob each flip a (possibly biased) coin upon the arrival of an item and send Carole one bit whenever the coin turns up heads.
—[Global] Alice and Bob know a rough estimate of $\Delta = \tau - C(t')$ from some prior time t', and each send a bit whenever the number of items they have received exceeds $\Delta/2$. Carole updates Alice and Bob with new estimates from time to time correspondingly.
—[Local] Alice and Bob each create a model for arrival times of items and communicate the model parameters to Carole; they send bits to summarize differences when their current data significantly differs from their models. If the sources are compressible, this can yield savings.

The question is: What is the (expected) communication cost of these procedures, and what is the optimal bound?

We study such functional monitoring problems more generally in which (a) there are $k \geq 2$ sites, and (b) we wish to monitor $C(t) = f(A_1(t) \cup \cdots \cup A_k(t))$ where $A_i(t)$ is the multiset of items collected at site i by time t, and f is a monotonically non-decreasing function in time. Note that the function f should depend only on the union of the multisets $A_i(t)$, and on which sites where the items have been received or the arrival order. There are two variants: in *threshold monitoring*, or simply *monitoring*, the goal is to determine when $C(t)$ (approximately) exceeds a threshold τ, as illustrated in the SUM problem above; in *value monitoring*, often also called *tracking*, we want to provide a good approximation to $C(t)$ at all times t. Note that for a single-valued function f, value monitoring directly solves threshold monitoring, and running $O(\frac{1}{\epsilon}\log T)$ instances of a threshold monitoring algorithm for thresholds $\tau = 1, (1+\epsilon), (1+\epsilon)^2, \ldots, T$ solves value monitoring with relative error $1+\epsilon$, assuming $1 \leq f \leq T$. So the two variants differ by at most a factor of $O(\frac{1}{\epsilon}\log T)$. In many applications, the threshold version is more important, so we focus on this case and we call such a problem (k, f, τ, ϵ) *distributed functional monitoring*. Our interests in these problems come from both applied and foundational concerns.

Applied motivations. (k, f, τ, ϵ) functional monitoring problems arise immediately in distributed monitoring systems. For example, in a sensor network, sensors are distributed to monitor the environment and detect certain events. The straight-

forward way is to take measurements periodically, send them to a central site, and use back-end systems to analyze the entire data trace. However, in many modern sensor networks applications, the sensors are distributed arbitrarily and work with battery power [Juang et al. 2002; Madden et al. 2005]. They have to conserve their power for long use as replacement is costly or even impossible. Since radio use is the biggest source of battery drain, frequently sending all the data from sensors to the central site will be very energy-inefficient; but reducing the frequency will increase the response time of event detection. On the other hand, these sensors have some memory and computing power, so it is possible for the sensors to perform (cheaper) local computations and be more careful in the usage of radio for communication. As many events are captured by testing if a certain function exceeds a threshold, they can be exactly formulated as (k, f, τ, ϵ) functional monitoring problems.

In this context, various (k, f, τ, ϵ) functional monitoring problems have been studied under names like "reactive monitoring" (in networking [Dilman and Raz 2001]) and "distributed triggers" (in databases [Jain et al. 2004]). Prior works have considered many different functions f [Babcock and Olston 2003; Cormode and Garofalakis 2005; Cormode et al. 2006; 2007; Das et al. 2004; Dilman and Raz 2001; Huang et al. 2007; Jain et al. 2004; Sharfman et al. 2006], and typically each of these presents algorithms (often variants of GLOBAL or LOCAL described earlier) with correctness guarantees, but no nontrivial communication bounds. Some of the above works take a *distributed streaming* approach where in addition to optimizing the bits communicated, the algorithms also optimize the space and time requirements of each of the sensors.

Foundational motivations. Distributed functional monitoring is a natural combination of two well studied computation models: the k-party communication model and data streaming. In *communication complexity* [Yao 1979], the problem is to study the minimum number of bits needed to compute a given function f of distributed inputs over k parties. Framed in our setting, the goal is to compute $C(t)$ for a particular time t. We call it a *one-shot* problem. It is clear that the monitoring problem is at least as difficult as the corresponding one-shot problem (cf. Proposition 1).

The *streaming model* [Alon et al. 1999] has received much attention in recent years. Here the goal is to track $C(t)$ for all t but there is only one site ($k = 1$), and we are interested in the space complexity of the tracking algorithm, not communication. There are many functions f that can be computed with a relative ϵ error in the streaming model, using $\text{poly}(1/\epsilon, \log n)$ space: this includes problems such as the frequency moments, clustering, heavy hitters, quantiles, and so on [Muthukrishnan 2005].

It is well known that if a problem can be solved in the streaming model with small space, it can also be solved in the communication model with small communication, with only an extra $O(k)$ factor. This connection has been well exploited to derive space lower bounds on the former. But it is unclear whether a space-efficient streaming algorithm also implies a communication-efficient protocol for the corresponding distributed functional monitoring problem. In this paper, we provide some positive evidence on this question by showing that for some problems, the communication upper bound for a functional monitoring problem is only an $O(k)$

factor larger than the space bound of the corresponding streaming algorithm, ignoring polylogarithmic factors; but for some other problems, a more appreciable gap remains.

The model. Below we define our computation model more formally, under which we will analyze the algorithms and derive lower bounds. Let $A = (a_1, \ldots, a_m)$ be a sequence of items, where $a_i \in [n]$. The sequence A is observed in order by $k \geq 2$ remote *sites* $S_1, \ldots, S_k$ collectively, i.e., the item a_i is observed by exactly one of the sites at time t_i, where $t_1 < t_2 < \cdots < t_m$. Let $A(t)$ be the multiset of items received up until time t from all sites, and let $f : [n]^m \to \mathbb{R}$ be the function to be monitored. There is a designated *coordinator* C that is responsible for deciding if $f(A(t)) \geq \tau$ for some given threshold τ. More precisely, for some parameter $0 < \epsilon \leq 1/4$, the coordinator should output 1 if $f(A(t)) \geq \tau$; output 0 if $f(A(t)) \leq (1-\epsilon)\tau$; and is allowed either answer in-between. If $f(A(t))$ is non-decreasing, we can equivalently ask the coordinator to decide a time instance t, at which point an alarm is raised, such that $t_a \leq t \leq t_b$, where $t_a = \arg\min_t\{f(A(t)) > (1-\epsilon)\tau\}$ and $t_b = \arg\min_t\{f(A(t)) \geq \tau\}$. The algorithm terminates when we reach time t. We also consider probabilistic protocols that may err with some probability $\delta < 1/2$.

We define the manner of communication more precisely as follows. There is a two-way communication channel between the coordinator C and each of the k sites, but there is no direct communication between any two sites. Communication can only be initiated by a site upon the arrival of an element; the coordinator never initiates communication spontaneously, nor does a site when no element arrives. Specifically, suppose site S_j receives the item a_i at time t_i. Based on its local status, S_j may choose to send a message to C, which in turn may trigger iterative communication with other sites. We assume that communication is instantaneous. When all communication finishes, all the sites who have been involved may have new statuses, getting ready for the next item a_{i+1} to arrive.

We assume that all parties know the values of τ, ϵ, and n in advance, but not m. The cost of an algorithm is measured by the number of bits that are communicated in total. We assume that the threshold τ is sufficiently large to simplify analysis and the bounds. Dealing with small τ's is mainly technical: we just need to carefully choose when to use the naïve algorithm that simply sends every single item to the coordinator.

The following simple observation implies that the monitoring problem is almost always as hard as the corresponding one-shot problem.

PROPOSITION 1. *For any monotone function f, an algorithm for (k, f, τ, ϵ) functional monitoring that communicates $g(k, n, m, \tau, \epsilon)$ bits implies a one-shot algorithm that communicates $g(k, n, m, \tau, \epsilon) + O(k)$ bits.*

PROOF. The site S_1 first starts running the monitoring algorithm on its local stream, while the rest pretend that none of their elements have arrived. When S_1 finishes, it sends a special message to the coordinator, which then signals S_2 to start. We continue this process until all k sites have finished, or an alarm is raised (output changes to 1) in the middle of the process. □

Our results. Several works in the database community have considered functional monitoring problems essentially in the model we described above [Cormode and

Garofalakis 2005; Cormode et al. 2005; Cormode et al. 2007; Keralapura et al. 2006], but the devised solutions typically are heuristics-based [Babcock and Olston 2003; Olston et al. 2003; Sharfman et al. 2006], with no or very large worst-case bounds on the communication. No lower bounds are known. In this paper, we take a first step towards a formal study of functional monitoring. In particular, we focus on the *frequency moments*, i.e., $f = F_p = \sum_i m_i^p$ where m_i is the frequency of item i, and derive both upper and lower bounds for monitoring these functions. Estimating the frequency moments has become the keystone problem in streaming algorithms since the seminal paper of Alon et al. [1999]. In particular, the first three frequency moments ($p = 0, 1, 2$) have received the most attention. F_1 is the simple SUM problem above, F_0 corresponds to the number of distinct items, and F_2 has found many applications in statistics and databases.

—For the (k, F_1, τ, ϵ) problem, we show deterministic bounds of $O(k \log 1/\epsilon)$ and $\Omega(k \log \frac{1}{\epsilon k})$[1]; and randomized bounds of $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ and $\Omega(\min\{k, \frac{1}{\epsilon}\})$, where δ is the algorithm's probability of failure. Hence, randomization can give significant asymptotic improvement for large k, and curiously, k is not an inherent factor. These bounds improve the previous result of $O(k/\epsilon \cdot \log(\tau/k))$ in [Keralapura et al. 2006].

—For the (k, F_0, τ, ϵ) problem, we give a (randomized) upper bound of[2] $\tilde{O}(k/\epsilon^2)$, which improves upon the previous result of $O(k^2/\epsilon^3 \log n \log \frac{1}{\delta})$ in [Cormode et al. 2006]. We also give a lower bound of $\Omega(k)$. It is well known [Alon et al. 1999] that any deterministic algorithm that solves even the one-shot F_0 or F_2 problem has to use $\Omega(n)$ communication.

—Our most involved results are for the (k, F_2, τ, ϵ) problem: we present an upper bound of $\tilde{O}(k^2/\epsilon + (\sqrt{k}/\epsilon)^3)$ improving the previous result of $\tilde{O}(k^2/\epsilon^4)$ [Cormode and Garofalakis 2005]. We also give an $\Omega(k)$ lower bound. The algorithm is a sophisticated variation of GLOBAL above, with multiple rounds, using different "sketch summaries" at multiple levels of accuracy.

Table I summarizes our results. For comparison, we also include the one-shot costs: observe that for F_0 and F_1, the cost of monitoring is no higher than the one-shot computation and close to the lower bounds; but for F_2 is there a clear gap to address.

In this paper, we are mainly interested in the communication cost of the algorithms, and our lower bounds hold even assuming that the remote sites have infinite computing power. Nevertheless, all our algorithms can be implemented with low memory and computing costs at the remote sites and the coordinator.

Related results. Since this research was first published, [Cormode et al. 2008], some other functions have been studied in the distributed functional monitoring/tracking framework in the theory community. In [Arackaparambil et al. 2009], Arackaparambil et al. considered some functions that are not monotone, such as the *empirical entropy*. They demonstrate that monotonicity is not essential for function monitoring to work well: they show that as more arrivals occur, the entropy cannot change too fast, and so the total monitoring cost can be bounded in

[1]We use the notation $\log x = \max\{\log_2 x, 1\}$ throughout the paper.

[2]The $\tilde{O}$ notation suppresses logarithmic factors in $n, k, m, \tau, 1/\epsilon, 1/\delta$.

Moment	Lower bound	Upper bound
monitoring F_0, randomized	$\Omega(k)$	$\tilde{O}(\frac{k}{\epsilon^2})$
one-shot F_0, randomized	$\Omega(k)$	$\tilde{O}(\frac{k}{\epsilon^2})$
monitoring F_1, deterministic	$\Omega(k \log \frac{1}{\epsilon k})$	$O(k \log \frac{1}{\epsilon})$
one-shot F_1, deterministic	$\Omega(k \log \frac{1}{\epsilon k})$	$O(k \log \frac{1}{\epsilon})$
monitoring F_1, randomized	$\Omega(\min\{k, \frac{1}{\epsilon}\})$	$O(\min\{k \log \frac{1}{\epsilon}, \frac{1}{\epsilon^2} \log \frac{1}{\delta}\})$
one-shot F_1, randomized	$\Omega(k)$	$O(k \log \frac{1}{\epsilon\sqrt{k}})$
monitoring F_2, randomized	$\Omega(k)$	$\tilde{O}(k^2/\epsilon + (\sqrt{k}/\epsilon)^3)$
one-shot F_2, randomized	$\Omega(k)$	$\tilde{O}(\frac{k}{\epsilon^2})$

Table I. Summary of the communication complexity for one-shot computation and monitoring of different frequency moments. The "randomized" bounds are expected communication bounds for randomized algorithms with failure probability $\delta < 1/2$.

terms of the number of arrivals. They also gave either improved or incomparable lower bounds for monitoring the frequency moments[3]. The function f in general need not be a single-valued function: it can return a set of values, e.g., the set of *heavy hitters* and *quantiles*. Communication-optimal algorithms for tracking these statistics were given in [Yi and Zhang 2009b]. In the model defined above, items are only inserted but never deleted. If deletions are allowed, the worst-case bounds often become large and trivial. Yi and Zhang [2009a] propose to use competitive analysis in this case, and considered the special case where there is only one site, but it remains unclear how to properly define the competitive ratio when $k \geq 2$.

2. A GENERAL ALGORITHM FOR F_P, $P \geq 1$

We first present a general algorithm based on each site monitoring only local updates. This gives initial upper bounds, which we improve for specific cases in subsequent sections.

The algorithm proceeds in multiple rounds, based on a generalization of the GLOBAL idea outlined in the introduction. Let u_i be the frequency vector $(m_1, \ldots, m_n)$ at the beginning of round i. Note that the pth frequency moment, F_p, of u_i is $\|u_i\|_p^p$, where $\|u_i\|_p$ denotes the ℓ_p norm of u_i. In round i, every site keeps a copy of u_i and a threshold t_i. Let v_{ij} be the frequency vector of recent updates received at site j during round i. Whenever the impact of v_{ij} causes the F_p moment locally to increase by more than t_i (or multiples thereof), the site informs the coordinator. After the coordinator has received more than k such indications, it ends the round, collects information about all k vectors v_{ij} from sites, computes a new global state u_{i+1} and distributes it to all sites.

More precisely, we proceed as follows. Define the round threshold $t_i = \frac{1}{2}(\tau - \|u_i\|_p^p)k^{-p}$, chosen to divide the current "slack" uniformly between sites. Each site j receives a set of updates during round i, which we represent as a vector v_{ij}.

[3]In particular, for sufficiently large threshold τ, they show an $\Omega(k \log(1/\epsilon))$ lower bound for the (k, F_1, τ, ϵ) problem, which matches the upper bound we show here.

During round i, whenever $\lfloor(\|u_i + v_{ij}\|_p^p - \|u_i\|_p^p)/t_i\rfloor$ increases, site j sends a bit to indicate this (if this quantity increases by more than one, the site sends one bit for each increase). After the coordinator has received k bits in total, it ends round i and collects v_{ij} (or some compact summary of v_{ij}) from each site. It computes $u_{i+1} = u_i + \sum_{j=1}^k v_{ij}$, and hence t_{i+1}, and sends (a representation of) both u_{i+1} and t_{i+1} to all sites, beginning round $i+1$. The coordinator changes its output to 1 when $\|u_i\|_p^p \geq (1-\epsilon/2)\tau$, and the algorithm terminates.

THEOREM 1. *At the end of round i, we have $\|u_i\|_p^p + kt_i \leq \|u_{i+1}\|_p^p \leq 2k^p t_i + \|u_i\|_p^p$. There can be at most $O(k^{p-1}\log\frac{1}{\epsilon})$ rounds.*

PROOF. We first define the function $\psi(x,y) = \|x+y\|_p^p - \|x\|_p^p$. ψ is convex in both its arguments for all $p \geq 1$, in the range where x and y are non-negative (have no negative components). The first inequality is straightforward: each site sends a bit whenever its local F_p moment increases by t_i, i.e., we monitor $\psi(u_i, v_{ij})$. Observe that providing all vectors are non-negative, we have $\psi(u_i, \sum_{j=1}^k v_{ij}) \geq \sum_{j=1}^k \psi(u_i, v_{ij})$ (this can be seen by analyzing each dimension of each vector in turn). Thus, we have

$$\|u_{i+1}\|_p^p - \|u_i\|_p^p = \|u_i + \sum_{j=1}^k v_{ij}\|_p^p - \|u_i\|_p^p \geq kt_i.$$

For the second inequality, we have (by Jensen's inequality on the second argument of ψ, and monotonicity on the first argument):

$$\begin{aligned}
\|u_i + \sum_{j=1}^k v_{ij}\|_p^p - \|u_i\|_p^p &= \psi(u_i, \sum_{j=1}^k v_{ij}) \\
&\leq \frac{1}{k}\sum_{j=1}^k \psi(ku_i, kv_{ij}) = k^{p-1}\sum_{j=1}^k \psi(u_i, v_{ij}) \\
&= k^{p-1}\sum_{j=1}^k (\|u_i + v_{ij}\|_p^p - \|u_i\|_p^p) < 2k^p t_i.
\end{aligned}$$

The last bound follows by observing that we see k messages from the sites, one for each increase of $\|u_i + v_{ij}\|_p^p - \|u_i\|_p^p$ by t_i, so this cannot be larger than $2kt_i$ (kt_i from changes that have been notified, and up to t_i at each of $k-1$ sites apart from the one that triggers the end of the round).

By our choice of t_i, we ensure that this upper bound on the current global value of F_p never exceeds τ during a round, and we terminate the procedure as soon as it exceeds $(1-\epsilon/2)\tau$. Analyzing the number of rounds, from the lower bound above, we have

$$\begin{aligned}
t_{i+1} &= \frac{1}{2}(\tau - \|u_{i+1}\|_p^p)k^{-p} \leq \frac{1}{2}(\tau - \|u_i\|_p^p - kt_i)k^{-p} \\
&= \frac{1}{2}(2k^{p-1} - 1)t_i k^{1-p}.
\end{aligned}$$

So $t_{i+1} \leq (1 - k^{1-p}/2)t_i \leq (1-k^{1-p}/2)^i t_0$. Since $t_0 = \tau k^{-p}/2$, and we terminate

when $t_i < \epsilon\tau k^{-p}/4$, it is clear that there can be at most $O(k^{p-1}\log 1/\epsilon)$ rounds before this occurs. □

We now consider various special cases of (k, F_p, τ, ϵ) monitoring depending on the choice of p:

Case 1: $p = 1$**.** For the case $p = 1$, the above immediately implies a bound of $O(k \log 1/\epsilon)$ messages of counts being exchanged. In fact, we can give a tighter bound: the coordinator can omit the step of collecting the current v_{ij}'s from each site, and instead just sends a message to advance to the next stage. The value of t_i is computed simply as $2^{-1-i}\tau/k$, and the coordinator has to send only a constant number of bits to each site to signal the end of round i. Thus, we obtain a bound of $O(k \log 1/\epsilon)$ *bits*.

Case 2: $p = 2$**.** When $p = 2$, in order to concisely convey information about the vectors v_{ij} we make use of *sketch* summaries of vectors [Alon et al. 1999]. These sketches have the property that (with probability at least $1 - \delta$) they allow F_2 of the summarized vector to be estimated with relative error ϵ, in $O(\frac{1}{\epsilon^2}\log\tau \log\frac{1}{\delta})$ bits. We can apply these sketches in the above protocol for $p = 2$, by replacing each instance of u_i and v_{ij} with a sketch of the corresponding vector. Note that we can easily perform the necessary arithmetic to form a sketch of $u_i + v_{ij}$ and hence find (an estimate of) $\|u_i + v_{ij}\|_2^2$. In order to account for the inaccuracy introduced by the approximate sketches, we must carefully set the error parameter ϵ' of the sketches. Since we compare the change in $\|u_i + v_{ij}\|_2^2$ to t_i, we need the error given by the sketch—which is $\epsilon'\|u_i+v_{ij}\|_2^2$ —to be at most a constant fraction of t_i, which can be as small as $\frac{\epsilon\tau}{2}$. Thus we need to set $\epsilon' = O(\frac{\epsilon}{k^2})$. Putting this all together gives the total communication cost of $\tilde{O}(k^6/\epsilon^2)$.

Case 3: $p > 2$**.** For larger values of p, we can again use sketch-like summaries. This time, we can make use of the data summary structures of Bhuvanagiri et al. [2006], since these have the necessary summability properties. Each sketch occupies space $O(\frac{p}{\epsilon'^{1+2/p}}n^{1-\frac{2}{p}}(\log^2 n)(\log^2 F_1)\log\frac{1}{\delta})$. Since in each round we exchange $O(k)$ sketches with parameter $\epsilon' = \epsilon/k^p$, and conclude in at most $O(k^{p-1}\log 1/\epsilon)$ rounds, the total communication cost is bounded by $\tilde{O}(\frac{p}{\epsilon^{1+2/p}}k^{2p+1}n^{1-\frac{2}{p}})$.

3. BOUNDS FOR F_1

The general algorithm in the above section yields a deterministic algorithm for F_1 which communicates $O(k \log\frac{1}{\epsilon})$ bits. This is almost optimal for deterministic algorithms, as indicated by the following lower bound, which actually follows from a reduction from the one-shot case. The lower bound for the one-shot case seems to be folklore, but we include it here for completeness.

LEMMA 1. *Any deterministic algorithm that solves the F_1 one-shot problem has to communicate $\Omega(k \log\frac{1}{\epsilon k})$ bits. This lower bound also holds on the expected cost for Las Vegas algorithms.*

PROOF. Let $N = \{0, 1, \ldots, \tau\}$. Let f be the Boolean function on N^k that we wish to compute, i.e., $f(t_1, \ldots, t_k) = 0$ if $\sum_{j=1}^k t_j \leq (1-\epsilon)\tau$, 1 if $\sum_{j=1}^k t_j \geq \tau$, and a "$*$", meaning it could be either one or zero, for the rest of the entries. We call the

Cartesian product $N_1 \times N_2 \times \cdots \times N_k$, where $N_1, \ldots, N_k \subseteq N$, an *$f$-monochromatic rectangle* if f is constant on it. Let $d(f)$ be the size of the smallest partition of N^k into disjoint f-monochromatic rectangles. Then the communication cost of computing f is lower bounded by $\Omega(\log_2 d(f))$ [Yao 1979].

For simplicity we assume that $\epsilon\tau$ is an integer. Consider the entries $(i_1\epsilon\tau, \ldots, i_k\epsilon\tau)$ in N^k such that $\sum_{j=1}^{k} i_j = \frac{1}{\epsilon}$. The function f has value 1 on these positions, and the claim is that any two of them cannot be in any f-monochromatic rectangle. To see this, consider any two different such entries $(i_1\epsilon\tau, \ldots, i_k\epsilon\tau)$ and $(i'_1\epsilon\tau, \ldots, i'_k\epsilon\tau)$ and let $\hat{i}_j = \min\{i_j, i'_j\}$. If any Cartesian product contains both $(i_1\epsilon\tau, \ldots, i_k\epsilon\tau)$ and $(i'_1\epsilon\tau, \ldots, i'_k\epsilon\tau)$, then it must also contain $(\hat{i}_1\epsilon\tau, \ldots, \hat{i}_k\epsilon\tau)$. However, since $\sum_{j=1}^{k} \hat{i}_j \leq \frac{1}{\epsilon} - 1$, f must take value 0 at $(\hat{i}_1\epsilon\tau, \ldots, \hat{i}_k\epsilon\tau)$.

By simple combinatorics, there are $\binom{\frac{1}{\epsilon}+k}{k}$ such entries, and we have the lower bound

$$\Omega(\log_2 d(f)) = \Omega\left(\log_2 \binom{\frac{1}{\epsilon}+k}{k}\right) = \Omega\left(k \log_2 \frac{\frac{1}{\epsilon}+k}{k}\right).$$

It is not difficult to show that $\Omega(k)$ is also a lower bound, so we can write the overall lower bound as $\Omega(k \log \frac{1}{\epsilon k})$. □

THEOREM 2. *Any deterministic algorithm that solves (k, F_1, τ, ϵ) functional monitoring has to communicate $\Omega(k \log \frac{1}{\epsilon k})$ bits.*

PROOF. If $1/\epsilon = \omega(k)$, then the one-shot lower bound is at least $\omega(k)$. Thus, invoking Proposition 1, the one-shot lower bound also becomes a lower bound for the continuous problem.

If $1/\epsilon = O(k)$, an $\Omega(k \log \frac{1}{\epsilon k}) = \Omega(k)$ lower bound can be proved by the following argument. Consider the least difficult case $\epsilon = 1/2$. We construct the following inputs. Pick an arbitrary permutation $\pi = (i_1, \ldots, i_k)$ of $\{1, \ldots, k\}$. We first send τ/k elements to S_{i_1}, then τ/k elements to S_{i_2}, ..., τ/k elements to S_{i_k}. Note that the coordinator must output 1 at the end on every input. We argue that the coordinator has to communicate with at least $k/2$ sites on every input. Suppose for contradiction that on some permutation π only $s < k/2$ sites have communicated with the coordinator when it first outputs 1. Consider the permutation π' in which these s sites are the first to receive elements. The same communication would occur on π' by the time the first $s\tau/k$ elements are sent, and the coordinator would mistakenly output 1. □

If we allow randomized protocols that may err with a certain probability δ, we can design a sampling based algorithm whose complexity is independent of k. This is to be contrasted with the one-shot case, where there is an $\Omega(k)$ lower bound even for probabilistic algorithms.

THEOREM 3. *There is a randomized algorithm for (k, F_1, τ, ϵ) functional monitoring with error probability at most δ that communicates $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ bits.*

PROOF. We present a randomized algorithm derived from a careful implementation of the COIN TOSS idea from the introduction, with error probability 1/3. By running $O(\log \frac{1}{\delta})$ independent instances and raising an alarm when at least half

of the instances have raised alarms, we amplify to success probability $1-\delta$, as required. Every time a site has received $\epsilon^2\tau/(ck)$ elements, where c is some constant to be determined later, it sends a signal to the coordinator with probability $1/k$. The server raises an alarm as soon as it has received $c/\epsilon^2 - c/(2\epsilon)$ such signals, and terminates the algorithm. The communication bound is immediate. For correctness, it is sufficient to prove the following: On any sequence A', the algorithm fails to output 0 with probability at most 1/6 if $F_1(A') \leq (1-\epsilon)\tau$, and fails to output 1 with probability at most 1/6 if $F_1(A') \geq \tau$. Then for the given input sequence A, applying this statement on $A(t_a)$ and $A(t_b)$ proves the theorem (where t_a and t_b are as defined in Section 1).

Let X be the number of signals received by the coordinator. Its expectation is at most

$$\mathrm{E}[X] \leq 1/k \cdot F_1/(\epsilon^2\tau/(ck)) = cF_1/(\epsilon^2\tau),$$

and at least

$$\mathrm{E}[X] \geq 1/k \cdot (F_1 - \epsilon^2\tau)/(\epsilon^2\tau/(ck)) = cF_1/(\epsilon^2\tau) - c.$$

Its variance is

$$\mathrm{Var}[X] \leq (ckF_1)/(\epsilon^2\tau) \cdot (1/k - 1/k^2) \leq cF_1/(\epsilon^2\tau).$$

If $F_1 \leq (1-\epsilon)\tau$, then the probability that the coordinator outputs 1 is (by Chebyshev inequality)

$$\begin{aligned}\Pr[X \geq c/\epsilon^2 - c/(2\epsilon)] &\leq \Pr[X - \mathrm{E}[X] \geq c/(2\epsilon)] \\ &\leq \frac{c(1/\epsilon^2 - 1/\epsilon)}{(c/(2\epsilon))^2} \leq \frac{4}{c}.\end{aligned}$$

Similarly, if $F_1 \geq \tau$, then the probability that the coordinator does not output 1 is

$$\begin{aligned}\Pr[X \leq c/\epsilon^2 - c/(2\epsilon)] &\leq \Pr[X - \mathrm{E}[X] \leq -c/(2\epsilon) + c] \\ &\leq \frac{c/\epsilon^2}{(-c/(2\epsilon) + c)^2} \leq \frac{1}{c(1/2-\epsilon)^2} \leq \frac{16}{c}.\end{aligned}$$

Choosing $c = 96$ makes both probabilities at most 1/6, as desired. □

Therefore, the randomized algorithm is better than the deterministic algorithm for large k. Combined with the deterministic bound, we obtain the bound in Table I. In addition, we also have the following lower bound:

THEOREM 4. *For any $\epsilon < 1/4$, any probabilistic protocol for (k, F_1, τ, ϵ) functional monitoring that errs with probability smaller than 1/2 has to communicate $\Omega(\min\{k, 1/\epsilon\})$ bits in expectation.*

PROOF. Following the Minimax Principle [Yao 1977], it suffices to demonstrate a probability distribution on the inputs, and show that any deterministic algorithm that errs with probability at most 1/2 has to communicate expected $\Omega(\min\{k, 1/\epsilon\})$ bits.

Without loss of generality we assume that $1/\epsilon$ is an integer. Let $s = \min\{k, 1/\epsilon\}$. We also assume that $k/s = \max\{1, \epsilon k\}$ is an integer. Otherwise, we have $k > 1/\epsilon$. In this case we can reduce k to $\lfloor \epsilon k \rfloor \cdot 1/\epsilon$, affecting k by at most a factor of 2, and

leaving the asymptotic result unchanged. We divide the k sites into s groups, with each group having k/s sites. Our inputs are constructed as follows. We pick a permutation $\pi = (i_1, \ldots, i_s)$ of $\{1, \ldots, s\}$ uniformly at random. We first send τ/k elements to each site in the i_1-th group, then send τ/k elements to each site in the i_2-th group, ..., and finally send τ/k elements to each site in the i_s-th group. We claim that the coordinator in any deterministic algorithm has to communicate with $s/2$ groups in expectation in order to response correctly with probability at least $1/2$. Note that a correct response to any input is to raise an alarm by outputting 1 at the end, and to output 0 while the i_s-th group has not received elements.

Two inputs π_1 and π_2 are said to be *equivalent* if the following two conditions hold: (1) the deterministic algorithm has the same communication pattern, i.e., the coordinator communicates with the same set of groups with the same messages in the same order; and (2) the order of the groups that have ever communicated with the coordinator is the same in π_1 and π_2. Accordingly we partition all the $s!$ inputs into equivalence classes $P_1, \ldots, P_r$. Consider a particular class P_i. Suppose that on a $\pi \in P_i$, the coordinator communicates with s_i groups, for some $s_i \leq s$. Note that any other π' in which these s_i groups are in the same order as in π must also be P_i, regardless of the other $s - s_i$ groups. Thus there are $s!/s_i!$ inputs in P_i.

Suppose on all inputs in P_i, the last communication is triggered by group l_i. Considering a particular $\pi \in P_i$, for the algorithm to correctly report the alarm at the end, l_i must be the last group in π that we send elements to, because no further communication is triggered after l_i. Among all the inputs in P_i, $(s-1)!/(s_i-1)!$ of them end with l_i, namely a fraction of s_i/s. So the probability that the algorithm succeeds is at most $(\sum_{i=1}^{r} \frac{s_i}{s}|P_i|)/s!$, which is required to be at least $1/2$. On the other hand, the expected number of groups communicated is $\sum_{i=1}^{r} s_i \frac{|P_i|}{s!} = s(\sum_{i=1}^{r} \frac{s_i}{s}|P_i|)/s! \geq s/2$. □

4. BOUNDS FOR F_0

We know that the F_1 problem can be solved deterministically and exactly (by setting $\epsilon = 1/\tau$) by communicating $O(k \log \tau)$ bits. For any $p \neq 1$, the same arguments of Proposition 3.7 and 3.8 in [Alon et al. 1999] apply to show that both randomness (Monte Carlo) and approximation are necessary for the F_p problem in order to get solutions with communication cost better than $\Omega(n)$ for any $k \geq 2$. So for the rest of the paper we only consider probabilistic protocols that err with some probability δ.

Prior work solved the (k, F_0, τ, ϵ) monitoring problem using a sketch data structure to estimate the number of distinct items [Cormode et al. 2006]. The algorithm proceeded with each site holding a copy of a global sketch, and updating this sketch with their distinct items. When the estimated number of distinct items had increased by a $1 + \frac{\epsilon}{2k}$ factor, the new sketch would be shared with all sites, at a cost of $O(\frac{k}{\epsilon^2} \log \frac{1}{\delta})$ in communication. Therefore, the number of rounds is bounded by $O(\log_{1+\epsilon/2k}(1/\epsilon))$, which gives a total communication cost of $O(\frac{k^2}{\epsilon^3} \log \frac{1}{\epsilon} \log \frac{1}{\delta})$. We can improve significantly on this cost, partly by directly addressing the threshold monitoring base, but also by using a greater understanding of the sketch data structure. The algorithms in [Cormode et al. 2006] always sends entire sketches. By being more "sketch-aware", and opening up the structure of the sketch, we can

reduce the amount of information that is sent. In particular, we generalize the sketch of [Bar-Yossef et al. 2002] in a distributed fashion. The basic idea is that, since this F_0 sketch changes "monotonically", i.e., once an entry is added, it will never be removed, we can communicate to the coordinator every addition to all the sketches maintained by the individual sites, and bound the cost in terms of the size of a single sketch.

THEOREM 5. *There is a randomized algorithm for the (k, F_0, τ, ϵ) functional monitoring problem with error probability at most δ that communicates $O(k(\log n + \frac{1}{\epsilon^2}\log\frac{1}{\epsilon})\log\frac{1}{\delta})$ bits.*

PROOF. Below we present an algorithm with error probability 1/3. Again, this probability can be driven down to δ by running $O(\log\frac{1}{\delta})$ independent copies of the algorithm.

Define t as the integer such that $48/\epsilon^2 \leq \tau/2^t < 96/\epsilon^2$. The coordinator first picks two random pairwise independent hash functions $f : [n] \to [n]$ and $g : [n] \to [6 \cdot (96/\epsilon^2)^2]$, and send them to all the remote sites. This incurs a communication cost of $O(k(\log n + \log\frac{1}{\epsilon})) = O(k \log n)$ bits. Next, each of the remote sites evaluates $f(a_i)$ for every incoming element a_i, and tests if the last t bits of $f(a_i)$ are all zeros. If so it evaluates $g(a_i)$. There is a local buffer that contains all the $g()$ values for such elements. If $g(a_i)$ is not in the buffer, we add $g(a_i)$ into the buffer, and then send it to the coordinator. The coordinator also keeps a buffer of all the unique $g()$ values it has received, and outputs 1 whenever the number of elements in the buffer exceeds $(1 - \epsilon/2)\tau/2^t$. Since each $g()$ value takes $O(\log\frac{1}{\epsilon})$ bits, the coordinator receives at most $O(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon})$ bits from each site, over the k sites, giving the communication bound stated above.

We next prove the correctness of the algorithm. It is sufficient to prove the following: On any sequence A', the algorithm outputs 1 with probability at most 1/6 if $F_0(A') \leq (1-\epsilon)\tau$, and outputs 0 with probability at most 1/6 if $F_0(A') \geq \tau$.

One source of error is g having collisions between the elements it is evaluated on. Since g should be evaluated on at most $96/\epsilon^2$ elements, the probability that g has collisions is at most 1/12. From now on we assume that g has no collisions, and will add 1/12 to the final error probability.

Let X be the number of distinct elements in A' that have zeros in their last t bits of the $f()$ value. We know [Bar-Yossef et al. 2002] that $\mathrm{E}[X] = F_0/2^t$ and $\mathrm{Var}[X] \leq F_0/2^t$.

If $F_0 \leq (1-\epsilon)\tau$, then the algorithm outputs 1 with probability

$$\begin{aligned}\Pr[X > (1-\epsilon/2)\tau/2^t] &\leq \Pr[X - \mathrm{E}[X] > \epsilon\tau/2^{t+1}] \\ &\leq \frac{4 \cdot \mathrm{Var}[X]}{(\epsilon\tau/2^t)^2} \leq \frac{4F_0/2^t}{(\epsilon\tau/2^t)^2} \leq \frac{4F_0}{\epsilon^2\tau \cdot 48/\epsilon^2} \leq \frac{1}{12}.\end{aligned}$$

When F_0 reaches τ, the probability of outputting 0 is

$$\begin{aligned}\Pr[X \leq (1-\epsilon/2)\tau/2^t] &\leq \Pr[X - \mathrm{E}[X] \leq -\epsilon\tau/2^{t+1}] \\ &\leq \frac{4 \cdot \mathrm{Var}[X]}{(\epsilon\tau/2^t)^2} \leq \frac{1}{12}.\end{aligned}$$

Thus, the total error probability in either case is at most 1/6, as desired. □

Unlike the F_1 case where there is a randomized algorithm whose communication complexity is independent of k, we show below that this is not the case for F_0. To obtain a lower bound for randomized algorithms we invoke Yao's Minimax Principle [Yao 1977], which requires us to construct a probability distribution on the inputs, and show that any deterministic algorithm has to communicate a certain number of bits in expectation (w.r.t the distribution of the inputs). For this purpose we can model any deterministic monitoring algorithm as follows. Each remote site S_i maintains a set of an arbitrary number of *triggering conditions*. Each triggering condition is a frequency vector $(m_1, \ldots, m_n) \in [m]^n$. The site S_i will initiate communication when and only when the frequency vector of the elements it has received so far is one triggering condition. The communication may in turn lead to communication between the coordinator and other remote sites. After all the communication is completed, those sites that have communicated with the coordinator are allowed to change their sets of triggering conditions arbitrarily. We will show that the constructed inputs will trigger communication at least $\Omega(k)$ times. An implicit assumption in this model is that only the current state matters but not *how* the state is reached. For instance if $(0, 1, 1, 0, \ldots, 0)$ is a trigger condition, the site will trigger communication no matter if a "2" is observed before a "3" or the other way round. However, this assumption is not an issue in our proof, as in our construction of the inputs, there is at most one way to reach any state vector.

THEOREM 6. *For any $\epsilon \leq 1/4$, $n \geq k^2$, any probabilistic protocol for (k, F_0, τ, ϵ) functional monitoring that errs with probability smaller than $1/2$ has to communicate $\Omega(k)$ bits in expectation.*

PROOF. Following the Minimax Principle [Yao 1977], it suffices to demonstrate a probability distribution on the inputs, and show that any deterministic algorithm that errs with probability at most $1/8$ has to communicate $\Omega(k)$ bits in expectation.

For simplicity, we will use $\tau = k$ in the proof. Similar constructions work for larger τ's. The inputs are constructed as follows. We first pick an integer r between 1 and $k/2$ uniformly at random. We then proceed in r rounds. In the first round, we randomly pick an element from $\{1, \ldots, k\}$ and send it to all the sites; the order is irrelevant (for concreteness, say in the order $S_1, \ldots, S_k$). In the second round, we do the same thing except that the element is now chosen from $\{k+1, \ldots, 2k\}$. We continue this process until in the r-th round, we uniformly randomly send a different element from $\{(r-1)k+1, \ldots, rk\}$ to each of the k sites. We denote by I_r the set of inputs that end in r rounds. It can be easily verified that for any input in I_r, the algorithm can correctly terminate during and only during the r-th round. It is helpful to think of the input construction as follows. At first, with probability $p = \frac{1}{k/2}$, we (a) pick a different element randomly and send it to each of the k sites; otherwise, we (b) pick one random element and send it to all the sites. In case (a) we terminate the construction, and in case (b), we proceed to the next round. In the second round, we do the same except that the probability of choosing case (a) is $p = \frac{1}{k/2-1}$. We continue the process in this fashion for a maximum of $k/2$ rounds, using $p = \frac{1}{k/2-i+1}$ in the i-th round.

Since the algorithm is correct with probability at least $7/8$, there are $s \geq k/4$ values of r: $r_1 \leq r_2 \leq \cdots \leq r_s$, such that the algorithm is correct with probability

at least 3/4 within I_{r_j} for each of $j = 1, \ldots, s$. Note that for any deterministic algorithm, these r_j's are fixed. For any $1 \leq j \leq s-1$, consider the triggering conditions just before the r_j-th round. Note that these triggering conditions may depend on the elements received in the first $r_j - 1$ rounds. So let us consider a particular history H of the first $r_j - 1$ rounds in which case (b) is always chosen. There are k^{r_j-1} such histories, and each happens with equal probability. Let $z_{i,\ell} = 1$ if S_i will trigger communication when the next element it receives is ℓ, and $z_{i,\ell} = 0$ otherwise. We claim that for at least half of these histories, the following condition must hold.

$$\sum_{i=1}^{k} \sum_{\ell=(r_j-1)k+1}^{r_j k} z_{i,\ell} \geq \frac{k}{2}. \tag{1}$$

Indeed, we will show in the following that if (1) does not hold for a history H, then conditioned on the input being in I_{r_j} and having H as its history, the probability that the algorithm errs is at least 1/2. If this were the case for more than half of the histories, then the error probability would be more than 1/4 for I_{r_j}, contradicting the previous assumption.

To prove that if (1) does not hold for H, the algorithm is very likely to fail in the next round if $r = r_j$, consider a random input in I_{r_j} with history H. Recall that a randomly selected element from $\{(r_j - 1)k + 1, \ldots, r_j k\}$ is given to each of the k sites. The coordinator can output 1 only if some site triggers communication, whose probability is at most (by the union bound)

$$\sum_{i=1}^{k} \left(\frac{\sum_{\ell=(r_j-1)k+1}^{r_j k} z_{i,\ell}}{k} \right) \leq \frac{1}{2}.$$

Therefore we conclude that for any r_j, (1) must hold for at least half of its histories. Now consider the case that the input π belongs to some I_r such that $r > r_j$. This happens with probability $1 - r_j/(k/2)$. We next compute the expected number of messages that π triggers in the r_j-th round. Suppose that (1) holds and π sends ℓ to all the sites. Note that $\sum_{i=1}^{k} z_{i,\ell}$ sites will be triggered, unless they receive a message from the coordinator telling them to change their triggering conditions. So at least $\sum_{i=1}^{k} z_{i,\ell}$ messages need to be transmitted. Thus, the expected number of messages that π triggers in the r_j-th round is

$$\frac{1}{2} \cdot \sum_{\ell=(r_j-1)k+1}^{r_j k} \left(\frac{1}{k} \cdot \sum_{i=1}^{k} z_{i,\ell} \right) \geq \frac{1}{4}. \tag{2}$$

Summing up (2) over all r_j, the total expected number of messages is at least $\sum_{j=1}^{s} \left(1 - \frac{r_j}{k/2}\right) \cdot \frac{1}{4} = \Omega(k)$. □

5. BOUNDS FOR F_2

In the following, we present an F_2 monitoring algorithm that combines the multi-round framework of our general monitoring algorithm and the AMS sketch [Alon et al. 1999], giving a total communication cost of $\tilde{O}(k^2/\epsilon + k^{3/2}/\epsilon^3)$. Prior work for this problem monitored a local sketch of the value distribution, and sent a sketch of

size $\tilde{O}(\frac{1}{\epsilon^2})$ every time this grew by a $1+\frac{\epsilon^2}{4k}$ factor [Cormode and Garofalakis 2005]. The total number of communications per site is $\tilde{O}(\frac{k}{\epsilon^2})$, giving a total communication cost over all k sites that is bounded by $\tilde{O}(\frac{k^2}{\epsilon^4})$. Thus our results strictly improve on this bound. Our algorithm consists of two phases. At the end of the first phase, we make sure that the value of F_2 is between $\frac{3}{4}\tau$ and τ; while in the second phase, we more carefully monitor F_2 until it is in the range $((1-\epsilon)\tau, \tau)$. Each phase is divided into multiple rounds. In the second phase, each round is further divided into multiple sub-rounds to allow for more careful monitoring with minimal communication. We use sketches such that with probability at least $1-\delta$, they estimate F_2 of the sketched vector within $1\pm\epsilon$ using $O(\frac{1}{\epsilon^2}\log n \log\frac{1}{\delta})$ bits [Alon et al. 1999]. For now, we assume that all sketch estimates are within their approximation guarantees; later we discuss how to set δ to ensure small probability of failure over the entire computation.

Algorithm. We proceed in multiple rounds, which are in turn divided into sub-rounds. Let u_i be the frequency vector of the union of the streams at the beginning of the ith round. We use $\|u_i\|$ to denote the ℓ_2 norm of u_i, and the F_2 of u_i can be simplified as $\|u_i\|^2 = u_i^2$. Let $\hat{u}_i^2$ be an approximation of u_i^2. In round i, we use a local threshold $t_i = \frac{(\tau-\hat{u}_i^2)^2}{64k^2\tau}$. Let $v_{ij\ell}$ be the local frequency vector of updates received at site j during subround ℓ of round i, and let $w_{i\ell} = \sum_{j=1}^{k} v_{ij\ell}$ be the total increment of the frequency vectors in subround ℓ of round i. During each (sub)round, each site j continuously monitors its $v_{ij\ell}^2$, and sends a bit to the server whenever $\lfloor v_{ij\ell}^2/t_i \rfloor$ increases.

Phase one. In phase one, there is only one subround per round. At the beginning of round i, the server computes a $\frac{5}{4}$-overestimate $\hat{u}_i^2$ of the current u_i^2, i.e., $u_i^2 \le \hat{u}_i^2 \le \frac{5}{4}u_i^2$. This can be done by collecting sketches from all sites with a communication cost of $O(k\log n)$. Initially $\hat{u}_1^2 = u_1^2 = 0$. When the server has received k bits in total from sites, it ends the round by computing a new estimate $\hat{u}_{i+1}^2$ for u_{i+1}^2. If $\hat{u}_{i+1}^2 \ge \frac{15}{16}\tau$, then we must have $u_{i+1}^2 \ge \hat{u}_{i+1}^2/\frac{5}{4} \ge \frac{3}{4}\tau$, so we proceed to the second phase. Otherwise the server computes the new t_{i+1}, broadcasts it to all sites, and proceeds to the next round of phase one.

Analysis of phase one. The following lemma guarantees that the algorithm will never need to terminate during phase one.

LEMMA 2. *At the end of round i in phase one, $u_{i+1}^2 < \tau$.*

PROOF. Assuming pessimistically that all sites are just below the threshold of sending the next bit, once the server has received k bits, by the Cauchy-Schwartz

inequality, we have $w_{i\ell}^2 = (\sum_{j=1}^k v_{ij\ell})^2 \leq k \sum_{j=1}^k v_{ij\ell}^2 < 2k^2 t_i$. Therefore,

$$\begin{aligned} u_{i+1}^2 &= (u_i + w_{i\ell})^2 = u_i^2 + 2u_i w_{i\ell} + w_{i\ell}^2 \\ &\leq u_i^2 + 2\|u_i\| \cdot \|w_{i\ell}\| + w_{i\ell}^2 \\ &< u_i^2 + 2\|u_i\|\sqrt{2k^2 t_i} + 2k^2 t_i \\ &\leq u_i^2 + \frac{\sqrt{2}}{4}\|u_i\|\frac{\tau - \hat{u}_i^2}{\sqrt{\tau}} + \frac{(\tau - \hat{u}_i^2)^2}{32\tau} \\ &\leq u_i^2 + \frac{\sqrt{2}}{4}\|u_i\|\frac{\tau - u_i^2}{\sqrt{\tau}} + \frac{(\tau - u_i^2)^2}{32\tau} \\ &= u_i^2 + \left(\frac{\sqrt{2}\|u_i\|}{4\sqrt{\tau}} + \frac{1}{32} - \frac{u_i^2}{32\tau}\right)(\tau - u_i^2). \end{aligned}$$

Since $\frac{\|u_i\|}{\sqrt{\tau}} \leq 1$, $\left(-\frac{u_i^2}{32\tau} + \frac{\sqrt{2}\|u_i\|}{4\sqrt{\tau}} + \frac{1}{32}\right)$ is always less than 1, and we have $u_{i+1}^2 < \tau$. □

The communication cost in each round is $O(k \log n)$ bits, and we bound the number of rounds:

LEMMA 3. *There are $O(k)$ rounds in phase one.*

PROOF. We can bound the number of rounds by showing that sufficient progress can be made in each round. In each round, we know $w_{i\ell}^2 = (\sum_{j=1}^k v_{ij\ell})^2 \geq \sum_{j=1}^k v_{ij\ell}^2 \geq k t_i$, thus

$$\begin{aligned} u_{i+1}^2 = (u_i + w_{i\ell})^2 &\geq u_i^2 + w_{i\ell}^2 \geq u_i^2 + k t_i \\ &= u_i^2 + \frac{(\tau - \hat{u}_i)^2}{64k\tau} \geq u_i^2 + \frac{(\tau - \frac{15}{16}\tau)^2}{64k\tau} \\ &= u_i^2 + \Theta(\tau/k). \end{aligned}$$

So the total number of rounds in this phase is $O(k)$. □

The communication cost of phase one is thus bound by $O(k^2 \log n)$. It would be possible to continue the first phase by using more accurate estimates $\hat{u}_i^2$ until u_i^2 reaches $(1-\epsilon)\tau$, but this would result in a communication cost of $\tilde{O}(k^2/\epsilon^3)$. Instead, the use of subrounds in the second phase gives an improved bound.

Phase two. In the second phase, the server computes a $(1+\epsilon/3)$-overestimate $\hat{u}_i^2$ at the start of each round by collecting sketches from the sites with a communication cost of $O(k/\epsilon^2 \log n)$. The server keeps an upper bound $\hat{u}_{i,\ell}^2$ on $u_{i,\ell}^2$, the frequency vector at the beginning of the ℓ-th sub-round in round i.

As above, during each sub-round, each site j continuously monitors its $v_{ij\ell}^2$, and sends a bit to the server whenever $\lfloor v_{ij\ell}^2/t_i \rfloor$ increases. When the server has collected k bits in total, it ends the sub-round. Then, it asks each site j to send a $(1 \pm \frac{1}{2})$-approximate sketch for $v_{ij\ell}^2$. The server computes an estimate $\tilde{w}_{i\ell}^2$ for $w_{i\ell}^2$ by combining these sketches. Note that $\tilde{w}_{i\ell}^2 \in (1 \pm \frac{1}{2}) w_{i\ell}^2$. The server computes the new upper bound $\hat{u}_{i,\ell+1}^2$ for $u_{i,\ell+1}^2$ as

$$\hat{u}_{i,\ell+1}^2 = \hat{u}_{i,\ell}^2 + 2\sqrt{2}\|\hat{u}_{i,\ell}\| \cdot \|\tilde{w}_{i\ell}\| + 2\tilde{w}_{i\ell}^2. \tag{3}$$

Indeed, since

$$u_{i,\ell+1}^2 = (u_{i,\ell} + w_{i\ell})^2 \le u_{i,\ell}^2 + 2\|u_{i,\ell}\| \cdot \|w_{i\ell}\| + w_{i\ell}^2,$$

and $u_{i,\ell}^2 \le \hat{u}_{i,\ell}^2$, $w_{i\ell}^2 \le 2\tilde{w}_{i\ell}^2$, we have $u_{i,\ell+1}^2 \le \hat{u}_{i,\ell+1}^2$. Then the server checks if

$$\hat{u}_{i,\ell+1}^2 + 3k\|\hat{u}_{i,\ell+1}\|\sqrt{t_i} < \tau. \tag{4}$$

If (4) holds, the server starts sub-round $\ell + 1$. The local threshold t_i remains the same. If (4) does not hold, the whole round ends, and the server computes a new $\hat{u}_{i+1}^2$ for u_{i+1}^2. If $\hat{u}_{i+1}^2 \ge (1 - \frac{2}{3}\epsilon)\tau$, the server changes its output to 1 and terminates the algorithm. Otherwise, it computes the new t_{i+1}, sends it to all sites, and starts the next round.

Analysis of phase two. Below we assume $\epsilon < \frac{1}{4}$. We first prove correctness. The second phase of the algorithm never raises a false alarm, since if $\hat{u}_{i+1}^2 \ge (1 - \frac{2}{3}\epsilon)\tau$, then $u_{i+1}^2 \ge \hat{u}_{i+1}^2/(1 + \epsilon/3) > (1 - \epsilon)\tau$. The following lemma implies that the algorithm will never miss an alarm either.

LEMMA 4. *For any round i, at the end of the ℓ-th sub-round, $u_{i,\ell+1}^2 < \tau$.*

PROOF. Since the algorithm did not terminate at the end of the $(\ell - 1)$-th sub-round, by the condition of (4) we have $\hat{u}_{i,\ell}^2 + 3k\|\hat{u}_{i,\ell}\|\sqrt{t_i} < \tau$. At the end of the ℓ-th sub-round when the server has collected k bits, assuming pessimistically that all sites are just below the threshold of sending the next bit, by the Cauchy-Schwartz inequality, we have $w_{i\ell}^2 = (\sum_{j=1}^k v_{ij\ell})^2 \le k \sum_{j=1}^k v_{ij\ell}^2 \le 2k^2 t_i$. Since

$$2k^2 t_i = \frac{2(\tau - \hat{u}_i^2)^2}{64\tau} \le \frac{1}{128}(\tau - \hat{u}_i^2),$$

$$\text{and} \quad k\|u_{i,\ell}\|\sqrt{t_i} = \|u_{i,\ell}\|\frac{\tau - \hat{u}_i^2}{8\sqrt{\tau}} \ge \frac{\sqrt{3}}{16}(\tau - \hat{u}_i^2),$$

we have $2k^2 t_i \le \frac{1}{8\sqrt{3}} k\|u_{i,\ell}\|\sqrt{t_i}$. Thus,

$$\begin{aligned} u_{i,\ell+1}^2 = (u_{i,\ell} + w_{i\ell})^2 &\le u_{i,\ell}^2 + 2\|u_{i,\ell}\| \cdot \|w_{i\ell}\| + w_{i\ell}^2 \\ &\le u_{i,\ell}^2 + 2\|u_{i,\ell}\|\sqrt{2k^2 t_i} + 2k^2 t_i \\ &\le u_{i,\ell}^2 + (2\sqrt{2} + \frac{1}{8\sqrt{3}})k\|u_{i,\ell}\|\sqrt{t_i} \\ &< \hat{u}_{i,\ell}^2 + 3k\|\hat{u}_{i,\ell}\|\sqrt{t_i} < \tau. \end{aligned}$$

□

Now we proceed to the analysis of the algorithm's communication complexity. It is clear that the cost of a sub-round is $O(k \log n)$ bits, since each $(1 \pm \frac{1}{2})$-approximate sketch for $v_{ij\ell}$ has $O(\log n)$ bits. Apart from the sub-round communication cost, each round has an additional $O(\frac{k}{\epsilon^2} \log n)$ cost to compute $\hat{u}_i$. All the other costs, e.g., the bits signaling the start and end of a sub-round, broadcasting t_i, etc., are asymptotically dominated by these costs. Therefore, the problem reduces to bounding the number of rounds and sub-rounds.

LEMMA 5. *In any round, the number of sub-rounds is $O(\sqrt{k})$.*

PROOF. At the end of the ℓ-th sub-round, the server has received k bits, so $w_{i\ell}^2 \geq \sum_{j=1}^{k} v_{ij\ell}^2 \geq kt_i$. Since $\tilde{w}_{i\ell}^2$ is a $(1 \pm \frac{1}{2})$-estimate of $w_{i\ell}^2$, we have $\tilde{w}_{i\ell}^2 \geq \frac{1}{2} w_{i\ell}^2 \geq kt_i/2$. According to (3),

$$\begin{aligned}
\hat{u}_{i,\ell+1}^2 &\geq \hat{u}_{i,\ell}^2 + 2\sqrt{2}\|\hat{u}_{i,\ell}\| \cdot \|\tilde{w}\| \\
&\geq \hat{u}_{i,\ell}^2 + 2\sqrt{2}\|\hat{u}_{i,\ell}\| \cdot \sqrt{\frac{kt_i}{2}} = \hat{u}_{i,\ell}^2 + \frac{1}{4}\|\hat{u}_{i,\ell}\| \cdot \frac{\tau - \hat{u}_i^2}{\sqrt{k\tau}} \\
&\geq \hat{u}_{i,\ell}^2 + \frac{1}{4} \cdot \frac{\sqrt{3}}{2} \cdot \frac{1}{\sqrt{k}}(\tau - \hat{u}_i^2) = \hat{u}_{i,\ell}^2 + \frac{\sqrt{3}}{8\sqrt{k}}(\tau - \hat{u}_i^2).
\end{aligned}$$

For any ℓ, if the ℓ-th sub-round starts, by (4) we have $\hat{u}_{i,\ell}^2 + 3k\|\hat{u}_{i,\ell}\|\sqrt{t_i} < \tau$, or

$$\tau > \hat{u}_{i,\ell}^2 + \frac{3}{8} \cdot \|\hat{u}_{i,\ell}\| \frac{\tau - \hat{u}_i^2}{\sqrt{\tau}} > \hat{u}_{i,\ell}^2 + \frac{3}{8} \cdot \frac{\sqrt{3}}{2}(\tau - \hat{u}_i^2).$$

$$\text{Rearranging,} \quad \hat{u}_{i,\ell}^2 < \tau - \frac{3\sqrt{3}}{16}(\tau - \hat{u}_i^2).$$

As $\hat{u}_{i,1}^2 = \hat{u}_i^2$, there are at most $\left(\tau - \frac{3\sqrt{3}}{16}(\tau - \hat{u}_i^2) - \hat{u}_i^2\right) / \left(\frac{\sqrt{3}}{8\sqrt{k}} \cdot (\tau - \hat{u}_i^2)\right) < 4\sqrt{k}$ sub-rounds in phase two. □

LEMMA 6. *The total number of rounds is* $O(\sqrt{k}/\epsilon)$.

PROOF. Focus on one round, say round i. Suppose there are $s < 4\sqrt{k}$ sub-rounds in this round. For any ℓ, we have $w_{i\ell}^2 < \tau/4$; else the subround would have ended earlier. So $\tilde{w}_{i\ell}^2 < 3\tau/8$. We first show how the upper bound $\hat{u}_{i,\ell}$ increases in each sub-round. From (3), $\hat{u}_{i,\ell+1}^2$ is at most

$$\hat{u}_{i,\ell}^2 + 2\sqrt{2}\sqrt{\tau} \cdot \|\tilde{w}_{i\ell}\| + 2\sqrt{3\tau/8} \cdot \|\tilde{w}_{i\ell}\| < \hat{u}_{i,\ell}^2 + 5\sqrt{\tau} \cdot \|\tilde{w}_{i\ell}\|,$$

$$\text{so} \quad \hat{u}_{i,s+1}^2 \leq \hat{u}_{i,1}^2 + 5\sqrt{\tau} \sum_{\ell=1}^{s} \|\tilde{w}_{i\ell}\|. \tag{5}$$

We know that $\hat{u}_{i,s+1}$ violates (4), so

$$\begin{aligned}
\tau &\leq \hat{u}_{i,s+1}^2 + 3k\|\hat{u}_{i,s+1}\|\sqrt{t_i} \leq \hat{u}_{i,s+1}^2 + \frac{3}{8}\|\hat{u}_{i,s+1}\| \frac{\tau - \hat{u}_i^2}{\sqrt{\tau}} \\
&< \hat{u}_{i,s+1}^2 + \frac{3}{8}(\tau - \hat{u}_i^2).
\end{aligned}$$

Substituting into (5), together with $\hat{u}_{i,1} \leq (1 + \epsilon/3)u_i^2$, we have

$$\sum_{\ell=1}^{s} \|\tilde{w}_{i\ell}\| > \frac{\tau - \frac{3}{8}(\tau - u_i^2) - (1 + \frac{1}{3}\epsilon)u_i^2}{5\sqrt{\tau}} = \frac{1}{8} \cdot \frac{\tau - (1 + \frac{8}{15}\epsilon)u_i^2}{\sqrt{\tau}}. \tag{6}$$

Next, we lower bound $u_{i+1}^2 = u_{i,s+1}^2$, to show that we must have made progress

by the end of this round. Since $u_{i,\ell+1}^2 = (u_{i,\ell} + w_{i\ell})^2 \geq u_{i,\ell}^2 + w_{i\ell}^2$, we have

$$\begin{aligned}
u_{i+1}^2 &\geq u_i^2 + \sum_{\ell=1}^{s} w_{i\ell}^2 \\
&\geq u_i^2 + \frac{1}{2}\sum_{\ell=1}^{s} \tilde{w}_{i\ell}^2 \geq u_i^2 + \frac{1}{2s}(\sum_{\ell=1}^{s} \|\tilde{w}_{i\ell}\|)^2 \qquad \text{(Cauchy-Schwarz inequality)} \\
&> u_i^2 + \frac{1}{64s\tau}(\tau - (1+\frac{8}{15}\epsilon)u_i^2)^2 \qquad \text{(by (6))} \\
&> u_i^2 + \frac{1}{256\sqrt{k}\cdot\tau}(\tau - (1+\frac{8}{15}\epsilon)u_i^2)^2 \qquad \text{(Lemma 5)}
\end{aligned}$$

Initially we have $u_1^2 \geq \frac{3}{4}\tau$, and the algorithm terminates as soon as u_i^2 exceeds $(1-\frac{2}{3}\epsilon)\tau$. For $a = 3, 4, \ldots, \log\frac{3}{2\epsilon}$ (assuming w.l.o.g. that $\frac{3}{2\epsilon}$ is a power of 2), we bound the number of rounds for u_i^2 to increase from $(1-2^{-a+1})\tau$ to $(1-2^{-a})\tau$, as:

$$\begin{aligned}
&\frac{\tau(1-2^{-a}-(1-2^{-a+1}))}{\frac{1}{256\sqrt{k}\cdot\tau}(\tau-(1+\frac{8}{15}\epsilon)(1-2^{-a})\tau)^2} + 1 \\
< &\frac{2^{-a}\tau}{\frac{1}{256\sqrt{k}\cdot\tau}(\frac{2^{-a}}{5}\tau)^2} + 1 = 2^a 25^2 \cdot 256\sqrt{k} + 1.
\end{aligned}$$

Summing over all a, we obtain that the total number of rounds is $O(\sqrt{k}/\epsilon)$. □

Combining Lemma 5 and 6, we know that there are a total of $O(k/\epsilon)$ sub-rounds and $O(\sqrt{k}/\epsilon)$ rounds. Thus phase two incurs a communication cost of $O((k^2/\epsilon + k^{3/2}/\epsilon^3)\log n)$. Recall that the cost of phase one is $O(k^2 \log n)$. So far we have assumed that all the estimates are always within the claimed approximation ranges. Since we have in total computed $O(\text{poly}(k/\epsilon))$ estimates, by running $O(\log\frac{k}{\epsilon\delta})$ independent repetitions and taking the median for each estimate, we can guarantee an overall error probability of no more than δ by the union bound. Thus, we conclude with the following.

THEOREM 7. *The (k, F_2, τ, ϵ) functional monitoring problem can be solved by an algorithm with a communication cost of $O((k^2/\epsilon + k^{3/2}/\epsilon^3)\log n \log\frac{k}{\epsilon\delta})$ bits and succeeds with probability at least $1-\delta$.*

Lastly, we comment that as a byproduct of the previous algorithm, the coordinator receives a sketch which summarizes the distribution. This sketch can then be used to approximate inner-product queries, wavelet and histogram decompositions etc., with bounds which follow from prior work on sketches. See [Cormode and Garofalakis 2005] for more details of such applications.

F_2 lower bound. Similar to the F_0 case, we prove an $\Omega(k)$ lower bound for monitoring F_2.

THEOREM 8. *For any $\epsilon \leq 1/4$, $n \geq k^2$, any probabilistic protocol for (k, F_2, τ, ϵ) functional monitoring that errs with probability smaller than $1/2$ has to communicate $\Omega(k)$ bits in expectation.*

PROOF. For simplicity, we will use $\tau = k^2$ in the proof. Similar constructions work for larger τ's.

We follow the same framework as in the proof of Theorem 6, with the following differences.

The inputs are constructed as follows. As before, we proceed in r rounds, where r is between 1 and $k/2$. At first, with probability $p = \frac{1}{k/2}$, we (a) pick one random element and send it to all the sites; otherwise, we (b) pick a different element randomly and send it to each of the k sites. In case (a) we terminate the construction, and in case (b), we proceed to the next round. In the second round, we do the same except that the probability of choosing case (a) is $p = \frac{1}{k/2-1}$. We continue this process in this fashion for a maximum of $k/2$ rounds, using $p = \frac{1}{k/2-i+1}$ in the i-th round. We denote by I_r the set of inputs that end in r rounds. It can be easily verified that for any input in I_r, the algorithm can correctly terminate during and only during the r-th round.

Arguing as before, we conclude that for any r_j, (1) holds for at least half of its histories. Now consider the case that the input π belongs to some I_r such that $r > r_j$. This happens with probability $1 - r_j/(k/2)$. When (1) holds and π sends ℓ_1 to S_1, ℓ_2 to S_2, etc., the number of messages that will be triggered is at least $z_{1,\ell_1} + \cdots + z_{k,\ell_k}$. Thus the expected number of messages that π triggers in the r_j-th round is at least (where summation is over all permutations $\ell_1, \ldots, \ell_k$)

$$\begin{aligned} \frac{1}{2} \cdot \sum \frac{1}{k!}(z_{1,\ell_1} + \cdots + z_{k,\ell_k}) &= \frac{1}{2} \cdot \frac{1}{k!}((k-1)!z_{1,1} + (k-1)!z_{1,2} + \cdots + (k-1)!z_{k,k}) \\ &\geq \frac{1}{4}. \end{aligned} \tag{7}$$

Summing (7) over all r_j, the total expected number of messages is at least $\sum_{j=1}^{s} \left(1 - \frac{r_j}{k/2}\right) \cdot \frac{1}{4} = \Omega(k)$. □

6. CONCLUSION AND OPEN PROBLEMS

For functional monitoring problems (k, f, τ, ϵ), we observe that for some functions, the communication cost is close to or the same as the cost for one-time computation of f, and that in some cases the cost can be less than the number of participants, k. Our results for F_2 make careful use of compact sketch summaries, switching between different levels of approximation quality to minimize the overall cost. These algorithms are more generally useful, since they immediately apply to monitoring L_2 and L_2^2 of arbitrary non-negative vectors, which is at the heart of many practical computations such as join size, wavelet and histogram representations, geometric problems, etc. [Cormode and Garofalakis 2005; Indyk 2004].

The immediate open question is to close the gap in the F_2 case: can a better lower bound than $\Omega(k)$ be shown, or do there exist $\tilde{O}(k \cdot \text{poly}(1/\epsilon))$ solutions? More broadly, there are many interesting directions to pursue. It remains open fully understanding the cost for monitoring other functions (or classes of functions), for example, statistics such as rolling average, information gain, variance; geometric descriptors such as coreset, ϵ-net and ϵ-approximation; various clusterings. It would be of interest to demonstrate a functional monitoring problem that is strictly harder than its one-shot version (ignoring polylogarithmic factors).

In some situations, it may be appropriate to use other measures of the cost, for example, the maximum cost per site rather than the total cost; or giving a competitive analysis relative to the best appropriately defined "off-line" adversary. There are also several variants of the basic model to consider, for example, the difference between one-way and two-way communication between the sites and the coordinator, the power of having a broadcast channel from the coordinator to all the sites, and the difference between having sliding windows and unbounded windows at the sites. As mentioned earlier, since the first published version [Cormode et al. 2008] of this paper there has already been subsequent research [Arackaparambil et al. 2009; Yi and Zhang 2009a; 2009b] which extends the scope of this work in various ways. We hope that ultimately, this line of study will lead to a new theory of *continuous* communication complexity.

Acknowledgments. We thank the anonymous reviewers for many helpful comments that have improved the presentation, and for identifying a hole in the proof of Theorem 4 in an earlier draft.

REFERENCES

Alon, N., Matias, Y., and Szegedy, M. 1999. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences 58*, 137–147.

Arackaparambil, C., Brody, J., and Chakrabarti, A. 2009. Functional monitoring without monotonicity. In *Proc. International Colloquium on Automata, Languages, and Programming*.

Babcock, B. and Olston, C. 2003. Distributed top-k monitoring. In *Proc. ACM SIGMOD International Conference on Management of Data*.

Bar-Yossef, Z., Jayram, T. S., Kumar, R., Sivakumar, D., and Trevisan, L. 2002. Counting distinct elements in a data stream. In *RANDOM*.

Bhuvanagiri, L., Ganguly, S., Kesh, D., and Saha, C. 2006. Simpler algorithm for estimating frequency moments of data streams. In *Proc. ACM-SIAM Symposium on Discrete Algorithms*.

Cormode, G. and Garofalakis, M. 2005. Sketching streams through the net: Distributed approximate query tracking. In *Proc. International Conference on Very Large Databases*.

Cormode, G., Garofalakis, M., Muthukrishnan, S., and Rastogi, R. 2005. Holistic aggregates in a networked world: Distributed tracking of approximate quantiles. In *Proc. ACM SIGMOD International Conference on Management of Data*.

Cormode, G., Muthukrishnan, S., and Yi, K. 2008. Algorithms for distributed functional monitoring. In *Proc. ACM-SIAM Symposium on Discrete Algorithms*.

Cormode, G., Muthukrishnan, S., and Zhuang, W. 2006. What's different: Distributed, continuous monitoring of duplicate resilient aggregates on data streams. In *Proc. IEEE International Conference on Data Engineering*.

Cormode, G., Muthukrishnan, S., and Zhuang, W. 2007. Conquering the divide: Continuous clustering of distributed data streams. In *Proc. IEEE International Conference on Data Engineering*.

Das, A., Ganguly, S., Garofalakis, M., and Rastogi, R. 2004. Distributed set-expression cardinality estimation. In *Proc. International Conference on Very Large Databases*.

Dilman, M. and Raz, D. 2001. Efficient reactive monitoring. In *IEEE INFOCOM*.

Huang, L., Nguyen, X., Garofalakis, M., Hellerstein, J., Joseph, A. D., Jordan, M., and Taft, N. 2007. Communication-efficient online detection of network-wide anomalies. In *IEEE INFOCOM*.

Indyk, P. 2004. Algorithms for dynamic geometric problems over data streams. In *Proc. ACM Symposium on Theory of Computation*.

Jain, A., Hellerstein, J., Ratnasamy, S., and Wetherall, D. 2004. A wakeup call for internet monitoring systems: The case for distributed triggers. In *Proceedings of the 3rd Workshop on Hot Topics in Networks (Hotnets)*.

Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L., and Rubenstein, D. 2002. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiments with zebranet. In *ASPLOS-X*.

Keralapura, R., Cormode, G., and Ramamirtham, J. 2006. Communication-efficient distributed monitoring of thresholded counts. In *Proc. ACM SIGMOD International Conference on Management of Data*.

Madden, S., Franklin, M., Hellerstein, J., and Hong, W. 2005. TinyDB: an acquisitional query processing system for sensor networks. *ACM Transactions on Database Systems 30,* 1, 122–173.

Muthukrishnan, S. 2005. *Data Streams: Algorithms and Applications*. Now Publishers.

Olston, C., Jiang, J., and Widom, J. 2003. Adaptive filters for continuous queries over distributed data streams. In *Proc. ACM SIGMOD International Conference on Management of Data*.

Sharfman, I., Schuster, A., and Keren, D. 2006. A geometric approach to monitoring threshold functions over distribtuted data streams. In *Proc. ACM SIGMOD International Conference on Management of Data*.

Yao, A. C. 1977. Probabilistic computations: Towards a unified measure of complexity. In *Proc. IEEE Symposium on Foundations of Computer Science*.

Yao, A. C. 1979. Some complexity questions related to distributive computing. In *Proc. ACM Symposium on Theory of Computation*.

Yi, K. and Zhang, Q. 2009a. Multi-dimensional online tracking. In *Proc. ACM-SIAM Symposium on Discrete Algorithms*.

Yi, K. and Zhang, Q. 2009b. Optimal tracking of distributed heavy hitters and quantiles. In *Proc. ACM Symposium on Principles of Database Systems*.