Brief Announcement: Tracking Distributed Aggregates over Time-based Sliding Windows

Graham Cormode
AT&T Labs–Research
Florham Park, NJ, USA
graham@research.att.com

Ke Yi
Hong Kong University of Science and Technology
Hong Kong, PRC
yike@cse.ust.hk

ABSTRACT

The area of distributed monitoring requires tracking the value of a function of distributed data as new observations are made. An important case is when attention is restricted to only a recent time period, such as the last hour of readings—the sliding window case. In this announcement, we outline a novel paradigm for handling such monitoring problems, which we dub the "forward/backward" approach. This provides clean solutions for several fundamental problems, such as counting, tracking frequent items, and maintaining order statistics. We obtain efficient protocols for these problems that improve on previous work, and are easy to implement. Specifically, we obtain optimal $O(\frac{k}{\varepsilon}\log(\varepsilon n/k))$ communication per window of n updates for tracking counts and heavy hitters with accuracy ε across k sites; and near-optimal communication of $O(\frac{k}{\varepsilon}\log^2(1/\varepsilon)\log(n/k))$ for quantiles.

Categories and Subject Descriptors

F.2.2 [**Analysis of algorithms and problem complexity**]: Nonnumerical algorithms and problems

General Terms

Algorithms

Keywords

Data streams, sliding windows

1. INTRODUCTION

Problems of distributed tracking involve trying to compute various aggregates over data that is distributed across multiple observing sites. Each site observes a stream of information, and aims to collaborate with the other sites to continuously track a function over the union of the streams. For example, a number of routers in a network might try to collaborate to track the most popular destinations. The goal is to allow a single distinguished entity, known as the "coordinator", to track the desired function. Within such settings, it is natural to only want to capture the recent behavior—say, the most popular destinations within the last 24 hours. Thus, attention is limited to a "time-based sliding window".

For these problems, the primary goal is to analyze the (total) communication required to achieve accurate tracking. This should be much smaller than the trivial solution of simply centralizing all the observations at the coordinator site. In this area, prior work has tended to be network topology agnostic, and measures just the total number of bytes transmitted by the protocols. Secondary goals include minimizing the space required at each site to run the protocol, and the time to process each new observation. These quantities are functions of k, the number of distributed sites, n, the total size of the input data, and ε, an approximation parameter to tolerate some imprecision in the computed answer.

Within this context, there has been much work on the "infinite window" case, where all historic data is included. Results have been shown for monitoring functions such as counts, distinct counts, order statistics, join sizes, entropy, and others [1, 3, 5, 6, 8, 10, 13]. Lately, there has been interest in only tracking a window of recent observations, defined by all those elements which arrived within the most recent w time units. Results in this model have been shown for tracking counts and frequent items [3], and for sampling [7].

Our results are most directly related to the recent work of Chan *et al.* [3]. The approach taken in [3] is somewhat complicated: the analysis of the proposed protocols is quite lengthy, and requires detailed analysis of multiple cases. They consider three problems: basic counting, which is to maintain the count of items observed within the window; heavy hitters, which is to maintain all items whose frequency (within the window) is more than a given fraction; and to maintain the quantiles of the distribution. Each problem tolerates an error of ε, and is parameterized by k, the number of sites participating in the computation, and n, the number of items arriving in a window. [3] shows (per window) communication costs of $O(\frac{k}{\varepsilon}\log\frac{\varepsilon n}{k})$ bits for basic counting, $O(\frac{k}{\varepsilon}\log\frac{n}{k})$ words for frequent items and $O(\frac{k}{\varepsilon^2}\log\frac{n}{k})$ words for quantiles. Our main contributions are simple algorithms with straightforward analysis which meet and in some cases improve on these bounds. To do this, we outline a conceptually simple approach for decomposing sliding windows, which also extends naturally to other problems in this setting. We call this the "forward/backward" framework. This can be applied to tracking counts, heavy hitters and quantiles to obtain optimal or near optimal communication bounds. It also extends to other functions, such as distinct counts and geometric properties.

Problem definitions and our results. Figure 1 shows the model: k sites each observe a stream S_i of item arrivals, and communicate with a single distinguished coordinator node to continuously compute some function of the union of the update streams.

The *basic counting* problem is to track (approximately) the number of items which have arrived across all sites within the last w time units. More precisely, let the stream of items observed at site i be S_i, a set of $(x, t(x))$ pairs, which indicates that an item x arrives at time $t(x)$. Then the exact basic count at time t is given by

$$C(t) = \sum_{1\le i\le k} |\{(x,t(x)) \in S_i \mid t - t(x) \le w\}|.$$

PODC'11, June 6–8, 2011, San Jose, California, USA.
ACM 978-1-4503-0719-2/11/06.

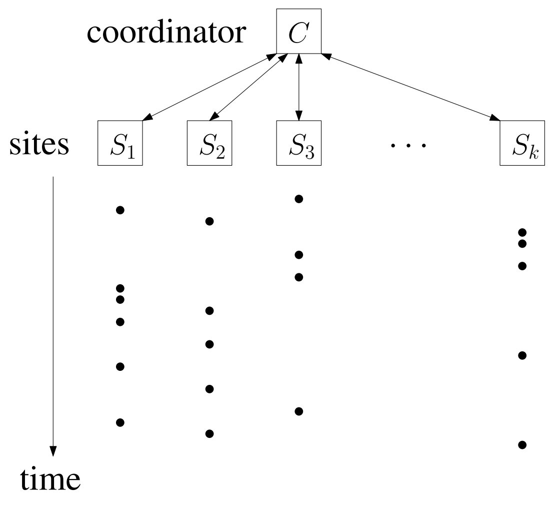

Figure 1: Schematic of the distribute streaming model

Tracking $C(t)$ exactly requires alerting the coordinator every time an item arrives or expires, so the goal is to track $C(t)$ approximately within an ε-error, i.e., the coordinator should maintain a $\tilde{C}(t)$ such that $(1-\varepsilon)C(t) \leq \tilde{C}(t) \leq (1+\varepsilon)C(t)$ at all times t.

The *heavy hitters* problem extends the basic counting problem, and generalizes the concept of finding the mode [11]. In the basic counting problem we count the total number of all items, while here we count the frequency of every distinct item x, i.e., the coordinator tracks the approximate value of

$$n_x(t) = \sum_{1 \leq i \leq k} |(x, t(x)) \in S_i \mid t - t(x) \leq w\}|.$$

Since it is possible that many $n_x(t)$ are small, say 0 or 1 for all x, requiring a multiplicative approximation for all x would require reporting all items to the coordinator. Consequently, the commonly adopted approximation guarantee for heavy hitters is to maintain a $\tilde{n_x}(t)$ that has an additive error of at most $\varepsilon C(t)$, where $C(t)$ is the total count of all items. This essentially makes sure that the "heavy" items are counted accurately while compromising on the accuracy for the less frequent items. In particular, all items x with $n_x(t) \leq \varepsilon C(t)$ can be ignored altogether as 0 is considered a good approximation for their counts. This way, at most $1/\varepsilon$ distinct items will have nonzero approximated counts.

The *quantiles* problem is to continuously maintain approximate order statistics on the distribution of the items. That is, the items are drawn from a total order, and we wish to retain a set of items $q_1, \ldots, q_{1/\varepsilon}$ such that the rank of q_i (number of input items within the sliding window that are less than q_i) is between $(i-1)\varepsilon C(t)$ and $(i+1)\varepsilon C(t)$ [12]. It is known that this is equivalent to the "prefix-count" problem, where the goal is to maintain a data structure on the sliding window such that for any given x, the number of items smaller than x can be counted within an additive error of at most $\varepsilon C(t)$.

The key insight of the forward-backward approach is that we can partition time into fixed intervals of length w, the desired window size. Now at any time, our sliding window of interests intersects two of these fixed partitions. In one of these partitions, new items are arriving only; in the other, old items are expiring only. We can consider these two windows independently, and combine approximations of each to obtain overall guarantees over the whole data. This simplifies the problem dramatically, since we now have to deal only with monotonic behavior (arrivals or departures), which, for many functions of interest, implies monotonic behavior of the desired output also. Consequently, we obtain monitoring algorithms for these problems which are based on running simple protocols assuming an unbounded window for the "forward" (arriving) part of the data, and building data structures which allow approximate computations of the desired function on the "backward" (departing) part of the data.

Problem	Communication Cost	Space Cost
Basic Counting	$O(\frac{k}{\varepsilon}\log(\varepsilon n/k))$ bits	$O(\frac{1}{\varepsilon}\log \varepsilon n)$
Heavy Hitters	$O(\frac{k}{\varepsilon}\log(\varepsilon n/k))$	$O(\frac{1}{\varepsilon}\log \varepsilon n)$
Quantiles	$O(\frac{k}{\varepsilon}\log^2(1/\varepsilon)\log(n/k))$	$O(\frac{1}{\varepsilon}\log^2(1/\varepsilon)\log n)$

Figure 2: Summary of Results. All bounds are in terms of words unless specified otherwise.

A summary of our results appears in Figure 2. Here, the communication cost is measured as the total amount of communication between all k sites and the central coordinator site, as a function of n, the number of observations in each window, and ε, the approximation parameter. We also list the space required local to each site to run the protocol.

2. REFERENCES

[1] C. Arackaparambil, J. Brody, and A. Chakrabarti. Functional monitoring without monotonicity. In *ICALP*, 2009.

[2] C. Busch, S. Tirthapura, and B. Xu. Sketching asycnhronous streams over sliding windows. In *ACM PODC*, 2006.

[3] H.-L. Chan, T.-W. Lam, L.-K. Lee, and H.-F. Ting. Continuous monitoring of distributed data streams over a time-based sliding window. In *STACS*, 2010.

[4] G. Cormode and M. Garofalakis. Sketching streams through the net: Distributed approximate query tracking. In *VLDB*, 2005.

[5] G. Cormode, M. Garofalakis, S. Muthukrishnan, and R. Rastogi. Holistic aggregates in a networked world: Distributed tracking of approximate quantiles. In *ACM SIGMOD* , 2005.

[6] G. Cormode, S. Muthukrishnan, and K. Yi. Algorithms for distributed, functional monitoring. In *ACM-SIAM SODA*, 2008.

[7] G. Cormode, S. Muthukrishnan, K. Yi, and Q. Zhang. Optimal sampling from distributed streams. In *ACM PODS*, 2010.

[8] Y. Emek and A. Korman. Efficient threshold detection in a distributed environment. In *ACM PODC*, 2010.

[9] P. Gibbons and S. Tirthapura. Distributed streams algorithms for sliding windows. In *ACM SPAA*, 2002.

[10] R. Keralapura, G. Cormode, and J. Ramamirtham. Communication-efficient distributed monitoring of thresholded counts. In *ACM SIGMOD* , 2006.

[11] F. Kuhn, T. Locher, and S. Schmid. Distributed computation of the mode. In *ACM PODC*, 2008.

[12] B. Patt-Shamir. A note on efficient aggregate queries in sensor networks. In *ACM PODC*, 2004.

[13] I. Sharfman, A. Schuster, and D. Keren. A geometric approach to monitoring threshold functions over distributed data streams. In *ACM SIGMOD* , 2006.

[14] K. Yi and Q. Zhang. Optimal tracking of distributed heavy hitters and quantiles. In *ACM PODS*, 2009.