

Bandwidth Efficient Distributed Monitoring Schemes

Yuval Alfassi
Computer Science Department
University of Haifa
Haifa, Israel
yuvalalfassi@gmail.com

Dani Keren
Computer Science Department
University of Haifa
Haifa, Israel
dkeren@cs.haifa.ac.il

Abstract— The distributed monitoring problem which tracks changes of a function over dynamic data in a distributed setting is a challenging problem in numerous real-world modern applications. Several monitoring schemes were proposed as an approach to this problem.

Here, we propose new distributed monitoring schemes which monitor the distributed function using much less communication bandwidth. Existing schemes send high dimensional vectors between the servers while we propose two methods for reducing the communication into one scalar and also two sketch-based methods which are a trade-off(...)

I. INTRODUCTION

Monitoring a function over large amount of dynamically changes data which changes in a distributed fashion is a common computer science challenge. Whether its monitoring features of large distributed dynamic graphs [*] or multiple air pollution sensors [**], innovative approaches had to be developed in order to cope with the difficulties.

The need of minimizing both the bandwidth and the processing power is expressed in [*]; good examples are internet of things objects which require communication with one another [**], or air pollution sensors which distributively have to determine whether there's air pollution [***].

Previous works were made on monitoring the entropy of distributed streams [*,], monitoring inner product, the number of triangles of distributed dynamic graphs..

The distributed monitoring problem is defined as follows:

- 1) There are n servers and a coordinator, server $server_i$ has vector v_i . the global vector is

$$v = \frac{1}{n} \sum_{i=1}^n v_i$$

- 2) A function f is monitored so its ε -approximated: for a threshold T ,

$$(1 - \varepsilon)f(v) \leq f(v) \leq (1 + \varepsilon)f(v)$$

The monitoring is done by bounding the function f by a 'convex bound' function [*] – a function c which bounds the function from above (below) and is used for the upper (lower) threshold monitoring.

Given convex f

The monitoring objective is to determine whether:

$$f(v) \leq T \quad (1)$$

II. PREVIOUS WORK

III. VECTOR SCHEME

The *Vector Scheme*'s idea is to balance the data vectors of the servers. when a server's local data vector gets out of the function's bound, this scheme would like to balance it with other data vector. It would be done by incorporating *slack vectors*, namely, $server_i$ would maintain a slack \vec{s}_i . It's important to note that the *Vector Scheme* makes sure that (III) $\sum \vec{s}_i = \vec{0}$ In order to take into consideration these *slacks*, a server raises a violation and initiates a communication channel with the coordinator if $f(v_i + s_i)$ exceeds the threshold; specifically, for a lower bound threshold, when $f(v_i + s_i) \leq T$. This ensures that whenever all the local constraint hold, the global constraint mentioned in section [1]. proof due to (I), (III):

$$\begin{aligned} f(v) &= f\left(\frac{1}{n} \sum_{i=0}^n v_i\right) = \frac{1}{n} f\left(\sum_{i=0}^n (v_i + s_i)\right) \\ &\leq \frac{1}{n} \sum_{i=0}^n f(v_i + s_i) \leq \frac{1}{n} (n \cdot T) = T \end{aligned} \quad (2)$$

When a violation occurs, i.e. $f(v_i + s_i) > T$ at a certain server, (2) cannot longer be proven so a *violation resolution* has to occur. In the *violation resolution* phase, the slack vectors are balanced so $f(v_i + s_i)$ would get inside the convex zone. When a server detects a local violation, it sends its local vector $(v_i + s_i)$ to the coordinator, which polls other servers for their local vector as well. When the average of those vectors is inside the convex zone, i.g. $f(E(v_i + s_i)) \leq T$. after that, the coordinator sends the average vector $(k - \text{number of polled nodes plus violated node}) - \frac{1}{k} \sum (v_i + s_i)$ to the polled nodes as well as the violated node, which update their slack to be $s_i \leftarrow -v_i + \frac{1}{k} \sum (v_i + s_i)$. Note that condition (III) still holds. When all the nodes are polled and the average vector still isn't inside the convex zone, a *full sync* has to be done, the real value of $f(v)$ is known, so the upper bound and lower bound reset to $(1 \pm \varepsilon)f(v)$ and the monitoring continues.

The *Vector Scheme* is extremely wasteful in communication bandwidth. Considaring that the data vectors are of very high dimension, this scheme sends the whole vectors.

IV. VALUE SCHEME

The *Value Scheme* is distributed monitoring scheme done by reducing the bandwidth from sending a whole vector to one scalar. The whole monitoring scheme is being reduced to one scalar passed. Though, it might have more false alarms than the *Vector Scheme*, and thus require more *full syncs*. The scalar that will be passed will represent the *value* of the convex bound function.

The *Value Scheme* maintains local scalar slack values; $server_i$ maintains the scalar λ_i . like *Vector Scheme*'s (III), $\sum \lambda_i = 0$. Here, the local server's constraint is whether $f(v_i) + \lambda_i \leq T$; hence, if all the local constraints are being held, the global constraint (1) will be held as well:

$$\begin{aligned} f(v) &= f\left(\frac{1}{n} \sum_{i=0}^n v_i\right) \leq \frac{1}{n} \sum_{i=0}^n f(v_i) \\ &= \frac{1}{n} \sum_{i=0}^n (f(v_i) + \lambda_i) \leq \frac{1}{n} (n \cdot T) = T \end{aligned} \quad (3)$$

When a violation occurs, i.e. $f(v_i) + \lambda_i > T$, proof 3 cannot longer hold, thus a violation resolution protocol has to be initiated.

A. Violation Resolution

The violation resolution protocol goes as follows: the violated server $server_i$ sends its local value $f(v_i) + \lambda_i$ which exceeds the threshold. The coordinators tries to 'balance' this scalar value by gradually polling other servers for their $f(v_j) + \lambda_j$ value. Let k be the number of polled servers plus the violated server, then, when $\frac{1}{k} \sum (f(v_i) + \lambda_i) \leq T$ the violation will be resolved by sending $\frac{1}{k} \sum (f(v_i) + \lambda_i)$ to the polled servers and the violated server, which in turn, will set their local scalar slack to $\lambda_i \leftarrow -v_i + \frac{1}{k} \sum (f(v_i) + \lambda_i)$. Its important to note that (*) and (**) are still holding and (***) holds as well.

V. DISTANCE SCHEME

The *Distance Scheme* is one more *distributed monitoring scheme* that relies on passing a single when communicating. The scalar that will be passed will represent the *distance* of the local vector at the server to the boundary of convex bound. This scheme is based on the *Distance Lemma* which is proved below.

A. The Distance Lemma

Let v_i be vectors and let C be a convex body. If the sum of the distances of the vectors inside C is greater than the sum of the distances to the boundary of the vectors from outside, that the average vector $\frac{1}{n} \sum v_i$ is also inside the convex set. proof:

- 1) Let p_1, \dots, p_k be points inside the convex body C and l_1, \dots, l_k their distances to the boundary.
- 2) Let q_1, \dots, q_m be points outside the convex body C and d_1, \dots, d_m the distance vectors from the boundary to the

points while c_1, \dots, c_m are the points on the boundary which are the source of the distance vectors d_i .

3) a

VI. SKETCHED DATA RESOLUTION

VII. SKETCHED CHANGE RESOLUTION

VIII. EXPERIMENTAL RESULTS

IX. CONCLUSIONS

X. SUMMARY

REFERENCES

- [1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.