Distributed Geometric Query Monitoring Using Prediction Models

NIKOS GIATRAKOS, ANTONIOS DELIGIANNAKIS, and MINOS GAROFALAKIS,
Technical University of Crete
IZCHAK SHARFMAN and ASSAF SCHUSTER, Technion

Many modern streaming applications, such as online analysis of financial, network, sensor, and other forms of data, are inherently distributed in nature. An important query type that is the focal point in such application scenarios regards actuation queries, where proper action is dictated based on a trigger condition placed upon the current value that a monitored function receives. Recent work [Sharfman et al. 2006, 2007b, 2008] studies the problem of (nonlinear) sophisticated function tracking in a distributive manner. The main concept behind the geometric monitoring approach proposed there is for each distributed site to perform the function monitoring over an appropriate subset of the input domain. In the current work, we examine whether the distributed monitoring mechanism can become more efficient, in terms of the number of communicated messages, by extending the geometric monitoring framework to utilize prediction models. We initially describe a number of local estimators (predictors) that are useful for the applications that we consider and which have already been shown particularly useful in past work. We then demonstrate the feasibility of incorporating predictors in the geometric monitoring framework and show that prediction-based geometric monitoring in fact generalizes the original geometric monitoring framework. We propose a large variety of different prediction-based monitoring models for the distributed threshold monitoring of complex functions. Our extensive experimentation with a variety of real datasets, functions, and parameter settings indicates that our approaches can provide significant communication savings ranging between two times and up to three orders of magnitude, compared to the transmission cost of the original monitoring framework.

Categories and Subject Descriptors: H.4 [**Information Systems Applications**]: Miscellaneous; H.2.4 [**Database Management**]: Systems—*Query processing*

General Terms: Algorithms, Experimentation, Performance

Additional Key Words and Phrases: Continuous distributed monitoring, data streams, prediction models

ACM Reference Format:
Nikos Giatrakos, Antonios Deligiannakis, Minos Garofalakis, Izchak Sharfman, and Assaf Schuster. 2014. Distributed geometric query monitoring using prediction models. ACM Trans. Datab. Syst. 39, 2, Article 16 (May 2014), 42 pages.
DOI: http://dx.doi.org/10.1145/2602137

1. INTRODUCTION

A wide variety of modern applications rely on the *continuous* processing of vast amounts of arriving data in order to support decision-making procedures in real time. Examples include network administration, stock market analysis, environmental,

This work was partially supported by the European Commission under ICT-FP7-FERARI-619491 (Flexible Event pRocessing for big dAta aRchitecture).
Authors' addresses: N. Giatrakos (corresponding author), A. Deligiannakis, and M. Garofalakis, School of Electronic and Computer Engineering, Technical University of Crete, University Campus – Kounoupidiana, Chania 73100, Greece; email: ngiatrakos@softnet.tuc.gr; I. Sharfman and A. Schuster, Computer Science Department, Technion – Israel Institute of Technology, Haifa 32000, Israel.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2014 ACM 0362-5915/2014/05-ART16 $15.00
DOI: http://dx.doi.org/10.1145/2602137

surveillance, and other application scenarios. These settings are, more often than not, inherently *distributed* in nature. For instance, consider the case of a network operation center where data is produced by hundreds or thousands of routers [Cormode and Garofalakis 2005, 2008; Cormode et al. 2005] or the case of environmental as well as control applications where wireless sensor network adoption has become of great importance [Madden et al. 2005].

Due to the distributed nature of data production in the aforementioned scenarios, the major challenge confronted by algorithms dealing with their manipulation is to reduce communication [Cormode and Garofalakis 2005, 2008; Cormode et al. 2005; Sharfman et al. 2006, 2007b, 2008; Das et al. 2004]. This happens because the central collection of data is not feasible in large-scale applications. Furthermore, in the case of sensor network deployments, central data accumulation results in depleting the power supply of individual sensors, reducing the network lifetime [Madden et al. 2005].

An important query type that is of the essence in the aforementioned fields regards the monitoring of a trigger condition defined upon the range of values that a function of interest receives [Sharfman et al. 2006, 2007b, 2008; Huang et al. 2006, 2007; Jain et al. 2004; Keralapura et al. 2006]. For instance, in order to perform spam detection on a number of dispersed mail servers, algorithms base their decisions on whether the value of the information gain function globally exceeds a given threshold [Sharfman et al. 2007b]. Moreover, in the example of the network operation center, Denial-of-Service (DoS) attacks are detected by attempting to pinpoint strangely high (based on a given threshold) number of distinct source addresses routing packets across various destinations within the network [Das et al. 2004].

Recently, the work in Sharfman et al. [2006, 2007b] has introduced a generic paradigm for monitoring general (nonlinear) functions defined over the average of local vectors maintained at distributed sites. Their proposed geometric approach essentially monitors the area of the input domain where the average vector may lie, rather than monitoring the function's value itself. The monitoring is performed in a distributed manner, by assigning each node a monitoring zone, expressed as a hypersphere, which is nothing more than a subset of the input domain where the average vector may lie. Communication is shown to be necessary only if at least one site considers it likely that the condition of the monitored function may have changed since the last communication between the sites.

In this work, we examine the potentials of a simple (yet powerful), easy to locally maintain approach in order to further reduce transmissions towards the central source. In particular, we foster prediction models so as to describe the evolution of local streams. The adoption of prediction models has already been proven beneficial in terms of bandwidth preservation [Cormode and Garofalakis 2005, 2008; Cormode et al. 2005] in distributed settings. Initially, we extend the geometric monitoring framework of Sharfman et al. [2006, 2007b] and illustrate how it can incorporate predictors, in order to forecast the evolution of local data vectors of sites. We exhibit the way the geometric monitoring framework is modified to encompass constructed predictors and identify the peculiarities occurring upon predictors' adoption. In contrast to the findings of prior works [Cormode and Garofalakis 2005, 2008; Cormode et al. 2005], we prove that the mere utilization of local predictions is hardly adequate to guarantee communication preservation even when predictors are quite capable of describing local stream distributions. We then proceed by establishing a theoretically solid monitoring framework that incorporates conditions potentially leading to fewer contacts with the central source. Eventually, we develop a number of mechanisms, along with extensive speculative analysis, that relax the previously introduced framework, base their function on simpler criteria, and in practice yield significant transmission reduction. Our main contributions are as follows.

—We introduce the adoption of prediction models in the setting of tracking general, nonlinear functions utilizing the geometric approach [Sharfman et al. 2006, 2007b]. We exhibit the way prediction models can be locally adopted by sites and we show the characteristics they attribute to the geometric approach. We then illustrate that the initial geometric monitoring framework of Sharfman et al. [2006, 2007b] is a special case of our more general prediction-based geometric monitoring framework.
—We point out the failure of conventional notions of good predictors to be applied in this setting and manage to establish a solid theoretic framework consisting of conditions that make it more likely for prediction models to exhibit reduced bandwidth consumption.
—We expose a number of novel tracking mechanisms relaxing the previously (hard to verify in a distributed manner) identified conditions. Using the simplest possible primitives regarding prediction models' behavior, we thoroughly study a number of simplified alternative tracking techniques possessing the potentials for communication preservation.
—We present a probabilistic analysis on the expected performance of our simplified alternative tracking mechanisms. This analysis describes an ideal case where continuous knowledge of global statistics is available. As, in practice, this type of statistics requires constant central data collection, we then use the extracted intuition to propose empirical decision-making procedures with respect to the best (i.e., the one that protracts central data collection) alternative choice.
—We look into extensions of our techniques for approximate function monitoring scenarios as the core of more generic query answering procedures. We come up with problem transformations which render accurate predictors capable of reducing bandwidth consumption. In our study, we introduce novel monitoring techniques tailored for the new setup and accordingly compare their function with the rest of the alternative schemes previously developed in our work.
—We present an extensive experimental analysis using a variety of real datasets, parameters, and functions of interest. Our evaluation shows that our approaches can provide significant communication load reduction with savings ranging from two times and in some cases reaching three orders of magnitude compared to the transmission cost of the original bounding algorithm.

Article Overview. This article proceeds as follows. In the next section we present related work. In Section 3 we formally present the geometric monitoring framework and some exemplary prediction models useful for the applications that we consider. In Section 4 we first explain the motivation for adopting predictors and we then demonstrate how prediction models can be incorporated within the geometric monitoring framework. We also demonstrate that simply using "good" predictors, that is, those approximating the local data maintained at nodes more accurately than a simple model that assumes them to be static, may not suffice for efficient distributed monitoring applications. We then seek to determine in Section 5 conditions and properties that predictor-based monitoring should exhibit in order to more likely result in reduced communication. We thus propose the notion of strong predictor-based monitoring models and examine their properties. In Section 6 we argue that efficiently, in terms of required bandwidth, verifying the properties of strong predictor-based monitoring models in a distributed environment seems infeasible. We thus propose, based on relaxed versions of the previously identified conditions, several practical alternatives built upon simpler assumptions on the ability of prediction models to describe incoming data distributions. Given many possible predictor-based models, the questions that naturally arise involve the choice of the appropriate model to use in each application and dataset. In Section 7 we first investigate whether such a decision can be made based on

probabilistic arguments and we explain why such a decision is hard to be made in such a way. We thus propose our CAA algorithm that automatically switches its decisions on which model to use at each time instance. Section 8 explains how our techniques can be adapted to efficiently support prediction-based approximate query answering procedures. We also explain under which conditions the monitoring process at each node can become very simple and computationally much more efficient. Our experimental analysis is incorporated in Section 9. Eventually, Section 10 includes concluding remarks.

2. RELATED WORK

Recently, substantial efforts have been devoted on tracking and querying distributed data streams [Cormode and Garofalakis 2007]. The geometric monitoring framework that is leveraged by our approaches was introduced in Sharfman et al. [2006, 2007b] and was later enhanced in Sharfman et al. [2008] and Keren et al. [2012]. The optimizations proposed in Sharfman et al. [2008] are orthogonal to our approaches, but note that the techniques of Sharfman et al. [2008] either require data to conform with a multivariate normal distribution or entail a number of solutions to a series of optimization problems that may increase the computational load. The latter renders their adoption unaffordable in resource-constrained environments such as Sharfman et al. [2007a] and Burdakis and Deligiannakis [2012]. On the contrary, our approaches are based on simple predictors' adoption that remain adaptable to changing data distributions and are easy to maintain even when resource constraints exist. In other work related to the geometric monitoring approach, Sharfman et al. [2007a] discuss an application of the framework of Sharfman et al. [2006, 2007b] to clustered sensor network settings, while Burdakis and Deligiannakis [2012] show how the tracking scheme can be utilized so as to detect outliers produced by motes. The recent work of Sagy et al. [2010] adopts the geometric approach and proposes a tentative bound algorithm to monitor threshold queries in distributed databases (rather than distributed data streams) for functions with bounded deviation, while Sagy et al. [2011] enhance ideas of Sagy et al. [2010] to answer vectorial top-k aggregation queries over distributed databases. Moreover, the work in Garofalakis et al. [2013] couples sketch summaries with the geometric monitoring framework focusing on specific monitored functions such as general inner products (i.e., join aggregates), special cases of L_2-norms (i.e., self-join sizes) and range aggregates (e.g., quantiles, histograms, wavelets, and heavy-hitters over the streams).

Prediction models in the context of distributed data streams have already been fostered in previous work to monitor one-dimensional quantiles [Cormode et al. 2005] and randomized sketch summaries [Cormode and Garofalakis 2008]. Their adoption has been proven beneficial in terms of reducing the communication burden. Contrary to previous approaches, our focus is on the benefits they can provide in the context of the geometric monitoring framework for tracking nonlinear threshold functions.

In related work regarding distributed trigger monitoring, Keralapura et al. [2006] provide a framework for monitoring thresholded counts over distributed data streams, while Jain et al. [2004] design techniques that decompose the problem of detecting when the sum of a distributed set of variables exceeds a given threshold. Based on Jain et al. [2004], anomaly detection techniques are studied in Huang et al. [2006, 2007]. The recent work of Cormode et al. [2011] provides upper and lower communication bounds for approximate monitoring of thresholded F_p moments, with $p = 0, 1, 2$.

Other works focus on tracking specific types of functions over distributed data streams. The work of Olston et al. [2003] considers simple aggregation queries over multiple sources, while Babcock and Olston [2003] focus on monitoring top-k values.

Table I. Notation Used

Symbol	Description
n	The number of sites
S_i	The i-th site
t_s	Timestamp of the last synchronization
$v(t)$	Global measurements vector at time t ($\sum_{i=1}^{n} w_i v_i(t) / \sum_{i=1}^{n} w_i$)
$e(t)$	Estimate vector at time t (equal to $v(t_s)$)
$e^p(t)$	The predicted estimate vector ($\sum_{i=1}^{n} w_i v_i^p(t) / \sum_{i=1}^{n} w_i$)
$v_i(t)$	Local measurements vector at S_i at time t
w_i	Number of data points at S_i
$u_i(t)$	Drift vector (equals to $e(t) + v_i(t) - v_i(t_s)$)
$v_i^p(t)$	Local predictor of S_i at time t
$u_i^p(t)$	Prediction deviation vector ($e^p(t) + v_i(t) - v_i^p(t)$)
$B_c^{\|r\|}$	Local constraint (ball) centered at c with radius $\|r\|$
D_e	Radius of maximum sphere (ball), centered at e, that can be inscribed without violating the threshold surface
D_{e^p}	Radius of maximum sphere (ball), centered at e^p, that can be inscribed without violating the threshold surface

Furthermore, Das et al. [2004] monitor set-expression cardinalities in a distributed system using a scheme for charging local changes against a single site's error tolerance. Yi and Zhang [2013] consider the problem of tracking heavy-hitters and quantiles in a distributed manner, establishing optimal algorithms to accomplish the task. Cormode et al. [2007] study the problem of clustering distributed data streams, while Zhang et al. [2008] generalize the previous approach to hierarchical environments. Eventually, the latest work of Gupta et al. [2013] studies the problem of distributed monitoring of thresholded ratio functions. The tracked ratio function is a simple ratio of sums $\sum_{i=1}^{n} n_i / \sum_{i=1}^{n} d_i$ where n_i, d_i refer to data items possessed by individual sites. On the contrary, the geometric approach can monitor much more complex ratio functions, examples of which are utilized in our experimental evaluation.

In the conference version of this article [Giatrakos et al. 2012], we presented the initial framework for monitoring nonlinear functions using prediction models. In this work we build on the work of Giatrakos et al. [2012] mainly by extending this framework to perform approximate query answering (Section 8). We also enhance the discussion on the selection of the model to use at each time by adding a theoretical viewpoint on this process (Section 7.1). We further manage to elaborate on the proofs of Corollary 6.3 and Proposition 6.4. Finally, we enhanced our experimental results by individually exploring the proposed tracking mechanisms' performance (Section 9.2) by investigating the impact of the accuracy of the predictors in the proposed function monitoring schemes utilizing appropriate synthetic datasets (Section 9.4) and by validating (Section 9.5) the usability of our newly proposed techniques in approximate query answering scenarios.

3. PRELIMINARIES

In this section we first provide helpful background work related to function monitoring using the geometric approach. We then describe local stream predictors which have been utilized in past work. The notation used in this article appears in Table I.

3.1. The Geometric Monitoring Framework

As in previous works [Cormode et al. 2005; Sharfman et al. 2007b, 2008; Cormode and Garofalakis 2005, 2008], we assume a distributed, two-tiered setting, where data arrives continuously at n geographically dispersed sites. At the top tier, a central coordinator exists that is capable of communicating with every site, while pairwise site communication is only allowed via the coordinating source.

Each site S_i, $i \in [1..n]$ participating at the bottom tier receives updates on its local stream and maintains a d-dimensional local measurements vector $v_i(t)$. The *global measurements vector* $v(t)$ at any given timestamp t is calculated as the weighted average of $v_i(t)$ vectors, $v(t) = \frac{\sum_{i=1}^{n} w_i v_i(t)}{\sum_{i=1}^{n} w_i}$, where $w_i \geq 0$ refers to the weight associated with a site. Usually, w_i corresponds to the number of data points received by S_i [Sharfman et al. 2006]. Our aim is to continuously monitor whether the value of a function $f(v(t))$, defined upon $v(t)$, lies above/below a given threshold T. We use the term *threshold surface* to denote the area of the input domain where $f(v(t)) = T$.

During the monitoring task using the geometric approach [Sharfman et al. 2006, 2007b], the coordinator may request that all sites transmit their local measurements vectors and subsequently calculates $v(t)$, performs the required check on $f(v(t))$, and transmits the $v(t)$ vector to all sites. The previous process is referred to as a *synchronization* step. Let $v_i(t_s)$ denote the local measurements vector that S_i communicated during the last synchronization process at time t_s. The global measurements vector computed during a synchronization step is denoted as the *estimate vector* e, where $e = \sum_{i=1}^{n} w_i v_i(t_s) / \sum_{i=1}^{n} w_i$.

After a synchronization, sites keep up receiving updates of their local streams and accordingly maintain their $v_i(t)$ vectors. At any given timestamp, each site S_i individually computes $v_i(t) - v_i(t_s)$ and the local *drift vector* $u_i(t) = e + (v_i(t) - v_i(t_s))$. Since

$$v(t) = \frac{\sum_{i=1}^{n} w_i v_i(t)}{\sum_{i=1}^{n} w_i} = e + \frac{\sum_{i=1}^{n} w_i (v_i(t) - v_i(t_s))}{\sum_{i=1}^{n} w_i} = \frac{\sum_{i=1}^{n} w_i u_i(t)}{\sum_{i=1}^{n} w_i},$$

$v(t)$ constitutes a convex combination of the drift vectors. Consequently, $v(t)$ will always lie in the convex hull formed by the $u_i(t)$ vectors: $v(t) \in Conv\ (u_1(t),\ \ldots,\ u_n(t))$, as depicted in Figure 1.

Please note that each site can compute the last known value of the monitored function as $f(e)$ and can thus determine whether this value lies above/below the threshold T. Since $v(t) \in Conv\ (u_1(t),\ \ldots,\ u_n(t))$, if the value of the monitored function in the entire convex hull lies in the same direction (above/below the threshold T) as $f(e)$, then it is guaranteed that $f(v(t))$ will lie in that side. In this case, the function will certainly not have crossed the threshold surface. The key question is: How can the sites check the value of the monitored function in the entire convex hull, since each site is unaware of the current drift vectors of the other sites? This test can be distributively performed as described in Theorem 3.1, while an example (in two dimensions) is included in Figure 1.

THEOREM 3.1 ([SHARFMAN ET AL. 2006, 2007B]). *Let $x, y_1, \ldots, y_n \in R^d$ be a set of d-dimensional vectors. Let $Conv\ (x,\ y_1,\ \ldots,\ y_n)$ be the convex hull of $x, y_1, \ldots, y_n$. Let $B^{\|\frac{x-y_i}{2}\|}_{\frac{x+y_i}{2}}$ be a ball centered at $\frac{x+y_i}{2}$ with a radius of $\|\frac{x-y_i}{2}\|$ that is, $B^{\|\frac{x-y_i}{2}\|}_{\frac{x+y_i}{2}} = \{z \in R^d : \|z - \frac{x+y_i}{2}\| \leq \|\frac{x-y_i}{2}\|\}$. Then, $Conv\ (x,\ y_1,\ \ldots,\ y_n) \subset \cup_{i=1}^{n} B^{\|\frac{x-y_i}{2}\|}_{\frac{x+y_i}{2}}$.*

With respect to our previous discussion, x corresponds to e while y_i vectors refer to the drift vectors $u_i(t)$. Hence, sites need to compute their local constraints in the form of $B^{\|\frac{e-u_i(t)}{2}\|}_{\frac{e+u_i(t)}{2}}$ and independently check whether a point within these balls may cause a

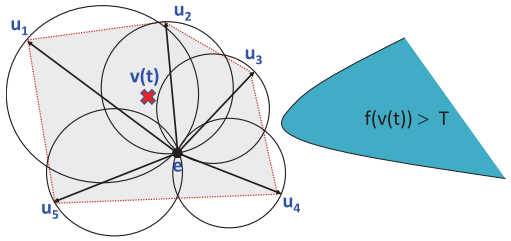


Fig. 1. Demonstration of the geometric framework rationale. $Conv\ (u_1, \ldots, u_n)$ is depicted in gray, while the actual position of e and the current $v(t)$ are shown as well. Black spheres refer to the local constraints constructed by sites to assess possible threshold crossing. $v(t)$ is guaranteed to lie within the union of these locally constructed spheres. Since none of the spheres crosses the threshold surface, the sites are certain that $f(v(t))$ and $f(e)$ are at the same side relative to the threshold T. Hence, no synchronization needs to be performed.

threshold crossing. If this indeed is the case (an event termed as a *local violation*), a synchronization step takes place. Note that, since $Conv\ (e, u_1, \ldots, u_n)$ is a subset of the union of local ball constraints, the framework may produce a synchronization in cases where the convex hull has not actually crossed the threshold surface (false positives).

In summary, each site in the geometric monitoring framework manages to track a subset of the input domain. The overall approach achieves communication savings since the coordinator needs to collect the local measurement vectors of the sites only when a site locally detects (in its monitored area of the input domain) that a threshold crossing may have occurred.

3.2. Local Stream Predictors

We now outline the properties of some prediction model options that have already been proven useful in the context of distributed data streams [Cormode and Garofalakis 2008; Cormode et al. 2005]. Please note, beforehand, that the concept of their adoption is to keep such models as simple as possible, and yet powerful enough to describe local stream distributions. It can easily be conceived that more complex model descriptors can be utilized, which, however, incur extra communication burden when sites need to contact the coordinating source [Cormode and Garofalakis 2008; Cormode et al. 2005]. In our setting, this translates to an increased data transmission overhead during each synchronization step. In our discussion hereafter, we utilize the term *predictor* to denote a prediction estimator for future values of a local measurements vector. Using a similar notation to the one of Section 3.1, we employ $v_i^p(t)$ to denote the prediction for the local measurements vector of site S_i at timestamp t.

The Static Predictor. The simplest guess a site may take regarding the evolution of its local measurements vector is that its coordinates will remain unchanged with respect to the values they possessed in the last synchronization: $v_i^p(t) = v_i(t_s)$. It is also evident that this predictor is trivial to maintain in both the sites and the coordinator. Moreover, it requires no additional information to be communicated towards the coordinator upon a synchronization step. Using the static predictor, in the absence of a synchronization step, the coordinator estimates that $v(t) = e$.

The static predictor may be a good choice only in settings where the evolution of the values in each local measurements vector is unpredictable, or local measurements vectors change rarely.

The Linear Growth Predictor. The next simple, but less restrictive, assumption that can be made is that local vectors will scale proportionally with time. In particular, $v_i^p(t) = \frac{t}{t_s} v_i(t_s)$ which is the only calculation individual sites and the coordinating source

Table II. Local Stream Predictors' Summary

Predictor	Info.	Pred. Local Vector (v_i^p)
Static	$\emptyset$	$v_i(t_s)$
Linear Growth	$\emptyset$	$\frac{t}{t_s} v_i(t_s)$
Velocity/Acceleration	vel_i	$v_i(t_s) + (t - t_s)vel_i + (t - t_s)^2 accel_i$

need to perform in order to derive an estimation of $v_i(t)$ at any given time. Please note that, using this predictor, the best guess that a coordinator can make for the value of $v(t)$ is equal to $\frac{t}{t_s} v(t_s) = \frac{t}{t_s} e$. As with the static predictor, the linear growth predictor requires no additional information to be transmitted upon a synchronization.

We can deduce that the linear growth predictor is built on the assumption that $v_i(t)$ vectors evolve, but that their evolution involves no direction alterations. Consequently, it can be adopted so as to approximate local streams in which $v_i(t)$ vectors' coordinates are expected to uniformly increase by a time-dependent factor.

The Velocity/Acceleration Predictor. The Velocity/Acceleration (*VA*) predictor is a much more expressive predictor. VA employs additional vectors that attempt to capture both the scaling and directional change that $v_i(t)$ may undertake. More precisely, in the *VA* predictor the future value of the local measurements vector is estimated as $v_i^p(t) = v_i(t_s) + (t - t_s)vel_i + (t - t_s)^2 accel_i$. Since the velocity vel_i and the acceleration $accel_i$ of the local stream are capable of expressing both possible types of $v_i(t)$ alterations, it provides an enriched way to approximate its behavioral pattern.

In a way similar to Cormode and Garofalakis [2008], when a synchronization is about to take place, S_i is required to compute the velocity vector vel_i utilizing a window of the W most recent updates it received. Given that window, the velocity vector can be calculated by computing the overall disposition as the difference between $v_i(t)$ and the local vector instance corresponding to the first position of the window.[1] Scaling this outcome by the time difference between the window extremes provides vel_i. In addition, the $accel_i$ value can be computed as the difference between the current velocity and corresponding velocity calculated in the previous synchronization. Scaling the previous result by $1/(t - t_s)$ computes a proper $accel_i$ vector. Additional approaches based on use of vel_i and $accel_i$ values can be found in Cormode and Garofalakis [2008].

It is easy to see that the flexibility provided by the *VA* predictor comes at the cost of the transmission of vel_i (along with $v_i(t)$) during each synchronization. We note that $accel_i$ does not need to be communicated to the coordinator, since the coordinator is already aware of the previously computed velocity vectors of each site.

Table II summarizes the described predictor characteristics. It is important to emphasize that the prediction-based monitoring framework described in the next sections can utilize any predictor and is thus not restricted to the predictors presented in this section.

4. PREDICTION-BASED GEOMETRIC MONITORING

In this section we first motivate the need to incorporate predictors in the geometric monitoring framework and then demonstrate how this can be achieved. We then illustrate that the initial geometric monitoring framework of Sharfman et al. [2006, 2007b] is a special case of our more general prediction-based geometric monitoring framework. Subsequently, we define the notion of a good predictor and demonstrate

[1]Please note that each update may not arrive at each timestamp. Thus, the timestamp of the first update in the window may in general be different than $t - W + 1$.

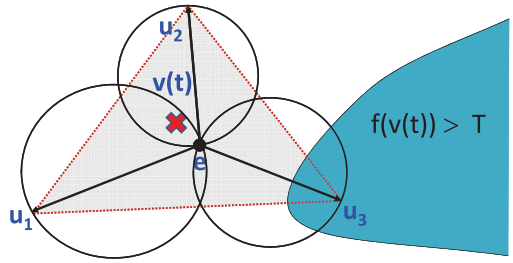


Fig. 2. Motivation for incorporating predictors in the geometric approach. The drift vector u_3 would cause a synchronization in the original framework. However, if the changes could have been predicted, then the coordinator could have expected the value of the estimate vector to actually be close to e and far from the threshold surface.

that good predictors lead to monitoring a smaller subset of the domain space, thus potentially leading to fewer synchronizations and hence fewer transmitted messages.

Motivation for Predictors. Figure 2 demonstrates a motivating example of why it may be beneficial to incorporate predictors in the geometric monitoring framework. In the illustrated example, the drift vector u_3 has crossed the threshold surface. Based on their definition, the direction of each drift vector essentially depicts how the values of the corresponding local measurements vector have changed since the last synchronization. Using the geometric monitoring approach, a synchronization will take place because S_3 will detect a threshold crossing. A plausible question is: Could we avoid such a synchronization step, if the changes in the values of the three local measurements vectors could have been predicted fairly accurately? For example, if we could have predicted the change (drift) in the local measurements vectors of each site fairly accurately, then we would have determined that $v(t)$ has probably not moved closer to the threshold surface and thus avoid the synchronization step. The preceding example motivates the need for prediction-based geometric monitoring.

How to Incorporate Predictors. As explained in Section 3.2, the coordinator can receive, during a synchronization step, information regarding the predicted local measurements vector $v_i^p(t)$ of each site. Thus, the coordinator will be able to compute an estimation of $v(t)$ provided by the local predictors as $e^p(t) = \frac{\sum_{i=1}^n w_i v_i^p(t)}{\sum_{i=1}^n w_i}$, which we will term as the *predicted estimate vector*. Based on $e^p(t)$, we now show that the coordinator can continuously check the potential threshold crossings. However, in this case a synchronization is required only when $e^p(t)$ and $v(t)$ are likely to be placed in different sides of the threshold surface.

In the context of the geometric monitoring framework, we first observe that

$$v(t) = \frac{\sum_{i=1}^n w_i v_i(t)}{\sum_{i=1}^n w_i} = e^p(t) + \frac{\sum_{i=1}^n w_i (v_i(t) - v_i^p(t))}{\sum_{i=1}^n w_i} = \frac{\sum_{i=1}^n w_i u_i^p(t)}{\sum_{i=1}^n w_i},$$

where $u_i^p(t) = e^p(t) + (v_i(t) - v_i^p(t))$ denotes the vector expressing the *prediction deviation*. Thus, similar to our analysis in Section 3.1, $v(t) \in Conv\ (u_1^p(t), \ldots, u_n^p(t)) \subset \bigcup_{i=1}^n B^{\| \frac{e^p(t) - u_i^p(t)}{2} \|}_{\frac{e^p(t) + u_i^p(t)}{2}}$. Since $v(t)$ lies in the convex hull $Conv(u_1^p(t), \ldots, u_n^p(t))$, each site S_i can monitor the ball that has as endpoints of its diameter the estimated predicted vector $e^p(t)$ and its prediction deviation $u_i^p(t)$.

Please note that the geometric monitoring approach of Sharfman et al. [2006, 2007b] corresponds to utilizing a static predictor (this leads to $v_i^p(t) = v(t_s)$, $e^p(t) = e$ and

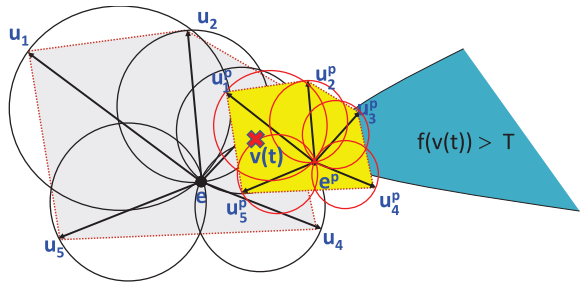


Fig. 3. The red balls demonstrate the local constraints of sites when using a sample good predictor. A good predictor results in the tighter convex hull $Conv\ (u_1^p(t), \ldots, u_n^p(t))$ (depicted in yellow). Here, fewer synchronizations are not guaranteed, since $\cup_{i=1}^{n} B^{\| \frac{e^p(t)-u_i^p(t)}{2} \|}_{\frac{e^p(t)+u_i^p(t)}{2}}$ crosses the threshold before $\cup_{i=1}^{n} B^{\| \frac{e-u_i(t)}{2} \|}_{\frac{e+u_i(t)}{2}}$.

$u_i^p(t) = u_i(t)$) and is thus a special case of our more general prediction-based monitoring framework.

Defining a Good Predictor. Upon utilizing a predictor, as long as local forecasts ($v_i^p(t)$) remain sound, we expect that they will approximate the true local vectors $v_i(t)$ to a satisfactory degree at any given timestamp. This means that each $v_i^p(t)$ will be in constant proximity to the $v_i(t)$ vector, when compared to $v_i(t_s)$. Formally, we state the following.

Property 1. A *good predictor* possesses the property

$$\|v_i(t) - v_i^p(t)\| \leq \|v_i(t) - v_i(t_s)\| \ \forall t \geq t_s.$$

Property 1 lies, implicitly or not, in the core of predictors' adoption in distributed stream settings. It expresses the notion of a useful, in terms of bandwidth consumption reduction, predictor present in previous works [Cormode and Garofalakis 2005, 2008; Cormode et al. 2005] which have managed to exhibit important improvements by exploiting the previous fact. Hence, we start by exploring the benefits of the notion of good predictors expressed by Property 1 within the geometric monitoring setting.

Predictors satisfying Property 1 yield stricter local constraints for the bounding algorithm compared to the original monitoring mechanism (Section 3.1). This happens because $\|v_i(t) - v_i^p(t)\| \leq \|v_i(t) - v_i(t_s)\| \Leftrightarrow \|u_i^p(t) - e^p(t)\| \leq \|u_i(t) - e\|$ and the radius of the constructed balls will always be smaller. An example of prediction-based monitoring is depicted in Figure 3.

Consequently, a good predictor results in the sites monitoring a tighter convex hull, namely $Conv\ (u_1^p(t), \ldots, u_n^p(t))$, than the corresponding convex hull of the original geometric monitoring framework. This yields the construction of tighter local constraints and, as already mentioned, a synchronization is required only when $e^p(t)$ is likely to be placed in a different side of the threshold surface to the one of $v(t)$. Hence, a synchronization is again caused when any ball $B^{\| \frac{e^p(t)-u_i^p(t)}{2} \|}_{\frac{e^p(t)+u_i^p(t)}{2}}$ crosses the threshold surface.

Despite the fact that this mechanism may in practice be useful, it cannot guarantee fewer synchronizations because $Conv\ (u_1^p(t), \ldots, u_n^p(t))$, although tighter, might still be placed closer than $Conv\ (u_1(t), \ldots, u_n(t))$ to the threshold surface. This in turn will cause some $B^{\| \frac{e^p(t)-u_i^p(t)}{2} \|}_{\frac{e^p(t)+u_i^p(t)}{2}}$ to cross the threshold before any $B^{\| \frac{e-u_i(t)}{2} \|}_{\frac{e+u_i(t)}{2}}$ does (Figure 3). This

observation shows that the conventional concept of good predictors fails to adapt in the current setting since it does not guarantee by itself fewer synchronizations.

5. STRONG MONITORING MODELS

Consider any point in time when all nodes are aware of a specific estimate vector e. The concluding observations of Section 4 raise a concern regarding the sufficient conditions that should be fulfilled for the predictors to yield a synchronization no sooner than the original framework. Apparently, this happens when the surface of the monitoring framework devised by the predictors is contained inside the monitored surface of the original framework. In other words, we need to define the prerequisites for constructing local constraints that are always included in $\bigcup_{i=1}^{n} B^{\| \frac{e-u_i(t)}{2} \|}_{\frac{e+u_i(t)}{2}}$ utilized by the original framework. Let $Sur(P)$ be the surface monitored by any alternative mechanism that adopts predictors while operating. A monitoring model is defined as *strong* if the following property holds.

Property 2. A *strong* predictor-based *monitoring model* possesses the property*:* $Sur(P) \subseteq \bigcup_{i=1}^{n} B^{\| \frac{e-u_i(t)}{2} \|}_{\frac{e+u_i(t)}{2}}$.

Based on its definition, a strong predictor-based monitoring model guarantees that, given an estimate vector e, it will result in a local violation no sooner than the original framework. Immediately after each time that the value of the function truly crosses the threshold, any monitoring model, including the one in the original framework, will require a synchronization at that particular time point in order to verify the threshold crossing. Thus, immediately after each true threshold crossing, strong predictor-based models are guaranteed to use the same estimate vector as the original framework and to result in a local violation no sooner than the original framework. Please note that there is no guarantee that the total number of synchronizations will be lower, since after the initial synchronization the original framework, had it been prefered instead, may have used a *different* estimate vector, due to synchronizations that may have occurred at a different time instance. However, the notion of strong predictor-based monitoring models offers us an intuitive goal that is *likely*, based on our experimental evaluation, to lead to fewer synchronizations. Our discussion on strong predictor-based monitoring models is important though, since observations derived from it will later be used in Section 6 as a building block for the introduction of some of our alternative predictor-based monitoring techniques. In particular, our discussion on the convex hull containment presented next inspires the introduction of our *average* and *safer* models while Section 5.2 provides the primitives for the loosened intersection monitoring frameworks of Section 6.2.

5.1. Containment of Convex Hulls

According to Theorem 3.1, after computing the local drift vectors and prediction deviations, we are free to choose any common *reference vector* in order to perform the monitoring task. Thus, it is not mandatory for the sites to use e and $e^p(t)$ as a common reference point in order to construct their monitoring zones. In fact, the sites could use any common point as an endpoint of the diameter of their monitoring zones.

An important observation that we prove in this section is that a predictor-based monitoring model satisfies Property 2 when: (1) every prediction deviation vector is contained in the convex hull of the estimate vector and the drift vectors defined by the original bounding algorithm, and (2) an appropriate reference vector is selected.

Before proving our observation, we first show that for any triplet of vectors $z, y, x \in R^d$, the condition $z \in B^{\|\frac{y-x}{2}\|}_{\frac{y+x}{2}}$ is equivalent to $\langle x-z, y-z\rangle \leq 0$, where the notation $\langle\cdot,\cdot\rangle$ refers to the inner product of two vectors. Whenever appropriate, we omit the temporal reference symbol (t) in the vectors to simplify the exposition.

LEMMA 5.1. *$z \in B^{\|\frac{y-x}{2}\|}_{\frac{y+x}{2}}$ if and only if $\langle x-z, y-z\rangle \leq 0$.*

PROOF. Recall that if $z \in B^{\|\frac{y-x}{2}\|}_{\frac{y+x}{2}}$, then the distance of z from the center $\frac{x+y}{2}$ does not exceed the radius $\|\frac{x-y}{2}\|$. Thus, $\|z-\frac{x+y}{2}\| \leq \|\frac{x-y}{2}\|$. Since $\|A\|^2 = \langle A, A\rangle$, simple calculations show that $\|z-\frac{x+y}{2}\| \leq \|\frac{x-y}{2}\| \Leftrightarrow \|z-\frac{x+y}{2}\|^2 \leq \|\frac{x-y}{2}\|^2 \Leftrightarrow \frac{1}{4}\langle 2z-(x+y), 2z-(x+y)\rangle - \frac{1}{4}\langle x-y, x-y\rangle \leq 0$. Recall that the inner product is both distributive, namely $\langle a+b, c\rangle = \langle a, c\rangle + \langle b, c\rangle$, and symmetric, namely $\langle a, b\rangle = \langle b, a\rangle$. Therefore

$$\begin{aligned}
\frac{1}{4}\langle 2z-(x+y), 2z-(x+y)\rangle - \frac{1}{4}\langle x-y, x-y\rangle &\leq 0 \Leftrightarrow \\
\frac{1}{4}\langle (z-x)+(z-y), (z-x)+(z-y)\rangle - \frac{1}{4}\langle (z-x)-(z-y), (z-x)-(z-y)\rangle &\leq 0 \Leftrightarrow \\
\langle z-x, z-y\rangle &\leq 0 \Leftrightarrow \\
\langle x-z, y-z\rangle &\leq 0 \ . \quad \square
\end{aligned}$$

We now proceed to prove in Lemma 5.2 that a predictor-based monitoring model that maintains each prediction deviation vector contained in the convex hull of the drift vectors defined by the original bounding algorithm is a strong predictor-based monitoring model if it also selects the same reference vector (e.g., e instead of e^p) as the original framework. A direct result is that the area monitored by the sites is a subset of the corresponding area of the original framework. This, in turn, leads to fewer synchronizations, since every time a site detects a potential threshold crossing in the predictor-based monitoring model, at least one site would also have detected the same threshold crossing (for the same estimate vector) in the original framework.

LEMMA 5.2. *Let $u_i^p \in Conv(u_1, \ldots, u_n)\ \forall i \in \{1..n\}$. Then* $B^{\|\frac{e-u_i^p}{2}\|}_{\frac{e+u_i^p}{2}} \subseteq \bigcup_{i=1}^n B^{\|\frac{e-u_i}{2}\|}_{\frac{e+u_i}{2}}$.

PROOF. For each $u_i^p \in Conv(u_1, \ldots, u_n)$ there exist $\lambda_1, \lambda_2, \ldots, \lambda_n$ such that $\lambda_i \geqslant 0$ ($i \in \{1..n\}$), $\sum_{i=1}^n \lambda_i = 1$ and $u_i^p = \sum_{i=1}^n \lambda_i u_i$. Let $h \in B^{\|\frac{e-u_i^p}{2}\|}_{\frac{e+u_i^p}{2}}$. We will show that for at least one of the u_i vectors, $h \in B^{\|\frac{e-u_i}{2}\|}_{\frac{e+u_i}{2}}$.

According to Lemma 5.1, $\langle h-e, h-u_i^p\rangle \leq 0$. Therefore

$$\langle h-e, h-u_i^p\rangle = \left\langle h-e, \sum \lambda_i h - \sum \lambda_i u_i\right\rangle = \left\langle h-e, \sum \lambda_i (h-u_i)\right\rangle = \sum \lambda_i \langle h-e, h-u_i\rangle \leq 0.$$

Since $\lambda_i \geqslant 0$, it follows that for at least one u_i with $\lambda_i > 0$, $\langle h-e, h-u_i\rangle \leq 0$, which implies (Lemma 5.1) that $h \in B^{\|\frac{e-u_i}{2}\|}_{\frac{e+u_i}{2}}$. $\square$

A trivial example of a strong predictor-based monitoring model is the static predictor which, as mentioned in Section 4, is equivalent to the original framework of Sharfman et al. [2006, 2007b]. Unfortunately, the containment constraints are not easily abided by any other chosen predictor and, even if they are, it appears hard to dictate a way

that allows sites to distributively identify that fact. We will revisit the convex hull containment issues in Section 6.1, when we introduce our *average* and *safer* predictor-based monitoring models.

5.2. Convex Hull Intersection Monitoring

An important observation that we make is that, since $v(t) \in Conv(u_1, \ldots, u_n)$ and $v(t) \in Conv(u_1^p, \ldots, u_n^p)$, these two convex hulls cannot be disjoint (Figure 3). One could thus seek ways to exploit this fact, which limits the possible locations of $v(t)$, in order to reduce the size of the monitoring zones of each site, which in turn will potentially lead to fewer detected threshold crossings. We thus seek to come up with new local constraints in the context of predictor-based monitoring models that cover the intersection of the two convex hulls and which also fulfill Property 2. To proceed towards this goal we first formally formulate an enhanced version of Property 1.

Property 3. A *universally good predictor* possesses the property $\|v_i(t) - v_i^p(t)\| \leq \|v_j(t) - v_j(t_s)\|$ for any pair of sites S_i, S_j.

In other words, for universally good predictors,

$$\min_{k=1..n} \|e - u_k\| \geq \max_{k=1..n} \left\|e^p - u_k^p\right\|. \tag{1}$$

Property 3 yields $\|u_i^p - e^p\| \leq \|u_j - e\|$ for any pair of sites S_i, S_j. The latter result is produced by simply adding as well as subtracting e^p, e to the left and right side of its inequality, respectively. The following lemma utilizes this fact to devise appropriate local constraints to be fostered at each site S_i.

LEMMA 5.3. *If Property 3 holds, each site S_i needs to examine whether $B^{\|\frac{e-u_i}{2}\|}_{\frac{e+u_i}{2}} \cap B^{\|e-u_i\|}_{e^p}$ crosses the threshold, since: $Conv(u_1, \ldots, u_n) \cap Conv(u_1^p, \ldots, u_n^p) \subset \cup_{k=1}^n B^{\|\frac{e-u_k}{2}\|}_{\frac{e+u_k}{2}} \cap B^{\|e-u_k\|}_{e^p}$.*

PROOF. We will demonstrate than any vector $h \in R^d$ that lies in $Conv\ (u_1, \ldots, u_n) \cap Conv\ (u_1^p, \ldots, u_n^p)$ is also included in at least one intersection $B^{\|\frac{e-u_k}{2}\|}_{\frac{e+u_k}{2}} \cap B^{\|e-u_k\|}_{e^p}$ of a site S_k ($k \in \{1..n\}$). What is certain is that, due to Property 3, $h \in Conv\ (u_1^p, \ldots, u_n^p) \Rightarrow h \in \cup_{k=1}^n B^{\|\frac{e^p-u_k^p}{2}\|}_{\frac{e^p+u_k^p}{2}} \Rightarrow h \in \cup_{k=1}^n B^{\|e^p-u_k^p\|}_{e^p} \Rightarrow h \in B^{\|e-u_k\|}_{e^p}$. Since h is definitely contained as well in at least one of the balls $B^{\|\frac{e-u_k}{2}\|}_{\frac{e+u_k}{2}}$ (Theorem 3.1) constructed by the sites, this implies that h will be examined by at least one site. The proof follows immediately. □

According to Lemma 5.3, universally good predictors guarantee Property 2, thus leading to a decreased synchronization frequency. Nevertheless, Lemma 5.3 simultaneously assumes that predictors are always universally good and no information sharing exists between the sites. In a large-scale distributed scenario, however, some sites may adhere to neither Property 3 nor Property 1. Obviously, any efficient monitoring algorithm has to take into consideration such situations and guarantee the "correctness" of the monitoring model together with Property 2. Note that the term "correctness" refers to the ability of the monitoring algorithm to ensure that the intersection is always covered by the union of sites' local constraints. Our evaluation of the overhead required to monitor that Lemma 5.3 is satisfied shows that this overhead is significant and that it outweighs the benefits of the mechanism. Thus, we seek to devise alternative implementations that are based on more relaxed conditions.

6. SIMPLIFIED ALTERNATIVES

6.1. Relaxing the Containment Condition

The containment of convex hulls (Section 5.1) as a sufficient prerequisite to achieve accordance with Property 2 is seemingly hard to achieve, let alone come up with ways to continuously check it in a distributive manner. To confront the preceding drawbacks we investigate an alternative approach which relaxes that condition. Instead of distributively checking the containment condition, we direct our interest to the more practical alternative of making it likely. Intuitively, we are looking for a way to monitor $v(t)$ such that the following requirements are fulfilled.

Requirement 1. The local constraints in the shape of constructed balls are tighter than those of the original framework (Section 3.1).

Requirement 2. The choice of the reference point should be as close as possible to e (due to the establishment of Lemma 5.2).

This pair of requirements renders the containment of new constraints in $\bigcup_{i=1}^{n} B^{\|\frac{e-u_i}{2}\|}_{\frac{e+u_i}{2}}$ more likely. Furthermore, we wish to invent an algorithm that avoids any communication among the sites, unless a threshold crossing is observed.

Since $v(t) = \frac{\sum_{i=1}^{n} w_i u_i^p}{\sum_{i=1}^{n} w_i}$ and $v(t) = \frac{\sum_{i=1}^{n} w_i u_i}{\sum_{i=1}^{n} w_i}$, for any $\mu \in \mathcal{R}$ we can express the true global vector as $v(t) = \frac{\sum_{i=1}^{n} w_i (\mu u_i^p + (1-\mu)u_i)}{\sum_{i=1}^{n} w_i}$. So, in order to monitor the current status of the true global vector we may reside to a new convex hull, namely $Conv\ (\mu u_1^p + (1-\mu)u_1, \ldots, \mu u_n^p + (1-\mu)u_i)$. We then find ourselves concerned with identifying a value for μ that may fulfill Requirements 1 and 2.

LEMMA 6.1. *For any $\frac{1}{2} \leq \mu \leq 1$, when Property 1 holds, tighter local constraints compared to the framework of Section 3.1 are guaranteed, that is, $\|(\mu u_i^p + (1-\mu)u_i) - (\mu e^p + (1-\mu)e)\| \leq \|u_i - e\|$.*

PROOF.

$$\|u_i^p - e^p\| \leq \|u_i - e\| \overset{\mu \geq 0}{\Leftrightarrow} \|\mu u_i^p - \mu e^p\| \leq \|\mu u_i - \mu e\| \Leftrightarrow$$

$$\|\mu u_i^p - \mu e^p + (1-\mu)(u_i - e) + (\mu - 1)(u_i - e)\| \leq \|\mu u_i - \mu e\|$$

By the triangle inequality,

$$\|\mu u_i^p - \mu e^p + (1-\mu)(u_i - e)\| - \|(\mu - 1)(u_i - e)\| \leq \|\mu u_i - \mu e\| \Leftrightarrow$$

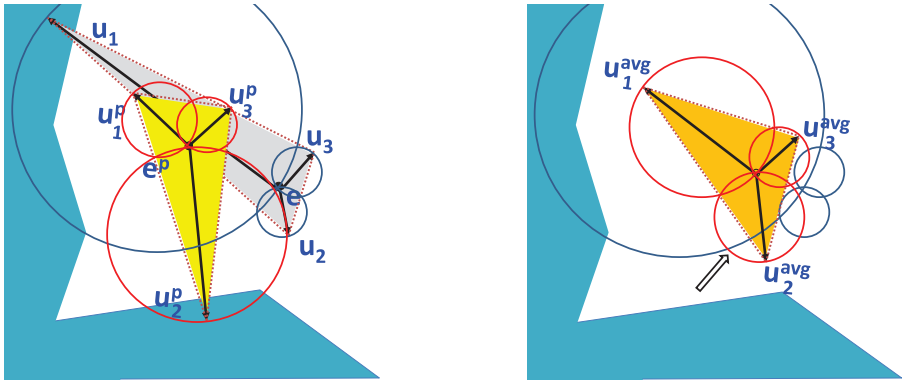
$$\|\mu u_i^p - \mu e^p + (1-\mu)(u_i - e)\| \leq (2\mu - 1)\|u_i - e\|.$$

Obviously, $(2\mu - 1) \geq 0 \Leftrightarrow \mu \geq \frac{1}{2}$. Additionally, the balls that are built by the original framework possess a radius of $\frac{\|u_i - e\|}{2}$ and as a result, for $\mu \leq 1$ we obtain

$$\|\mu u_i^p - \mu e^p + (1-\mu)(u_i - e)\| \leq (2\mu - 1)\|u_i - e\| \leq \|u_i - e\|.$$

The latter inequality completes the proof. □

The previous lemma provides a rough upper, as well as lower, bound to the value of μ such that $\|(\mu u_i^p + (1-\mu)u_i) - (\mu e^p + (1-\mu)e)\| \leq \|u_i - e\|$ in every site. This means that sites construct tighter constraints than the ones they possessed using the original framework.



(a) The original and the prediction-based convex hulls cross the threshold. The blue and red balls depict the areas monitored by each site, correspondingly, for the original and the prediction-based frameworks. S_2 violates Property 1 producing a larger prediction deviation (u_2^p) than the corresponding drift vector's (u_2) length.

(b) Convex hull and local constraints of the average model. Threshold crossing is prevented with stricter local constraints (except for S_2) and increased $\bigcup_{i=1}^{n} B^{\| \frac{e^p+e}{4} - \frac{u_i^p+u_i}{4} \|}_{\frac{e^p+e+u_i^p+u_i}{4}}$ area contained in the constraints of the original bounding algorithm.

Fig. 4. The effect of the average model adoption.

The Average Model. Lemma 6.1 shows that setting $\mu = \frac{1}{2}$ meets Requirement 1 and simultaneously provides beforehand some minimum knowledge with respect to the closest we can move $\mu e^p + (1-\mu)e$ towards e for Requirement 2 to be satisfied as well. Based on these, we are able to devise a first simpler alternative to the containment of convex hulls notion, which we term as the *average model*. The average model monitors $Conv\left(\frac{u_1^p+u_1}{2}, \ldots, \frac{u_n^p+u_n}{2}\right) \subseteq \bigcup_{i=1}^{n} B^{\| \frac{e^p+e}{4} - \frac{u_i^p+u_i}{4} \|}_{\frac{e^p+e+u_i^p+u_i}{4}}$ by a priori picking a value of $\mu = \frac{1}{2}$.

Figure 4 depicts an example of the average model adoption, where both the original and the prediction-based convex hulls cross the threshold surface in different areas (we included three sites in this example to simplify the exposition). In Figure 4(a), notice that for S_2 Property 1 is violated. Despite this fact, as shown in Figure 4(b), the average model can still ward off threshold crossing, nearly achieving containment of its constraints (balls) in those of the original bounding algorithm.

The Safer Model. We now discuss an alternative model that relaxes Requirement 2. Following a rationale similar to that of Sharfman et al. [2008], we observe that at any given timestamp, the sites can individually choose the reference point $\mu e^p + (1-\mu)e$, $\frac{1}{2} \leq \mu \leq 1$ which is farther from the threshold surface and, at the same time, ensures smaller local constraints. Note that, by being far from the threshold surface, a reference point makes the local constraints of any predictor-based monitoring model less possible to cause a crossing of the threshold surface [Sharfman et al. 2008]. This second alternative is termed as the *safer model*.

At the first step of the algorithm, every site starts with $\mu_1 = \frac{1}{2}$ and calculates $\mu_1 e^p + (1-\mu_1)e$. In addition, let e_1^* denote the vector lying on the threshold surface and being the closest to $\mu_1 e^p + (1-\mu_1)e$ [2]. Every site is capable of individually computing $\|\mu_1 e^p + (1-\mu_1)e - e_1^*\|$ and thus determining the distance the first examined reference point yields. To restrain the computational intensiveness of the technique, we define a number of allowed steps θ, such that in every subsequent step $1 \leq j \leq \theta$ the sites

[2]The details on how to compute e^* can be found in Sharfman et al. [2008].

employ a value of $\mu_j = \mu_{j-1} + \frac{1}{2^\theta}$ until $\mu_\theta = 1$. Eventually, the μ_j value that induces the largest distance is chosen. Notice that, using this framework, the sites can reach a consensus regarding μ without any additional communication. This happens due to the fact that the choice of the final μ is based on common criteria related to the threshold surface and the e, e^p vectors that are known to all sites.

6.2. Loosened Intersection Monitoring

So far in this section we have proposed simplistic alternatives that relax the convex hull containment condition that was discussed in Section 5.1. The presented (*average* and *safer*) predictor-based monitoring models do manage to avoid any direct communication between the sites unless a threshold crossing is detected. Although they do not necessarily abide by Property 2, these models encompass Requirements 1 and 2 (for the average model) and are thus in practice likely to substantiate a condition that is hard to check in a distributed manner.

We next aim at inventing a loosened version for the intersection monitoring model of Section 5.2. As previously, we wish to come up with a mechanism that avoids any communication between sites unless a threshold crossing happens and simultaneously makes Property 2 highly likely. Property 1 is again set as a simple prerequisite, but note that all our algorithms in this section remain correct even if it does not hold, since local constraints still totally cover the monitored area of the input domain. In Section 5.2 we saw that $v(t) \in Conv(u_1, \ldots, u_n) \cap Conv(u_1^p, \ldots, u_n^p)$ while in this section we demonstrate that $v(t)$ also lies in any $Conv\ (\mu u_1^p + (1-\mu)u_1, \ldots, \mu u_n^p + (1-\mu)u_i)$ which for $\frac{1}{2} \leq \mu \leq 1$ possesses the desired characteristics formulated in Requirements 1 and 2. The following lemma provides a primitive result on how the intersection monitoring can be achieved using the aforementioned logic. For ease of exposition, we use $Conv_\cap^3$ to denote the triple intersection of these three (original, predicted, and weighted) convex hulls, while $Sur(Conv_\cap^3) \equiv max_{i=1..n} B_e^{\|e-u_i\|} \cap max_{i=1..n} B_{e^p}^{\|e^p-u_i^p\|} \cap max_{i=1..n} B_{\mu e^p+(1-\mu)e}^{\|\mu e^p+(1-\mu)e-\mu u_i^p-(1-\mu)u_i\|}$, where $max_{i=1..n} B_c^{\|r\|}$ denotes the corresponding (in each maximization term) ball of maximum radius.

LEMMA 6.2. *For any $\mu \in R$, the area inscribed in $Conv_\cap^3$ is covered by the region induced by $Sur(Conv_\cap^3)$.*

PROOF. Initially notice that

$$Conv(u_1, \ldots, u_n) \subset \bigcup_{i=1}^{n} B_{\frac{e+u_i}{2}}^{\|\frac{e-u_i}{2}\|} \subset \max_{i=1..n} B_e^{\|e-u_i\|}, \tag{2}$$

$$Conv\left(u_1^p, \ldots, u_n^p\right) \subset \bigcup_{i=1}^{n} B_{\frac{e^p+u_i^p}{2}}^{\|\frac{e^p-u_i^p}{2}\|} \subset \max_{i=1..n} B_{e^p}^{\|e^p-u_i^p\|}, \tag{3}$$

$$\begin{aligned} &Conv\left(\mu u_1^p + (1-\mu)u_1, \ldots, \mu u_n^p + (1-\mu)u_n\right) \\ &\subset \bigcup_{i=1}^{n} B_{\frac{\mu e^p+(1-\mu)e+\mu u_i^p+(1-\mu)u_i}{2}}^{\|\frac{\mu e^p+(1-\mu)e-\mu u_i^p-(1-\mu)u_i}{2}\|} \subset \max_{i=1..n} B_{\mu e^p+(1-\mu)e}^{\|\mu e^p+(1-\mu)e-\mu u_i^p-(1-\mu)u_i\|}. \end{aligned} \tag{4}$$

So, each time, the maximum balls cover the corresponding convex hulls. We want to prove that the intersection of the latter balls also covers the intersection of the convex hulls. The proof will be derived by contradiction.

Suppose that a vector $h \in Conv_\cap^3$ exists. Now assume that the vector h does not lie in at least one of $max_{i=1..n} B_e^{\|e-u_i\|}$, $max_{i=1..n} B_{e^p}^{\|e^p-u_i^p\|}$, or $max_{i=1..n} B_{\mu e^p+(1-\mu)e}^{\|(\mu e^p+(1-\mu)e)-(\mu u_i^p+(1-\mu)u_i)\|}$.

However, this would violate at least one of the Propositions 2,3,4, which is a contradiction. This concludes the proof. □

Reckon, however, that the ascertainment of Lemma 6.2 cannot be tracked in a distributed manner. This happens because the site that determines each maximum ball may be different. Should Property 1 and thus (for $\frac{1}{2} \leq \mu \leq 1$) Lemma 6.1 hold, what sites actually need to perform so that they can distributively track $Conv_{\cap}^{3}$ is to use $B_{e}^{\|e-u_i\|}$, $B_{e^p}^{\|e-u_i\|}$ and $B_{\mu e^p+(1-\mu)e}^{\|e-u_i\|}$. To understand this, please observe that if both $\|e^p-u_i^p\|$, $\|(\mu e^p+(1-\mu)e)-\mu u_i^p-1-\mu)u_i\| \leq \|e-u_i\|$ (due to Property 1 and Lemma 6.1, respectively), it is evident that $Sur(Conv_{\cap}^{3}) \subset max_{i=1..n}B_{e}^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|} \cap max_{i=1..n}B_{\mu e^p+(1-\mu)e}^{\|e-u_i\|}$.

Hence, the site that possesses the maximum $\|e-u_i\|$ will check whether the intersection of its locally constructed balls crosses the threshold. Provided that the intersection of the local balls does not cross the threshold at any site (and thus at the site with the maximum $\|e-u_i\|$ as well), synchronization can safely be avoided. At this point, we would be interested in identifying proper values for μ that refine the range ($\frac{1}{2} \leq \mu \leq 1$) established in Lemma 6.1. Nonetheless, the following corollary shows that if we have to employ $\|e-u_i\|$ as the radius of the balls, $max_{i=1..n}B_{\mu e^p+(1-\mu)e}^{\|e-u_i\|}$ does not refine the intersection outcome.

COROLLARY 6.3. *For any* $0 \leq \mu \leq 1$, $max_{i=1..n}B_{\mu e^p+(1-\mu)e}^{\|e-u_i\|}$ *does not refine the region induced by* $max_{i=1..n}B_{e}^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|}$.

PROOF. Assume that there exists a vector $h \in max_{i=1..n}B_{e}^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|}$. Then

$$\|h-e\| \leq \max_{i=1..n}\|e-u_i\| \overset{1\geq\mu}{\Rightarrow} (1-\mu)\|h-e\| \leq (1-\mu)\max_{i=1..n}\|e-u_i\|$$

and $\|h-e^p\| \leq max_{i=1..n}\|e-u_i\| \overset{\mu\geq 0}{\Rightarrow} \mu\|h-e^p\| \leq \mu max_{i=1..n}\|e-u_i\|$.

Summing the aforesaid and merely applying the triangle inequality, we obtain

$$\|h-(\mu e^p+(1-\mu)e)\| \leq \mu\|h-e^p\|+(1-\mu)\|h-e\| \leq \max_{i=1..n}\|e-u_i\|.$$

The latter result exhibits that if a vector lies in the intersection of $max_{i=1..n}B_{e}^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|}$, it will definitely lie in the $max_{i=1..n}B_{\mu e^p+(1-\mu)e}^{\|e-u_i\|}$ as well. Consequently that ball does not contribute to refining the monitored area. □

Corollary 6.3 is true for any $0 \leq \mu \leq 1$, but note that in order to ensure $\|e^p-u_i^p\| \leq \|e-u_i\|$ and $\|(\mu e^p+(1-\mu)e)-\mu u_i^p-(1-\mu)u_i\| \leq \|e-u_i\|$, we had already assumed that $\frac{1}{2} \leq \mu \leq 1$. Hence it suffices to check whether the pair $B_{e}^{\|e-u_i\|} \cap B_{e^p}^{\|e-u_i\|}$ crosses the threshold in at least one site[3]. Figure 5 provides an exemplary application of the intersection monitoring procedure described so far, where $max_{i=1..n}B_{e}^{\|e-u_i\|}$, $max_{i=1..n}B_{e^p}^{\|e-u_i\|}$ are produced by S_1.

On the other hand, following an intuition similar to the one utilized in the average model, an alternative is to track $Conv\ (u_1^p,\ldots,u_n^p) \cap Conv\ (\frac{u_1^p+u_1}{2},\ldots,\frac{u_n^p+u_n}{2})$ instead. In other words, this time each site S_i needs to individually construct two balls using e^p and $\frac{e^p+e}{2}$ as centers and $M = max\{\|e^p-u_i^p\|, \|\frac{e^p+e}{2}-\frac{u_i^p+u_i}{2}\|\}$ as the common radius (please note that M refers to the maximum of the pair of local radii). Subsequently,

[3]Even if the site that determines the maximum radius finds that Property 1 does not hold, Corollary 6.3 is still valid upon replacing $\|e-u_i\|$ with $\|e^p-u_i^p\|$.

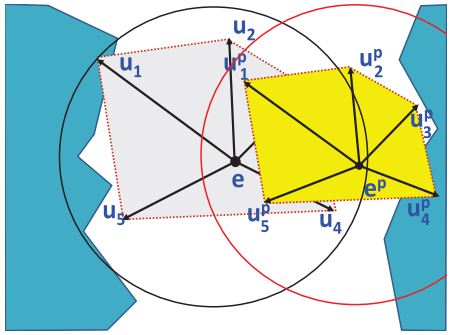


Fig. 5. Loosened intersection monitoring. $max_{i=1..n}B_e^{\|e-u_i\|}$, $max_{i=1..n}B_{e^p}^{\|e-u_i\|}$ are produced by S_1 which is the one that checks $max_{i=1..n}B_e^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|}$. No threshold crossing occurs despite that individual convex hulls violate the threshold surface.

a synchronization is caused when at least one S_i detects that the locally constructed intersection crosses the threshold.

We conclude our study by showing the condition which makes the latter intersection tracking preferable, as it results in smaller local constraints compared to $max_{i=1..n}B_e^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|}$.

PROPOSITION 6.4. *When Property 1 holds and* $max_{i=1..n}B_e^{\|e-u_i\|} \supseteq max_{i=1..n}B^M_{\frac{e^p+e}{2}}$, *then* $max_{i=1..n}B^M_{e^p} \cap max_{i=1..n}B^M_{\frac{e^p+e}{2}} \subseteq max_{i=1..n}B_e^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|}$.

PROOF. For any vector $h \in max_{i=1..n}B^M_{e^p} \cap max_{i=1..n}B^M_{\frac{e^p+e}{2}}$ we have

$$\|h - e^p\| \leq M \overset{\|e^p-u_i^p\| \leq \|e-u_i\|}{\leq} max\|e - u_i\|$$

which entails that $h \in max_{i=1..n}B_{e^p}^{\|e-u_i\|}$. If in addition $max_{i=1..n}B_e^{\|e-u_i\|} \supseteq max_{i=1..n}B^M_{\frac{e^p+e}{2}}$ then $h \in max_{i=1..n}B_e^{\|e-u_i\|}$ as well. Hence $h \in max_{i=1..n}B_e^{\|e-u_i\|} \cap max_{i=1..n}B_{e^p}^{\|e-u_i\|}$. □

7. CHOOSING AMONGST ALTERNATIVES

So far, we investigated a number of simpler alternatives that loosen the strong monitoring frameworks of Section 5, namely the convex hull containment as well as the intersection monitoring framework. We based our analysis on Property 1 as an intuitive assumption also employed in past studies [Cormode et al. 2005; Cormode and Garofalakis 2005, 2008] and evolved it to practical tracking mechanisms together with appropriate speculative analysis. Nonetheless, upon relaxing the monitoring conditions we also relaxed their conformity to Property 2, that is, the prerequisite for strong predictor-based monitoring models. Since the coordinator is supposed to a priori dictate the predictor-based tracking alternative that should be uniformly utilized by sites at least until the next synchronization, we need to provide a decision-making mechanism that enables it choose among the available options and adjust its decisions on their anticipated performance with respect to communication savings.

The available tracking options that do not belong (excluding the trivial choice of the original framework) to the strong predictor-based monitoring models' class include:

—Model 0: monitoring of $Conv(u_1, \ldots u_n)$ as in Section 3.1;
—Model 1: monitoring of $Conv(u_1^p, \ldots, u_n^p)$ as in Section 4;
—the average model;

—the safer model;
—Intersection 1: tracking of $max_{i=1..n} B_e^{\|e-u_i\|} \cap max_{i=1..n} B_{e^p}^{\|e-u_i\|}$; and
—Intersection 2: tracking of $max_{i=1..n} B_{e^p}^{M} \cap max_{i=1..n} B_{\frac{e^p+e}{2}}^{M}$.

In this section we first seek to examine in Section 7.1 whether the selection of the used monitoring model can be decided based on probabilistic models that try to estimate the probability of a threshold violation using each model. We describe the limitations and difficulties of basing our decisions on the results of our probabilistic analysis and then (in Section 7.2) seek to devise an adaptive way of chosing amongst the alternative models, based on statistics collected during the operation of the algorithm.

7.1. A Probabilistic Viewpoint

We initially seek to analyze the estimated benefit of all the presented tracking mechanisms from a probabilistic point of view. Our analysis formulates and subsequently examines a worst-case scenario for each of the proposed models. Eventually, it attempts to compare their expected performance with respect to the communication load.

Going back to the example depicted in Figure 3, we recall that a tighter convex hull by a prediction model may not necessarily result in fewer local threshold crossings (local violations). For example, in the example of Figure 3, the main reason why the constraints of the prediction-based convex hull cause a threshold crossing is the distance of e^p from the threshold surface. As a consequence, despite the fact that $Conv(u_1, \ldots, u_n)$ yields larger balls, the second factor that affects the communication savings of the mere $Conv\ (u_1^p, \ldots, u_n^p)$ monitoring is the distance of the common reference point from the threshold surface. This fact was also noted when presenting the rationale behind the safer model's adoption throughout our study.

Figure 6 depicts the maximum balls, centered at e and e^p respectively, that can be inscribed without crossing the threshold surface. Let D_e and D_{e^p} denote the radius of these maximum balls, respectively. Then

$$u_i \in B_e^{D_e} \Rightarrow B_{\frac{e+u_i}{2}}^{\|\frac{e-u_i}{2}\|} \subset B_e^{\|e-u_i\|} \subset B_e^{D_e}, \quad \forall i \in \{1..n\}$$

and similarly

$$u_i^p \in B_{e^p}^{D_{e^p}} \Rightarrow B_{\frac{e^p+u_i^p}{2}}^{\|\frac{e^p-u_i^p}{2}\|} \subset B_{e^p}^{\|e^p-u_i^p\|} \subset B_{e^p}^{D_{e^p}}, \quad \forall i \in \{1..n\}.$$

For any drift vector u_i to cause a violation, it needs to exit the corresponding ball. Therefore, utilizing Markov's inequality [Garofalakis et al. 2002], we can bound the probability of a threshold crossing in the original monitoring framework [Sharfman et al. 2007b] as follows.

$$Pr_{violation_i}^{Model0} \leq Pr\{\|e-u_i\| \geq D_e\} \leq \frac{E(\|e-u_i\|)}{D_e}$$

In the same way, for $Conv(u_1^p, \ldots, u_n^p)$ monitoring,

$$Pr_{violation_i}^{Model1} \leq Pr\left\{\|e^p-u_i^p\| \geq D_{e^p}\right\} \leq \frac{E\left(\|e^p-u_i^p\|\right)}{D_{e^p}}.$$

Notice that as long as the norm of the *expected drift* does not exceed the radius of $B_e^{D_e}$, then $E(\|e-u_i\|) \leq D_e$ (a similar discussion also holds for the prediction deviation vectors case). This ensures that the provided upper bound receives a meaningful value ≤ 1. Similar bounds can be extracted for the case of the average (for $\mu = \frac{1}{2}$) as well as the

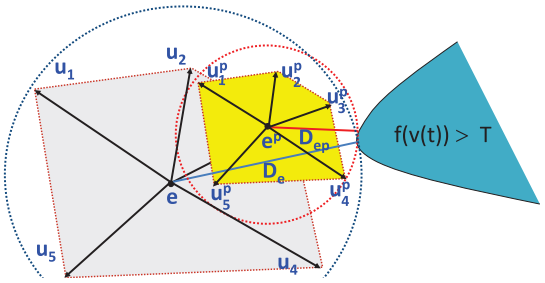


Fig. 6. Maximum spheres centered at the reference points e and e^p that can be inscribed without violating the threshold surface. The radius of these spheres is denoted as D_e and D_{e^p}, respectively. Apart from the size of the local constraints, the position of the utilized reference points determines the probability of a synchronization occurrence.

safer model.

$$\begin{aligned} Pr^{Safer}_{violation_i} &\leq Pr\{\|\mu e^p + (1-\mu)e - (1-\mu)u_i + \mu u_i^p\| \geq D_{\mu e^p + (1-\mu)e}\} \\ &\leq \frac{E(\|\mu e^p + (1-\mu)e - (1-\mu)u_i + \mu u_i^p\|)}{D_{\mu e^p + (1-\mu)e}} \end{aligned}$$

Please recall that the safer model essentially seeks to maximize the denominator of the preceding fraction. Moreover, due to Property 1, the safer model can also yield a reduced nominator compared to the original monitoring framework. Based on Lemma 6.1, for $\frac{1}{2} \leq \mu \leq 1$, we have

$$\begin{aligned} &\|(\mu u_i^p + (1-\mu)u_i) - (\mu e^p + (1-\mu)e)\| \leq \|u_i - e\| \\ &\Rightarrow E(\|\mu e^p + (1-\mu)e - (1-\mu)u_i + \mu u_i^p\|) \leq E(\|e - u_i\|). \end{aligned}$$

Of course, these ascertainments should come as no surprise given our discussion in Section 6.

For a synchronization to take place, at least one out of the n sites needs to cause a violation. Assuming that the data streams arriving at the various sites do not exhibit any dependency, the probability of a synchronization, taking Model 1 for instance, can be bounded by

$$Pr^{Model1}_{Sync} = 1 - \Pi_{i=1}^{n}(1 - Pr^{Model1}_{violation_i}) \leq 1 - \frac{\Pi_{i=1}^{n}(D_{e^p} - E(\|e^p - u_i^p\|)}{(D_{e^p})^n}.$$

Similar results can be extracted for both the original and the safer model.

Regarding the intersection monitoring frameworks, we point out that the probability of violating the threshold surface can be upper bounded by the probability that local spheres ($max_{i=1..n}B_e^{\|e-u_i\|}$ and $max_{i=1..n}B_{e^p}^{\|e-u_i\|}$) simultaneously cause a violation. The upper bound is valid since, while these local spheres may both cause a threshold crossing at the same time, the monitored intersection may still avoid a violation. However, the contrary is not possible, that is, if the monitored intersection results in a threshold crossing, then both local spheres result in a threshold crossing as well. Therefore

$$\begin{aligned} Pr^{Intersection1}_{violation} &\leq Pr\left\{\max_{i=1..n}\|e - u_i\| \geq D_e \cap \max_{i=1..n}\|e - u_i\| \geq D_{e^p}\right\} \\ &= Pr\left\{\max_{i=1..n}\|e - u_i\| \geq max\{D_e, D_{e^p}\}\right\} \leq \frac{E\left(\max_{i=1..n}\|e - u_i\|\right)}{max\{D_e, D_{e^p}\}}. \end{aligned}$$

Table III. Probability of Synchronizations of the Proposed Alternative Tracking Schemes

Alternative	Probability of Synchronizations
Model 0	$O\left(1-\frac{\Pi_{i=1}^{n}(D_e-E(\|e-u_i\|)}{(D_e)^n}\right)$
Model 1	$O\left(1-\frac{\Pi_{i=1}^{n}(D_{e^p}-E(\|e^p-u_i^p\|)}{(D_{e^p})^n}\right)$
Safer & Average Models	$O\left(1-\frac{\Pi_{i=1}^{n}(D_{\mu e^p+(1-\mu)e}-E(\|(\mu u_i^p+(1-\mu)u_i)-(\mu e^p+(1-\mu)e)\|)}{(D_{\mu e^p+(1-\mu)e})^n}\right)$
Intersection 1	$O\left(\frac{E(max_{i=1..n}\|e-u_i\|)}{max\{D_e,D_{e^p}\}}\right)$
Intersection 2	$O\left(\frac{E(M)}{max\{D_{e^p},D_{\frac{e^p+e}{2}}\}}\right)$

The corresponding bound for $Intersection2$ can be derived as $\frac{E(M)}{max\{D_{e^p},D_{\frac{e^p+e}{2}}\}}$. Our intersection monitoring schemes determine whether to call for a synchronization relying only on the maximum inscribed sphere. Therefore, the bound on the violation probability provided before also constitutes a bound on the probability of a synchronization. Given the preceding, Table III summarizes the worst-case bounds for the probability of synchronization at any given time instance (the temporal reference symbol t in the table is omitted to simplify the exposition).

Difficulties of Using our Probabilistic Viewpoint. In contrast with the nominator, the denominator of all the extracted bounds can be computed during the synchronization, as is the case with the constant D_e. D_{e^p} is altered due to the movement of e^p but can be determined individually by sites as they share the common information about the posed threshold and e^p at any given timestamp. Obviously, the nominator changes as time passes and sites need to be continuously aware of the expected drifts of the candidate models so as to consult the corresponding bounds and dynamically choose the best monitoring model.

As a matter of fact, it is difficult to extract any generic decision-making mechanism regarding the best choice of the tracking mechanism, unless we are aware of the exact probability density function of $Pr_{Sync}^{Chosen\_Model}(t)$ at any given t. To simplify things, one may assume that not only $Pr_{Sync}^{Chosen\_Model}$, but also the quantities that bound this probability remain relatively stable and compute all probabilities given past statistics that are transmitted to nodes during some synchronization.

Despite the fact that our previous analysis may be interesting, it incorporates strong assumptions which tremendously limit its applicability to practical tracking scenarios (such as lack of dependency between local data streams and $Pr_{Sync}^{Chosen\_Model}$'s stability). Moreover, any global statistics that need to be continuously transmitted for computing the probabilities of Table III increase the overall bandwidth consumption, which is clearly not desirable. Because of the aforementioned limitations, in what follows we devise a more empirical approach that selects the monitoring model to be adopted based on the recent performance of these models, while also not requiring any statistics to be transmitted by the coordinator to the nodes.

7.2. An Empirical Viewpoint

In order to provide an appropriate decision-making mechanism, we require that sites keep up monitoring all the six options mentioned earlier. This monitoring will take place only for models that would not result in any local transmission since the last synchronization (i.e., we stop monitoring an alternative model for which we detect that

a transmission would have been caused). Notice that one model has been chosen as the main model after the last synchronization. Thus, for each of the six alternatives, the sites maintain 6 bits, where the i-th bit is set iff the corresponding monitoring mode would have resulted in at least one transmission since the last synchronization. A synchronization can still be caused only by the main model. Upon a synchronization, however, together with $v_i(t)$ and the velocity vector in the case of the velocity acceleration model choice, sites attach 5 bits (they do not need to send a bit for the current model being used) on their messages. The 5 transmitted bits per node obviously constitute a very small overhead. Please note that the following facts hold in our adaptive algorithm.

—No site has a violation using the current model in a previous time instance (since the previous synchronization).
—An alternative model that has its corresponding bit to 1 in any of the sites would not have been better than the model currently being used, since it would have resulted in a transmission in a prior (or the current) time instance.
—Based on the preceding observation, we decide to switch to an alternative model only if the corresponding bits for this model were equal to 0 in all the sites.

In case of multiple alternatives with unset bits, a random choice among such alternatives is performed. We henceforth term this empirical viewpoint for Choosing Amongst Alternatives as the CAA approach.

8. APPROXIMATE QUERY ANSWERING

8.1. Absolute Threshold Monitoring

In the previous sections we focused on tracking functions where the posed threshold T, and thus the threshold surface of the input domain, remained stable throughout the monitoring process. Nevertheless, in many cases [Cormode and Garofalakis 2005, 2007, 2008; Cormode et al. 2005, 2007; Olston et al. 2003] it is particularly interesting to continuously maintain at the coordinator an approximation (i.e., given some accuracy constant $\varepsilon > 0$) of the true value of the monitored function. In terms of the concepts previously presented in our prediction-based monitoring framework, such a need can be expressed in the following tracking task.

$$|f(v(t)) - f(e^p(t))| \leq \varepsilon \tag{5}$$

In this scenario, the coordinator is capable of constantly providing at any given timestamp t ε-approximate answers ($f(e^p(t))$) regarding the value of the monitored function, utilizing the predicted estimate vector. Central data collection is caused only when the error in the answer may exceed the specified accuracy bound ε.

It is not difficult to see that Inequality (5) can be decomposed into two tracking tasks, namely $f(v(t)) \leq \varepsilon + f(e^p(t))$ and $f(v(t)) \geq f(e^p(t)) - \varepsilon$, which can be handled separately by the empirical CAA approach devised in the previous section. This shows that our techniques are general enough to accommodate any approximate function monitoring requirement. Nevertheless, an undesirable effect of this approach is that it doubles the computational load on individual sites, since this time each site is required to assess whether two d-dimensional balls of local constraints (one for each of the tracking tasks given before) are crossing the threshold surface[4]. This is a time-consuming task especially when d is high and more than one function is monitored.

[4]Note that to assess crossings, sites essentially need to find the two couples of vectors included in each constructed ball that yield max/min $f()$ values (where $f()$ is the function of interest) and check if $maxf()$ lies in a different side of the threshold compared to $minf()$. Finding these d-dimensional vectors involves solving a couple of constrained optimization problems.

Thus, the computational load may in turn lead to lags in the delivery of synchronization decisions, which finally affects the real-time delivery of answers to the streaming applications. Additionally, in resource-constrained environments such as those examined in Burdakis and Deligiannakis [2012] and Sharfman et al. [2007a], this computational load would deplete their remaining energy supply even for mediocre-dimensioned vectors.

Therefore, in what follows we propose a problem transformation tailored for the approximate query answering task (Inequality (5) as well as Inequality (8) discussed later on). The advantage of the proposed transformation is that it reduces the tracking process to a single monitoring task where threshold crossing tests use mere L_2-norm comparisons. Furthermore, the transformed geometric setting ensures that good predictors that satisfy Property 1 can always yield reduced communication burden (Sections 8.1.1 and 8.1.2). As a result, it favors Model 1's adoption and forms a proper scenario for assessing the usefulness of our CAA approach versus sole Model 1's good performance (Section 9.5).

To achieve the desired transformation, we reside to the notion of Holder continuity. More precisely, a function $f : R^d \longrightarrow R$ is called Holder continuous on R^d if, for Holder constants $H_f, \alpha > 0$, the change in the output of f can be bounded by a corresponding change in the input. In our notation,

$$|f(e^p(t)) - f(v(t))| \leq H_f \cdot \|e^p(t) - v(t)\|^\alpha. \tag{6}$$

For $\alpha = 1$ the function is called Lipschitz continuous. Examples of functions that can be easily verified that conform with Holder continuity include, but are not limited to, general L_p-norms that abide by the reverse triangle inequality as well as trigonometric functions such as the cosine similarity. All these functions have been adopted within the geometric monitoring framework for detecting outliers in sensor network settings [Burdakis and Deligiannakis 2012].

Given Inequalities (5) and (6), we can actually come up with a value $\delta > 0$ such that

$$\|e^p(t) - v(t)\| \leq \delta \Rightarrow H_f \cdot \|e^p(t) - v(t)\|^\alpha \leq H_f \cdot \delta^\alpha \leq \varepsilon \tag{7}$$

which forms the final version of our monitoring task, since $|f(e^p(t)) - f(v(t))| \leq H_f \cdot \|e^p(t) - v(t)\|^\alpha \leq H_f \cdot \delta^\alpha \leq \varepsilon$. Taking L_p-norm monitoring [Burdakis and Deligiannakis 2012] as an example, the corresponding constants are set to $H_f = 1$, $a = 1$ and $\delta = \varepsilon$. Despite the fact that the aforesaid Holder condition is not always achievable for any tracked f on R^d, after a synchronization, we can define a neighborhood $U \subset R^d$ of the input domain for which Holder's condition holds and have sites call for a synchronization whenever the resulting vectors may exit this neighborhood (in addition to tracking the ε-approximation requirement). Hence, we expect that a wide variety of functions can be covered by the analysis that we present in this section.

Inequality (7) actually transforms the original monitoring target of Inequality (5) to a δ-approximation on the values of the input domain $\|e^p(t) - v(t)\| \leq \delta$. In the remainder of this section we discuss how this special tracking task can be performed by two alternative approaches.

—We can decompose the problem into a set of simple local constraints (DLC algorithm, described in Section 8.1.1). Using this technique, each site will be able to monitor a simple local constraint that is based only on the deviation of its actual local measurements vector $v_i(t)$ from its corresponding predicted vector $v_i^p(t)$.
—We can adopt the prediction-based monitoring framework (Section 8.1.2). While this technique shares some similarities with the DLC algorithm, it also allows the application of the average model and the intersection monitoring techniques, thus enabling the use of our CAA algorithm.

8.1.1. Distributed Monitoring by Simple Decomposition to Local Constraints (DLC). In order to achieve the δ-approximation target, we first decompose the problem of distributively ensuring that $\|e^p(t) - v(t)\| \leq \delta$ to simple local constraints that sites can monitor, in order to decide whether a synchronization process needs to take place. The upcoming lemma elaborates on the latter issue, introducing an appropriate sufficient condition.

LEMMA 8.1. *When the local constraint*

$$\|v_i^p(t) - v_i(t)\| \leq \delta$$

holds for any site $S_i, i \in \{1..n\}$ *at time t, then*

$$\|e^p(t) - v(t)\| \leq \delta,$$

that is, the estimation of the true global vector that is kept at the coordinating source is always a δ*-approximation of the true global vector.*

PROOF. Starting by the local constraint of a single site S_i we obtain

$$\|v_i^p(t) - v_i(t)\| \leq \delta \overset{w_i \geq 0}{\Leftrightarrow} \|w_i \cdot v_i^p(t) - w_i \cdot v_i(t)\| \leq w_i \cdot \delta.$$

Summing for n sites, we have

$$\sum_{i=1}^{n} \|w_i \cdot v_i^p(t) - w_i \cdot v_i(t)\| \leq \sum_{i=1}^{n} w_i \cdot \delta \overset{\text{triangle inequality}}{\Rightarrow}$$

$$\|\sum_{i=1}^{n} w_i \cdot v_i^p(t) - \sum_{i=1}^{n} w_i \cdot v_i(t)\| \leq \sum_{i=1}^{n} w_i \cdot \delta \overset{\sum_{i=1}^{n} w_i > 0}{\Leftrightarrow}$$

$$\|\frac{\sum_{i=1}^{n} w_i \cdot v_i^p(t)}{\sum_{i=1}^{n} w_i} - \frac{\sum_{i=1}^{n} w_i \cdot v_i(t)}{\sum_{i=1}^{n} w_i}\| \leq \delta \Leftrightarrow$$

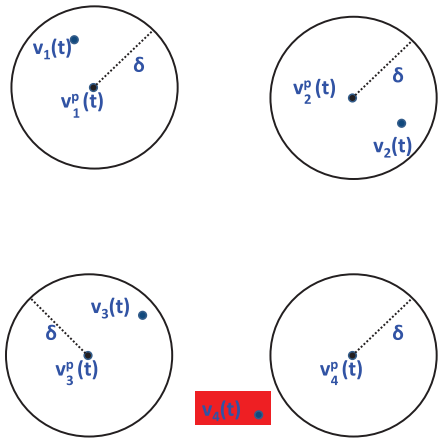
$$\|e^p(t) - v(t)\| \leq \delta$$

which concludes the proof. □

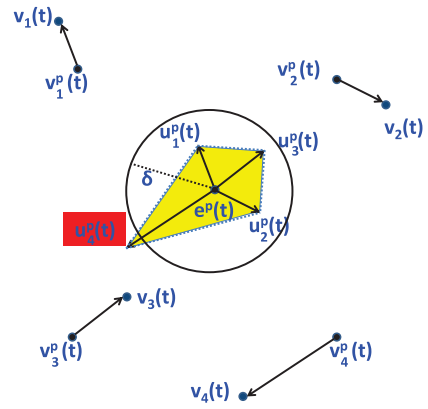
As long as Lemma 8.1 holds, no transmission is required. Each site keeps up acquiring updates and locally maintains the required information for the computation of predictors at the next synchronization, as described in Section 3.2. If $\|v_i^p(t) - v_i(t)\| \leq \delta$ is not satisfied in at least one site, then a synchronization occurs. Thus, the decomposition to local constraints essentially results in sites constructing balls centered at $v_i^p(t)$ with a radius of δ and monitoring whether $v_i(t)$ lies within this ball (see Figure 7(a)). Moreover, notice that the previously stated tracking scheme does not require any information regarding e^p to be broadcasted to sites, as e^p is not involved in the local constraint that sites need to check.

8.1.2. Applying the Prediction-Based Geometric Approach. Our second option is that of fostering the prediction-based geometric monitoring framework. Before doing so, we need to identify the basic elements of the tracking process. We start by studying the function of interest $q(v(t)) = \|e^p(t) - v(t)\| \leq \delta$ according to Inequality (7). Note that $e^p(t)$ is made known to all the sites after each synchronization process and it can thus be treated as a constant in the monitored function $q(v(t))$.

Again, let us assume that the coordinator has collected all the local vectors of monitored sites at a previous timestep. Then, the central source extracts e and e^p, which



(a) Distributed δ-approximate monitoring by simple decomposition to local constraints for four sites. The local vector $v_4(t)$ of site S_4 exits the ball of radius δ, centered at $v_4^p(t)$.



(b) Application of Model 1 for δ-approximate tracking. The prediction deviation vector of site S_4 exits the ball centered at e^p with a radius of δ size.

Fig. 7. Comparison of approximate query answering approaches.

are broadcasted to the network. Having received the estimate as well as the predicted estimate vectors, every site S_i individually computes the drift vector $u_i(t)$ and the prediction deviation vector $u_i^p(t)$.

Nonetheless, it is worth focusing on the shape of the threshold surface in more detail. The region where the monitored convex hull $Conv\ (u_1^p(t), \ldots, u_n^p(t))$ is allowed to lie is again the ball $B_{e^p}^{\delta}$ (depicted in Figure 7(b)), centered at $e^p(t)$ with radius δ, which is convex. Due to the convexity of $B_{e^p}^{\delta}$, the convex hull of interest cannot cause threshold crossing without having at least one of its vertices $(u_1^p(t), \ldots, u_n^p(t))$ doing so. Hence, for Model 1 and Model 0 it suffices for every site to check if $u_i^p(t) = e^p + (v_i(t) - v_i^p(t))$ is included in $B_{e^p}^{\delta}$. Overall, we again check a spherical constraint of radius δ, which is violated only when $\|v_i(t) - v_i^p(t)\| > \delta$. Our previous discussion renders the presented approaches of Model 1, Model 0, and DLC equivalent in terms of the size of the consulted local constraints. Note that, as explained in Section 4, the size of the constructed constraints is an important factor in determining the communication cost of a given monitoring scheme.

However, we have also noted that the choice of the reference point is the other major criterion in tuning the (minimum) distance of the tracked convex hull (or intersection) from the threshold surface. Notice that the adoption of an alternative tracking mechanism (such as the average or the safer models) does not have an impact on the threshold surface. In fact, based on the estimation $e^p(t)$ on which the coordinator bases its ε-approximate query answers, the surface of $B_{e^p}^{\delta}$ forms the threshold surface independently of the used alternative tracking mechanism.

Given the preceding, the adoption of the average model of Section 6.1 enforces sites to check whether $\frac{u_i^p + u_i}{2}$ exits $B_{e^p}^{\delta}$. As regards the safer model, notice that since $B_{e^p}^{\delta}$ forms the threshold surface, it will always be reduced to the choice of Model 1, since e^p will always be the reference point that is farther from that surface. Eventually, the loosened intersection monitoring schemes of Section 6.2 can be applied by having sites check if the local intersection of the corresponding balls exceeds the surface of $B_{e^p}^{\delta}$. As a result,

the intersection monitoring schemes together with the average model are added tools which the DLC approach cannot exploit.

Hence, the empirical CAA approach of Section 7.2 is still applicable and capable of providing additional flexibility compared to the framework of Section 8.1.1 to the distributed tracking process. This is because CAA can tune both the size of the local constraints and their position, as well as choosing the preferable monitoring technique depending on the recent performance of the corresponding alternative tracking mechanisms.

8.2. Relative Threshold Monitoring

Another form of approximate function tracking which is often utilized in the context of data streams [Cormode and Garofalakis 2008; Cormode et al. 2005; Liu et al. 2012] is that of relative threshold monitoring. In particular, this case reflects to the following monitoring task (for $f(v(t)) \geq 0$ – otherwise, the displayed inequality is reversed).

$$(1-\varepsilon) f(v(t)) \leq f(e^p(t)) \leq (1+\varepsilon) f(v(t)) \tag{8}$$

In other words, we want to ensure that the approximate answer provided by the coordinator using $f(e^p(t))$ is within $\pm\varepsilon f(v(t))$ units from the current value of the tracked f. Nevertheless, since sites are not aware of $f(v(t))$, we need to modify our prior monitoring approach to render it capable of accomplishing relative threshold monitoring.

$$\begin{cases} \dfrac{f(e^p(t))}{1+\varepsilon} \leq f(v(t)) \\ \dfrac{f(e^p(t))}{1-\varepsilon} \geq f(v(t)) \end{cases} \Leftrightarrow \begin{cases} f(e^p(t)) - \dfrac{f(e^p(t))}{1+\varepsilon} \geq f(e^p(t)) - f(v(t)) \\ \dfrac{f(e^p(t))}{1-\varepsilon} - f(e^p(t)) \geq f(v(t)) - f(e^p(t)) \end{cases}$$

The previous two conditions can both be ensured by checking whether

$$|f(e^p(t)) - f(v(t))| \leq \min\left\{\left| f(e^p(t)) - \frac{f(e^p(t))}{1+\varepsilon}\right|, \left|\frac{f(e^p(t))}{1-\varepsilon} - f(e^p(t))\right|\right\} \leq \min\left\{\left|\frac{\varepsilon f(e^p(t))}{1+\varepsilon}\right|, \left|\frac{\varepsilon f(e^p(t))}{1-\varepsilon}\right|\right\}.$$

An equivalent result can be obtained for the case when $f(v(t)) \leq 0$. The latter inequality resembles Inequality (5), with the difference being that the right side is now dynamically adjusted based on the (known to the sites) current value of $f(e^p(t))$. Thus, the tracking task can be conducted utilizing the concepts presented in Sections 8.1.1 and 8.1.2.

8.3. Discussion

A question that naturally arises is whether the absolute or relative threshold monitoring tasks can also be performed in cases of functions that either are not Holder or Lipschitz continuous or their conformity with Holder continuity is hard to be verified. As previously noted, one option is to identify a neighborhood $U \subset R^d$ where local continuity for the tracked $f()$ exists. In addition to that option, in the following examples we study functions where our framework may be rendered applicable by exploiting properties of the adopted predictor (Section 8.3.1) or by properly transforming the monitored $f()$ (Section 8.3.2). We choose to elaborate on these particular functions due to their popularity in distributed data-stream monitoring settings [Cormode and Garofalakis 2008, 2007, 2005] and within the geometric monitoring scheme [Burdakis and Deligiannakis 2012].

8.3.1. Exploiting Properties of Our Predictor. We first describe a case in which the relative threshold monitoring task can be performed even if the function $f()$ under study is not globally Holder or Lipschitz continuous. To tackle this problem, we demonstrate that it may be possible to exploit the properties of the adopted predictors in order to

achieve bounding the change in the output by a corresponding change in the input, as in Inequality (6).

Example 8.2. Assume that, as in Cormode and Garofalakis [2008], we would like to perform relative threshold monitoring on the self-join $\|v(t)\|^2$ function, which is not a Lipschitz-continuous function over all reals. Our local vectors are frequency distribution vectors [Cormode and Garofalakis 2008], that is, streaming tuples apart from either $+1$ or -1 updates on d coordinates, and we choose to adopt the LG predictor (Table II). Consider a distributed system where data arrive at sites at predefined time intervals called epochs [Burdakis and Deligiannakis 2012]. If the distributed system currently is at the $N-th$ epoch of its operation and N_s denotes the epoch when the last synchronization process took place, we have (we keep $e^p(t)$ in $\|v(t)-e^p(t)\|$ to show the correspondence with Inequality (5))

$$\begin{aligned}
&|f(v(t))-f(e^p(t))| = \|f(v(t))-f(e^p(t))\| = \|\|v(t)\|^2-\|e^p(t)\|^2\| \\
&= \|v(t)+e^p(t)\|\|v(t)-e^p(t)\| = \left\|v(t)+\frac{N}{N_s}e\right\|\|v(t)-e^p(t)\| \\
&\overset{\substack{triangle\\ inequality}}{\leq} \left(\|v(t)\|+\left\|\frac{N}{N_s}e\right\|\right)\|v(t)-e^p(t)\| \leq \left(nN\sqrt{d}+\left\|\frac{N}{N_s}e\right\|\right)\|v(t)-e^p(t)\| \\
&\leq \left(nN\sqrt{d}+\left\|\frac{N}{N_s}e\right\|\right)\delta.
\end{aligned}$$

Thus, in order to enforce the relative threshold monitoring task, the sites need to continuously check whether

$$\left(nN\sqrt{d}+\left\|\frac{N}{N_s}e\right\|\right)\delta \leq min\left\{\frac{\varepsilon f(e^p(t))}{1+\varepsilon}, \frac{\varepsilon f(e^p(t))}{1-\varepsilon}\right\}.$$

This yields the desired value of δ as $\delta = \frac{min\{\frac{\varepsilon f(e^p(t))}{1+\varepsilon}, \frac{\varepsilon f(e^p(t))}{1-\varepsilon}\}}{(nN\sqrt{d}+\|\frac{N}{N_s}e\|)}$, which can be individually computed by sites at any given epoch, as long as they share the common information about e, e^p and ε, the dimensionality d of $v_i(t)$, the size of the network n, and the values of N_s, N. □

8.3.2. Exploiting Function Transformations. We now provide an example of absolute threshold monitoring tasks that can be performed even if it is not straightforward whether the monitoring functions are Lipschitz or Holder continuous. The basic idea is to properly transform the monitoring function to other functions that are well known to abide by that type of continuity.

Example 8.3. Suppose we have installed a number of sensors in a server room so as to continuously measure the environmental conditions (such as temperature, radiance, noise levels, and so on) under which the computing machines operate. In order to assure unhindered operation, we let motes collect a number of w recent observations based on which we intend to monitor the resemblance that the global vector $v(t)$ exhibits with a given vector g representing the "normal" values of the quantities sampled by the sensor nodes. The resemblance between the $g, v(t)$ pair of vectors can be determined based on the extended Jacccard (or Tanimoto) coefficient (Tan) [Burdakis and Deligiannakis 2012; Giatrakos et al. 2013]:

In order to perform the desired transformation of the Tanimoto coefficient

$$Tan(g, v(t)) = \frac{g \cdot v(t)}{\|g\|^2 + \|v(t)\|^2 - g \cdot v(t)}$$

the initial requirement of

$$|Tan(g, e^p(t)) - Tan(g, v(t))| \leq \varepsilon$$

is first decomposed in two constraints

$$\begin{cases} T_1(t) = Tan(g, e^p(t)) - \varepsilon \ \leq \ Tan(g, v(t)) \\ T_2(t) = Tan(g, e^p(t)) + \varepsilon \ \geq \ Tan(g, v(t)) \end{cases}$$

and then transformed as follows [Giatrakos et al. 2013] (for $T_1(t),\ T_2(t) \in [0, 1]$).

$$\begin{cases} \left\| \frac{T_1(t)+1}{2T_1(t)} g - v(t) \right\| \ \leq \ \frac{\sqrt{-4T_1(t)^2 + (T_1(t)+1)^2}}{2T_1(t)} \|g\| \\ \left\| \frac{T_2(t)+1}{2T_2(t)} g - v(t) \right\| \ \geq \ \frac{\sqrt{-4T_2(t)^2 + (T_2(t)+1)^2}}{2T_2(t)} \|g\| \end{cases}$$

Notice that since $g, e^p(t))$, ε have been broadcasted, $T_1(t)$ and $T_2(t)$ can be individually computed by sites. Now then, subtracting the previous inequalities we get

$$\begin{aligned} &\left\| \frac{T_1(t)+1}{2T_1(t)} g - v(t) \right\| - \left\| \frac{T_2(t)+1}{2T_2(t)} g - v(t) \right\| \\ &\leq \frac{\sqrt{-4T_1(t)^2 + (T_1(t)+1)^2}}{2T_1(t)} \|g\| - \frac{\sqrt{-4T_2(t)^2 + (T_2(t)+1)^2}}{2T_2(t)} \|g\| \end{aligned}$$

which can be ensured if

$$\begin{aligned} &\left\| \frac{T_1(t)+1}{2T_1(t)} g - v(t) \right\| - \left\| \frac{T_2(t)+1}{2T_2(t)} g - v(t) \right\| \Bigg| \\ &\leq \left| \frac{\sqrt{-4T_1(t)^2 + (T_1(t)+1)^2}}{2T_1(t)} - \frac{\sqrt{-4T_2(t)^2 + (T_2(t)+1)^2}}{2T_2(t)} \right| \|g\|. \end{aligned}$$

The left side of the preceding can be bounded as

$$\begin{aligned} \left| \left\| \frac{T_1(t)+1}{2T_1(t)} g - v(t) \right\| - \left\| \frac{T_2(t)+1}{2T_2(t)} g - v(t) \right\| \right| &\overset{\substack{reverse\\ triangle\\ inequality}}{\leq} \\ \left\| \left(\frac{T_1(t)+1}{2T_1(t)} + \frac{T_2(t)+1}{2T_2(t)} \right) g - 2v(t) \right\| &= \\ \left\| \left(\frac{T_1(t)+1}{2T_1(t)} + \frac{T_2(t)+1}{2T_2(t)} \right) g - 2e^p(t) - 2(v(t) - e^p(t)) \right\| &\overset{\substack{triangle\\ inequality}}{\leq} \\ \left\| \left(\frac{T_1(t)+1}{2T_1(t)} + \frac{T_2(t)+1}{2T_2(t)} \right) g - 2e^p(t) \right\| + 2\|v(t) - e^p(t)\| &\leq \\ \left\| \left(\frac{T_1(t)+1}{2T_1(t)} + \frac{T_2(t)+1}{2T_2(t)} \right) g - 2e^p(t) \right\| + 2\delta &\leq \\ \left| \frac{\sqrt{-4T_1(t)^2 + (T_1(t)+1)^2}}{2T_1(t)} - \frac{\sqrt{-4T_2(t)^2 + (T_2(t)+1)^2}}{2T_2(t)} \right| \|g\|. & \end{aligned}$$

Obviously, $\delta = |\frac{\sqrt{-4T_1(t)^2+(T_1(t)+1)^2}}{2T_1(t)} - \frac{\sqrt{-4T_2(t)^2+(T_2(t)+1)^2}}{2T_2(t)}|\|g\| - \|(\frac{T_1(t)+1}{2T_1(t)} + \frac{T_2(t)+1}{2T_2(t)})g - 2e^p(t)\|$ and $H_f = 2, \alpha = 1$. □

9. EVALUATION RESULTS

In order to evaluate our algorithms, we developed a simulation environment in Java. We utilized two real datasets to derive data-stream tuples arriving at every site in the network. *Corpus* consists of 804,414 records present in the Reuters Corpus (RCV1-v2) [Lewis et al. 2004] collection. Each record corresponds to a news story to which a list of terms (features) and appropriate categorizations have been attributed. As in Sharfman et al. [2007b, 2008] we focus on the following features: *Bosnia, Ipo, Febru* while monitoring their coexistence with the *CCAT* (the CORPORATE / INDUSTRIAL) category. Aiming at identifying the relevance of these features to the CCAT category at any given time, we monitored two different functions involving the chi-square (χ^2) and mutual information (*MI*) score. We utilized the Corpus dataset in order to test our techniques using the cash register streaming paradigm, that is, taking into consideration the whole history of the tuples arriving at the various sites. In any given timestamp, after the receipt of a new tuple each site forms a vector which consists of four dimensions for the χ^2 and three dimensions for the *MI* case. These vectors have one of their positions set, while the rest remain zero. In particular, for both the functions the first position of the vector is set if the term and the category co-occur, the second if the term occurs without the CCAT category, the third in case CCAT is present without the term, while the fourth (only for χ^2 score) if neither of them appears in the incoming tuple.

Due to the nature of the incoming (binary) vectors and the utilized cash register paradigm, the previously described environment may be considered moderate to change and be thought of as easily predictable by our techniques. In order to test our algorithms in more dynamic conditions we utilized one more dataset. The *Weather* dataset includes Solar Irradiance, Wind Speed, Wind Peak, and Temperature measurements from the station in the University of Washington and for the year 2002 [Deligiannakis et al. 2004, 2007], where each file incorporates 523,439 records of measurements. We used the Weather datasets so as to monitor the Variance (Var) and the Signal-to-Noise ratio (StN) functions. We also tested our query answering techniques (Section 8) while performing approximate L_2 and cosine (Cos) similarity monitoring. We utilized Var, L_2, and Cos which have already been used within the geometric monitoring framework [Sharfman et al. 2007a; Burdakis and Deligiannakis 2012]. In addition, the StN function equals the ratio between the mean and the standard deviation ($\frac{\mu}{\sigma}$) in a given window of measurements and can thus be applicable to globally quantify the noise present in the measurements.

Eventually, we produced a synthetic dataset, named *Synthetic*. This dataset includes 523,439 records of measurements generated by a VA predictor of constant velocity $v = 5$ degrees/time unit and zero acceleration. This means that, during the monitoring task, local predictors can perfectly estimate the current values of vectors and corresponding prediction drifts are zeroed. Then, in the original synthetic dataset, we impose different degrees of noise between 10% and 50% distorting each measurement by $\pm noise\%$. We utilize the Synthetic dataset with different degrees of noise to assess the impact of the accuracy of the predictor to the performance of Model 1 which is mainly affected by the predictor's accuracy. We compare Model 1's communication cost against the rest of the proposed alternatives while tracking the Coefficient of Variation (CoV) function. CoV is chosen due to the fact that it constitutes a normalized measure of dispersion caused by the injected noise percentage.

In each experiment we first measure the number of messages transmitted in the network across different thresholds for a network configuration consisting of 10 sites.

We then use the middle-case threshold and plot the number of transmitted messages when increasing the scale of the distributed environment (the number of sites). We denote the performance of the original bounding algorithm (Section 3.1) by "Model 0", while "Model 1" refers to the mere application and monitoring of the prediction-based bounding algorithm (Section 4). "CAA" shows the performance of the Choosing Amongst Alternatives framework that was introduced in Section 7.2. We also provide indicative results of individual monitoring alternatives performance (apart from incorporating them in CAA). The alternatives are termed as "Average", "Safer", and "Intersection1" denoting the monitoring of the intersection between the original and the predicted convex hull, while "Intersection2" refers to monitoring the intersection between the average convex hull and the predicted one. As we demonstrate, none of these alternative models consistently outperforms the others. On the other hand, our CAA algorithm intelligently adapts to the characteristics of the data and picks the correct model, often switching the model of choice even within the same dataset.

For each of the lines in the graphs, we enclosed the chosen predictor using *LG* to denote the Linear Growth predictor and $VA - W$ so as to declare a Velocity/Acceleration predictor with a window of W measurements[5].

9.1. Corpus Dataset—Cash Register Paradigm

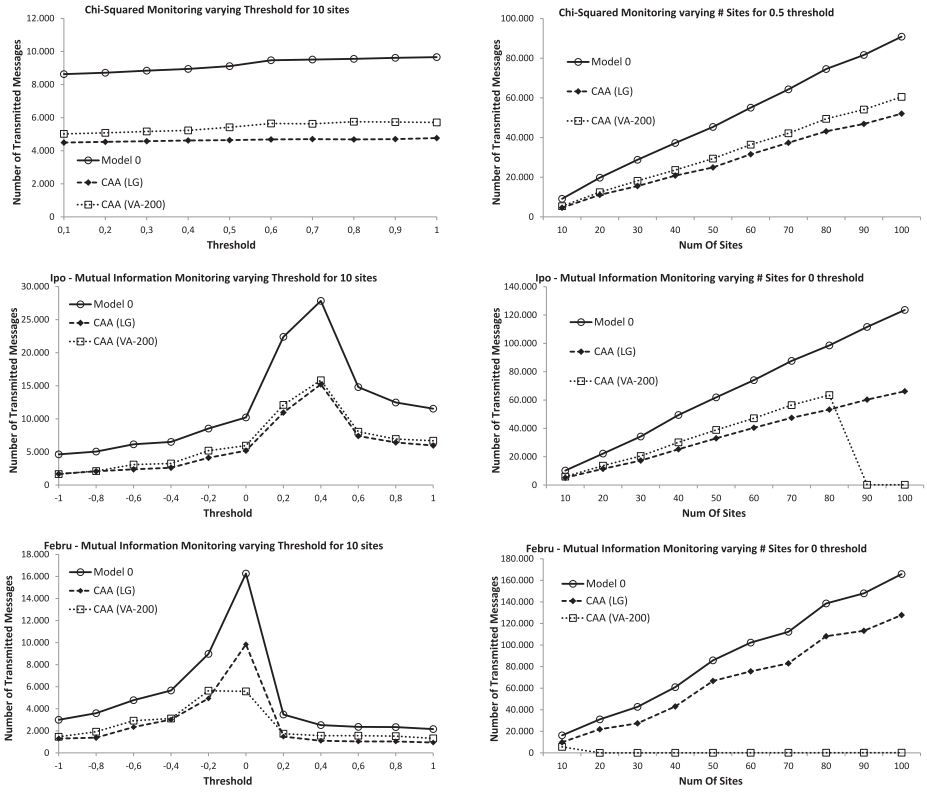
We begin our study by examining the performance of our techniques in the Corpus dataset on par with the cash register paradigm adoption. Figure 8 depicts the performance of Model 0 and of the CAA approach when using the *LG* and the *VA* predictors. Since almost 6000 documents are received within a period of a month [Sharfman et al. 2008], we choose a $W = 200$ window for the VA predictor which is expected to be roughly the amount of news stories received daily. Each column of the figure corresponds to the case of the terms "Bosnia", "Ipo", and "Febru", respectively.

Sensitivity to Threshold—Chi Square. As shown at the top of the first (left) column of Figure 8, where the χ^2 function for the term "Bosnia" is monitored, Model 0 appears to always be about 2 and 1.85 times worse in terms of the number of transmitted messages when compared to the CAA(LG) and CAA(VA-200) approaches, respectively, for different threshold values (Figure 8(a)) using 10 sites.

Sensitivity to Threshold—Mutual Information (MI). Moving to the second and third row of the left column of Figure 8(a) we investigate the cases of the "Ipo" as well as "Febru" terms, monitoring the *MI* function across different threshold values for a 10-site configuration (note that *MI* is calculated as a logarithm, therefore the negative threshold values in that axis). In these graphs the peak that occurs at 0.4 and 0 for the "Ipo" and "Febru" involves an accumulation of synchronizations around the average value that the *MI* function possesses along the run. We again observe that CAA(LG) performs 1.75–2.1 times better than the Model 0 case for both monitored terms. Despite the fact that CAA(VA-200) is proved slightly worse compared to CAA(LG), it is still able to better amend the peak that occurs in "Febru" monitoring for a 0 threshold.

Sensitivity to Number of Sites. Eventually, switching to Figure 8(b), for the "Bosnia" term (top figure on the right) the relative benefits remain almost the same across all network scales. For the "Ipo" term monitoring (middle row on the right) we observe that Model 0 is steadily more than 1.8 times worse than the CAA(LG) choice across different scales. CAA(VA-200) performs worse than the CAA(LG) case for network

[5]We focus on comparing the performance of our prediction-based geometric monitoring techniques against Sharfman et al. [2006, 2007b] (Model 0), since we expect our prediction-based methods to give similar benefits when operating over the ellipsoidal bounding regions of Sharfman et al. [2008] to those seen in our current study using spherical constraints as in Sharfman et al. [2006, 2007b].



(a) cash-register-varying threshold (b) cash-register-varying sites

Fig. 8. Corpus dataset: Performance of our techniques in the cash register streaming paradigm.

configurations up to 80 sites. Nonetheless, the introduction of additional sites (along with their respective substreams) in 90-, 100-site cases, causes the *MI* function to always lie below the posed zero threshold since the "Ipo" term becomes more rare. This fact is perfectly read by the CAA(VA-200) approach, the transmitted messages of which approach zero. A similar behavior appears early at the third row of the right column of Figure 8(b) for the "Febru" case where the introduction of more than 10 sites causes *MI* to be negative, which is again accurately pinpointed by the CAA(VA-200) monitoring model reaching savings of three orders of magnitude size.

In the previously presented graphs we omitted the lines for the mere application of Model 1 to keep the diagram readable, since Model 1 shows almost identical (actually CAA can occasionally save a few tens of extra messages) behavior with its corresponding CAA applications. CAA possesses the ability to recognize the utility of Model 1 (the mere application of predictors as described in Section 4) in this setting and encompass it throughout its operation. On the other hand, this fact exhibits the ability of Model 1 to provide an efficient solution in environments where v_i values evolve relatively slowly. Nonetheless, as we will shortly present, this is not always true in scenarios where more dynamic updates occur.

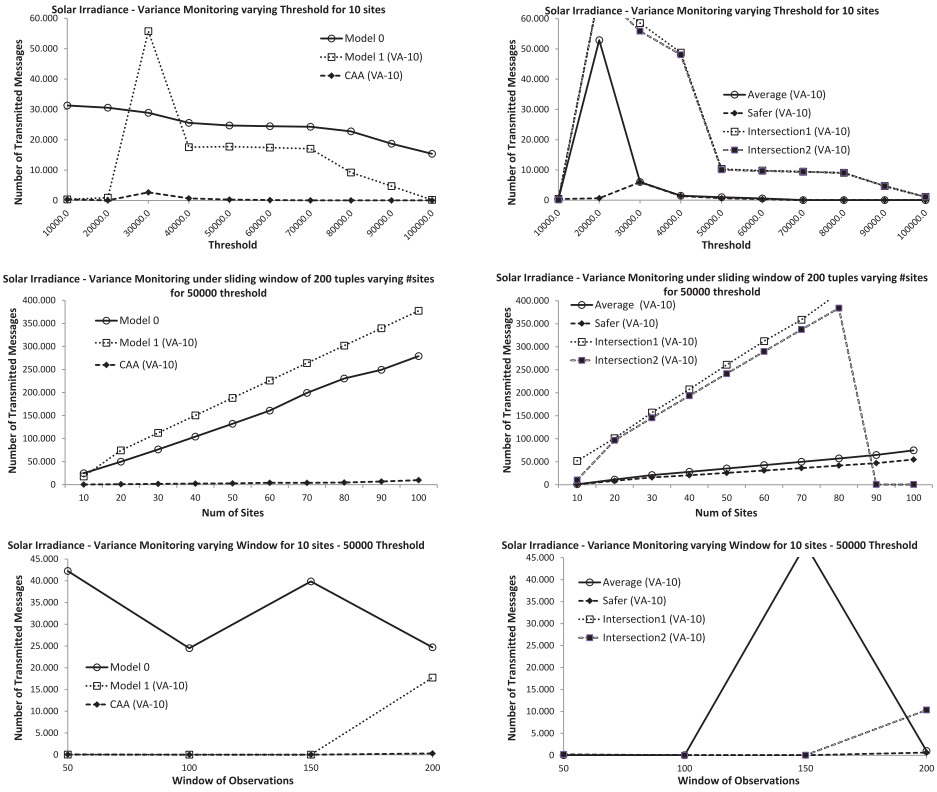
9.2. Weather Data—Sliding-Window Paradigm

We proceed to the sliding-window operation, using the Weather dataset and monitoring the Var and StN functions. Please note that in all the graphs presented in the current

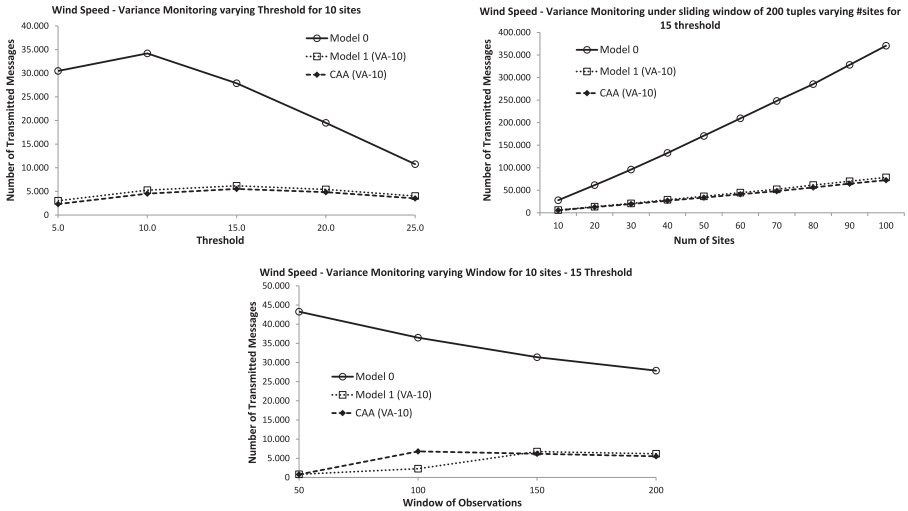
section the Linear Growth predictor is not applicable since it assumes that local vectors (v_i) uniformly evolve by a time-dependent factor (Section 3.2), which is obviously unrealistic for the physical measurements in the Weather data and the sliding-window application scenario. We thus compare the performance of Model 0, Model 1(VA-W), and CAA(VA-W) cases in our study. For the CAA(VA-W) monitoring model we again choose the window based on natural time units' division. We uniformly utilize a prediction window $W = 10$ which corresponds to the latest minutes of received observations, except for the Wind Peak data where $W = 50$ was chosen to adjust predictions to the expected frequency of the peaks in the wind blasts. For each dataset-function pair, the default value of the sliding-window size, over which the corresponding function is computed, is 200 measurements. However, we also perform a sensitivity analysis on this parameter as well.

Variance Monitoring. Figure 9(a) plots the performance of the techniques in the case of Var monitoring in the Solar Irradiance Data. On the left of the first row of the figure we observe that the cost of Model 0 ranges between 11 and 600 times larger than the cost yielded by CAA(VA-10) monitoring model, while CAA(VA-10) ensures up to 500 times lower cost even when compared with Model 1 across different thresholds. A case of particular interest shows up for a threshold of 30000. There, Model 1 shows a peak in the number of transmitted messages which are higher even when compared to Model 0. This happens due to the existence of specific sites whose drift vectors approach the threshold surface as noted in Figure 3. Obviously, increasing the threshold to 40000 alters the threshold surface and thus hinders the same sites to cause threshold violations. Nonetheless, CAA(VA-10) maintains low transmission cost due to the loosened intersection monitoring capabilities (Section 6.2) that it embodies. We will revisit this issue in the next section were we look into the operational details of the CAA monitoring model. In the meantime we note that the same applies for the second row on the left of Figure 9(a) where increasing the scale of the network results in CAA(VA-10) savings that reach a number of 30 times compared to Model 0 and they become even larger when compared to Model 1. Eventually, the third row on the left of the same figure shows the resilience of our techniques when altering the employed size of observations encapsulated in the sliding window for 10 sites. CAA(VA-10) shows similar behavior when enlarging the window. Model 1 yields more synchronizations for a window of 200 observations since enlarging the window causes the variance values within it to increase, and thus some sites approach the posed threshold of 50000. The lack of the alternative mechanisms that are incorporated in CAA leads sites merely utilizing Model 1 to threshold crossings. Finally, Model 0 exhibits high sensitivity to the number of values that local vectors (v_i) are built upon.

On the right column of Figure 9(a) we also provide indicative results of individual monitoring models' performance. To keep the diagram readable, instead of incorporating all seven value series on the same graph, we group the performances of the Average, Safer, Intersection1, and Intersection2 on the right diagram using the same scale on the vertical axis (parts of the series that exceed the used maximum value of the vertical axis are left outside the plot) and present the two graphs side by side so that the trends can easily be observed. In all three rows of the right column of Figure 9(a) we observe that the CAA approach holds the best performance in the vast majority of the cases, with the exception of Intersection1 performing better for 90 and 100 sites in the middle row of the figure. Apart from this, the Safer model more closely approaches CAA's performance upon varying the threshold (top figure on the right). When the degree of distribution of sites increases (middle row on the right) the Safer as well as the Average model, although less efficient, appear to be acceptable secondary choices, while varying the window size (bottom row on the right) leaves the



(a) Solar Irradiance, Variance Monitoring



(b) Wind Speed, Variance Monitoring

Fig. 9. Weather dataset: Performance of our techniques in the sliding-window streaming paradigm for variance monitoring.

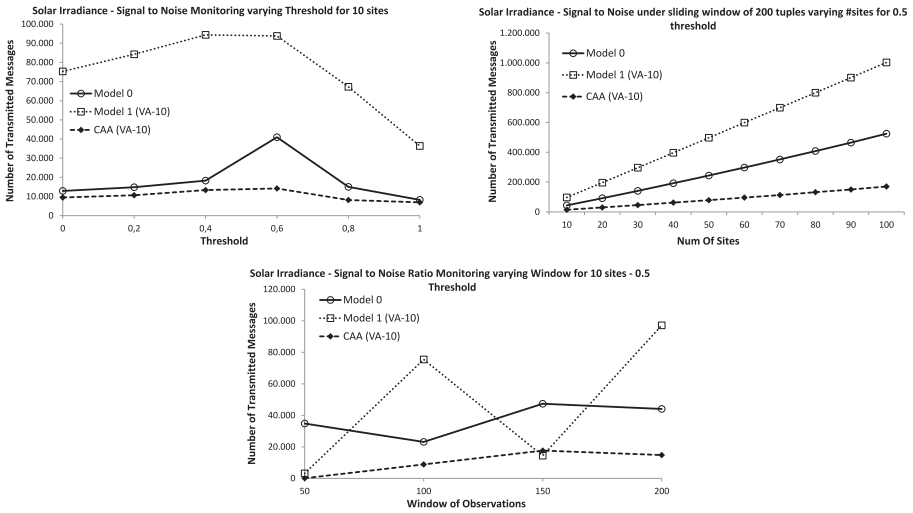
Intersection1 and Safer models unaffected. These observations (together with those noted in the upcoming figures) validate our initial claim that no sole alternative consistently holds the best performance.

Figure 9(b) presents corresponding results for Var monitoring in the Wind Speed dataset. Model 1 is slightly worse (almost 1%) in terms of transmitted messages compared to CAA(VA-10) when varying the threshold (first column at the top of the figure) and across different network scales (second column at the top), yet both result in savings ranging between 3 and 13 times compared to the message cost of Model 0. Furthermore, at the bottom row of Figure 9(b) we observe that both CAA(VA-10) and Model 1 remain resilient to altering the sliding-window size, ensuring significant benefits when compared to Model 0. Notice that, for a window of 100 observations, Model 1 performs better than CAA(VA-10). Recall that CAA resolves ties in the choice of the monitoring mechanism (Section 7) by picking a random model among those which did not cause a threshold crossing. Thus, when a particular model is always the appropriate choice, the adaptive CAA algorithm may sometimes end up transmitting slightly more messages. The results are similar for the Wind Peak dataset which we omit.

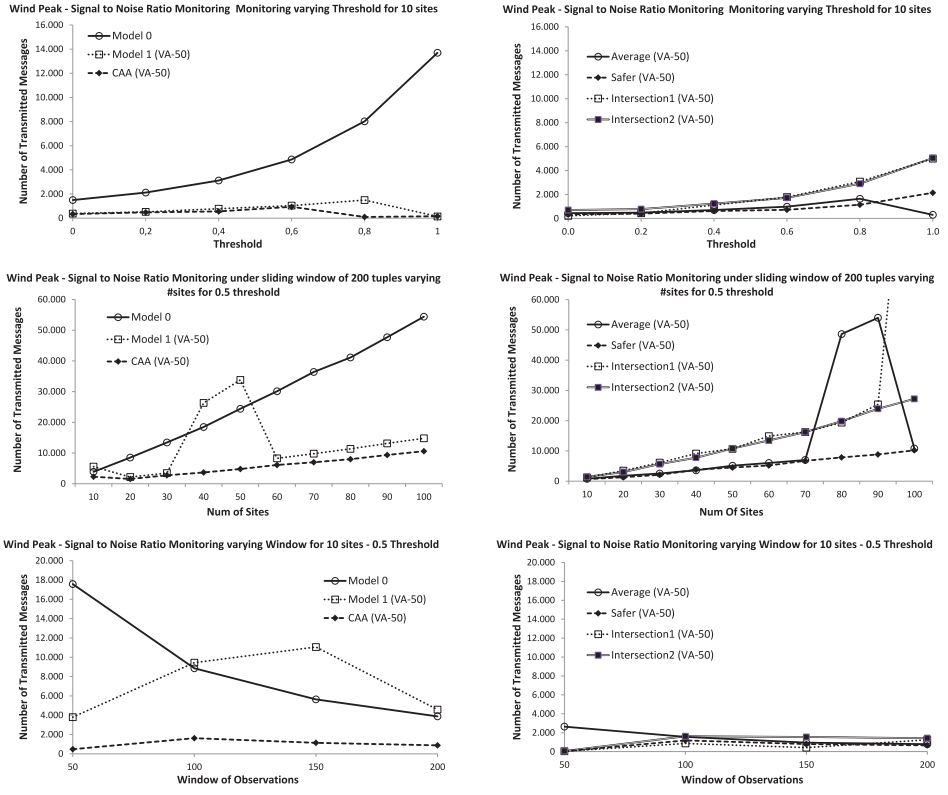
Signal-to-Noise Monitoring. In our next experiment we utilized the same motif for analyzing the performance of our techniques in monitoring the StN function. We begin our discussion with the Solar Irradiance dataset in Figure 10(a). CAA(VA-10) performs up to three times better than Model 0 when varying the threshold for 10 sites (left column at the top of the figure) and up to five times across network configurations of 10–100 sites (second column at the top). Model 1 is again the worst choice as it yields 2–5 times higher cost compared to Model 0 across different thresholds and appears over two times worse than Model 0 for different scales. At the bottom row of Figure 10(a), it is evident that CAA(VA-10) again remains mostly unaffected to different window sizes, while Model 1 exhibits a wavy behavior depending on the accuracy of the employed VA-10 predictor.

We then analyze the performance of the Wind Peak data in StN monitoring (Figure 10(b)) (the Wind Speed data had similar behavior). On the left column of the figure we observe that Model 1 and CAA possess similar performance across different thresholds with savings ranging between 4 and 85 times compared to the cost of Model 0. The same holds in large part when varying the network scale (middle row on the left in Figure 10(b)) where savings reach a factor of 5. An exception occurs for 40- and 50-site configuration cases. This is another occasion where site predictors lie close to the threshold surface for the given threshold of 0.5 and CAA manages to achieve increased savings due to the intersection monitoring capacity. As more sites are added in the subsequent steps (60–100-site configurations) the predicted estimate's (e^p) position is affected and thus the sites that were previously causing synchronizations (despite their restricted local constraints, i.e., balls) were ousted from the threshold surface, stabilizing the cost of Model 1. Eventually, as with the previously examined functions-dataset pairs, the CAA monitoring model is not sensitive to changes in the window size (bottom row on the left of Figure 10(b)) while Model 0 and Model 1 exhibit opposite trends upon enlarging it. Overall, Model 0's cost is 5 to 35 times the transmission cost of CAA, while savings against Model 1 range between 4 to 9 times across different window sizes.

As in Figure 9(a), we include performance results of individual model's adoption for the Wind Peak dataset and StN monitoring on the right column of Figure 10(b). It is once again obvious that the CAA approach manages to derive the most out of the available alternatives function as its performance approaches or exceeds the more efficient individual monitoring model, irrespective of varying the threshold (top figure on the right column of Figure 10(b)), the number of sites (middle row on the right), or the size of the window (bottom row on the right). In the case of varying the posed threshold,



(a) Solar Irradiance, StN Monitoring



(b) Wind Peak, StN Monitoring

Fig. 10. Weather dataset: Performance of our techniques in the sliding-window streaming paradigm for StN monitoring.

Table IV. Case Study: Solar-Var vs. Threshold Monitoring

Threshold	Model 0	Model 1	Average	Safer	Intersection 1	Intersection 2
10000	0	15	0	3	0	0
30000	105	0	0	0	4	25
50000	7	0	0	0	5	3
70000	0	0	1	0	0	1
90000	0	0	1	0	0	1

the Average and the Safer models' performance is closer to the one of the CAA's, while the same holds in most part upon varying the number of sites in the network. On the other hand, varying the window size appears to slightly favor the Intersection1 monitoring framework which exhibits slightly better performance compared to CAA.

9.3. CAA Operational Insights

We are now providing additional details regarding the choices that CAA makes throughout its operation to investigate the stem of its benefits. Since it is hard to present analytic statistics of alternative models' usage for every single case of the previously discussed graphs, we focus on two situations where Model 1 exhibits possibly unexpected peaks in the number of messages and examine the tools that CAA utilizes to avoid similarly high message exchange.

The first of the aforementioned cases regards the Solar Irradiance under Var monitoring against different thresholds and for 10 sites (left figure of Figure 9(a)). Table IV shows the CAA choices for different thresholds. We point out that for threshold $>$10000 (where it exhibits low costs) Model 1 is never employed by CAA. For the threshold-30000 case, Model 0 appears as the most frequent choice but it is only used during the first synchronizations until predictors are stabilized around the threshold surface (if Model 0 was continuously picked, CAA would have had similar cost to Model 0). Afterwards, the loosened intersection framework is chosen which safely leads the monitoring procedure to the decrement of the transmission cost, as shown in Figure 9(a).

Comparing the choices of the CAA approach in Table IV with the top-right graph in Figure 9(a), a contradiction seems to arise since the table shows that CAA often chooses the Intersection1 and Intersection2 tracking models, which, however, appear to lack performance upon operating individually in Figure 9(a). This reveals another important aspect of the integration of the alternative monitoring frameworks inside the CAA approach. This is because the intersection monitoring frameworks cause many synchronizations (almost continuous central data collection) at the beginning to the middle of the monitoring process and they carry this communication burden irrespectively of how well they perform afterwards. This explains the bad performance of Intersection1 and Intersection2 in Figure 9(a). On the other hand, the CAA approach (as Table IV shows) prefers some other tracking alternative while intersections are inefficient and elects the loosened intersection monitoring schemes only when they are ready to prevent synchronizations. In other words, CAA can achieve increased performance by determining the order by which the available alternatives will be used, thus possessing the ability to choose the proper alternative at the right time throughout the tracking process.

The second case we distinguished during our discussion was the peak that occurs when monitoring the Wind Peak data under the StN function for network configurations of different scale (middle figure of Figure 10(b)). As Table V shows, for 10 sites the savings CAA provides are mostly attributed to the Average and Safer model usage, while for 40, 50, and more sites, this is combined with the single time that Intersection2 is employed by CAA, which hinders communication for a considerable amount of time. These choices are in accordance with the results extracted in Figure 10(b) where

Table V. Case Study: Wind Peak-StN vs. No. Sites Monitoring

# Sites	Model 0	Model 1	Average	Safer	Intersection 1	Intersection 2
10	35	16	25	35	1	3
40	8	6	12	19	0	1
50	13	10	7	17	0	1
80	12	14	9	14	0	1
90	9	9	12	21	0	1

Model 1, the Average, and Safer models exhibit comparable performance to CAA's performance.

9.4. Impact of Predictor's Accuracy

In the current section we analyze the impact of the accuracy of the predictor on the performance that individual monitoring models and the CAA approach exhibit. We employ our Synthetic dataset produced as described in the introductory part of Section 9 and Coefficient of Variation (CoV) is the function under study. The window size employed in our VA predictors equals the window of observations' size. Since site measurements were generated based on a specific VA model, when no noise exists, local predictors are absolutely accurate and Model 1's convex hull is reduced to a single point. Figure 11 depicts corresponding results upon varying the percentage of noise imposed in local measurements, the posed threshold, the scale of the distributed network, and the window size.

At the top row of Figure 11 we present the communication cost of the monitoring models upon varying the posed threshold for a fixed amount of 25% noise in the streaming tuples. Once again the CAA approach achieves the best performance among the monitoring models, with Model 1 and Model 0 falling short up to an order and two orders of magnitude, respectively. As regards the rest of the alternatives (right column of the first row), the Average and the Safer Model may require up to 19 times more messages compared to CAA, Intersection1 appears up to four times worse while Intersection2 better approaches CAA's performance, being up to 3.5 times worse. Inspecting Model 1's cost, which is mainly affected by the accuracy of the local predictors, we observe a middle case for the resulting overhead for a threshold of 0.8. We therefore choose that threshold value to test our models against different noise percentages imposed on the original datasets.

At the second row of Figure 11 we present the communication cost of the monitoring models upon varying the percentage of noise (0%–50%) imposed on the site's local measurements for fixed 0.8 threshold value and 10-sites configuration. Model 1 remains resilient when injecting 0% to 10% noise (0% noise represents the case of perfectly accurate predictors). Its communication cost is doubled in each of the following steps as measurements become more obscured from VA predictors' estimations, until it reaches a maximum for 50% noise. As we expected, increasing the amount of noise deteriorates the performance of Model 1 as the monitored convex hull is enlarged. Model 0 appears the worst choice, being 2–37 times worse compared to Model 1. Reckoning that Model 0 assumes stability of sites' measurements, it is also severely affected by increasing the percentage of noise in the dataset, reaching the maximum communication cost for 50% noise. On the other hand, CAA is resilient for up to 30% injected noise, wherein its communication cost reaches its maximum value for the case of 40% and remains stable in the next 50% noise level. CAA appears up to 2.5 times better compared to Model 1 and at least seven times better than Model 0. The Average model better approaches CAA's performance for noise levels up to 20%. For 30% noise the Safer Model falls short by an order of magnitude while Intersection1 provides five times worse performance.

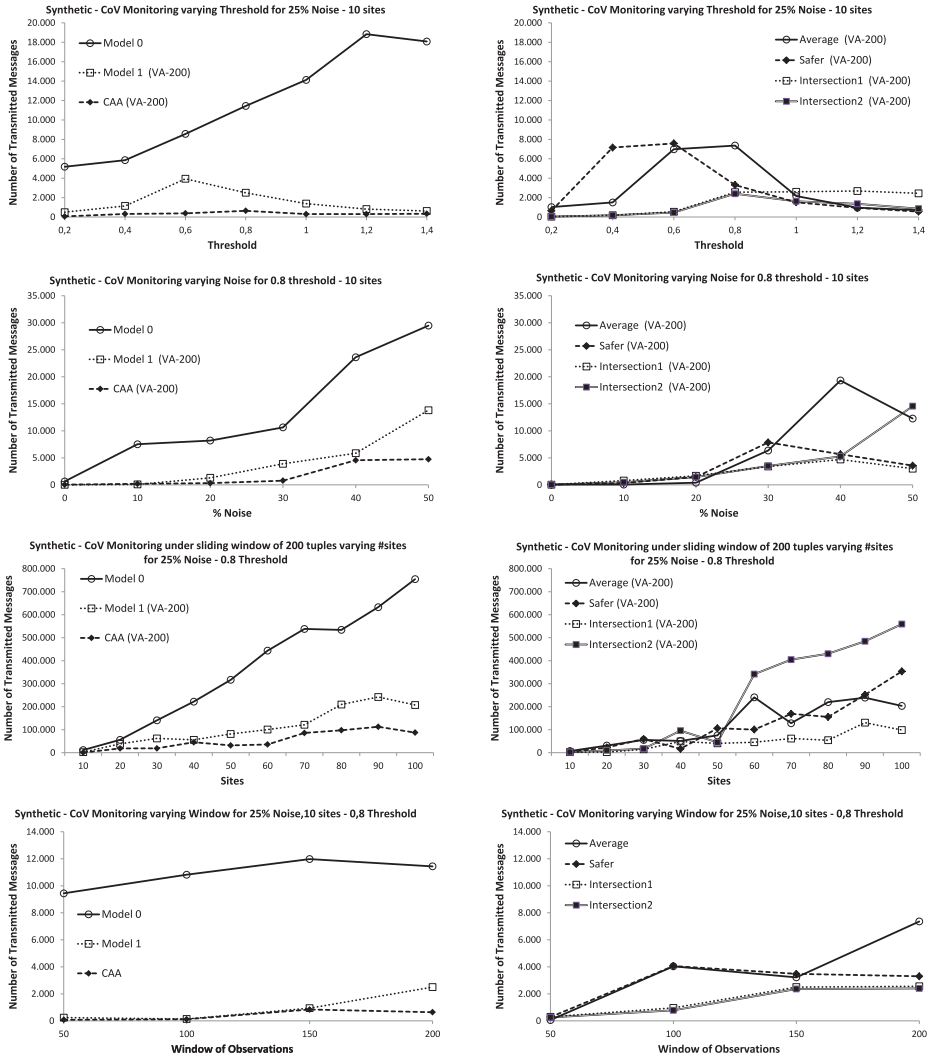
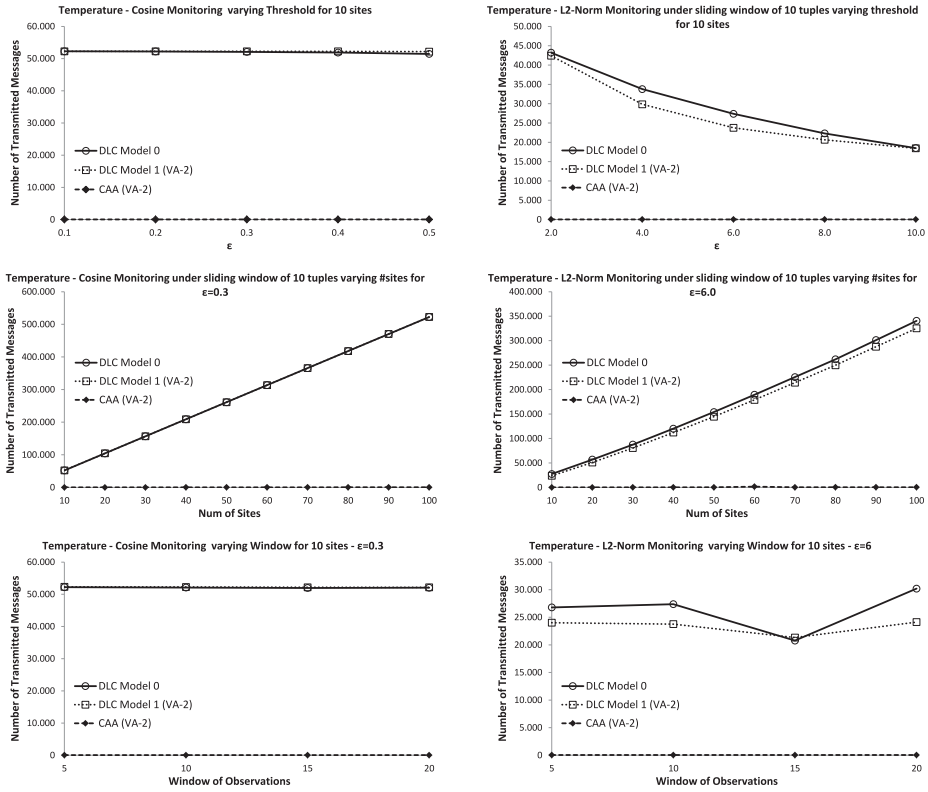


Fig. 11. Synthetic dataset: CoV monitoring.

Intersection2 as well as the Safer model resemble CAA for 40% noise even yielding lower overhead for the 50% case.

Another possible way of increasing the overall variation in the dataset is by increasing the number of sites participating in the distributed network, even when keeping the noise percentage unchanged (25%). Hence, in the third row of Figure 11, we present the communication cost of the various alternatives for different network scales. Increasing the number of sites aggravates the total error (dispersion from a perfect predictor) in the dataset and thus Model 1's cost boosts with the number of sites. Model 0 is up to nine times worse compared to the CAA approach while Model 1 provides up to two times worse cost. The rest of the alternatives fluctuate a lot across various network sizes with the middle case of the Safer Model appearing at least two times worse than CAA for network sizes above 50. The Intersection1 model resembles CAA with up to



(a) temperature, approximate Cos monitoring (b) temperature, approximate L_2 monitoring

Fig. 12. Weather dataset—Temperature: Approximate query answering performance.

30% higher number of transmitted messages, performing proportionately better for 70–80 sites in the network.

To complete the picture we also experiment with different window sizes as the number of noisy measurements in a window has an impact on the performance of our predictors and the corresponding monitoring models. In this case, we choose a VA predictor's window size equivalent to the window of observations. This choice stems from the intuition that, should no noise exist (the noise is again set 25%), the previous VA window should perfectly forecast the observations of the upcoming one. The outcomes presented at the bottom row of Figure 11 show in large part that the more noisy measurements that exist in a window, the higher the communication cost for all the presented alternatives. This situation is once again better remedied by the CAA approach which constitutes the more efficient choice amongst the presented alternatives.

9.5. Approximate Query Processing

Finally, in Figure 12 we present an experimental analysis regarding our prediction-based query answering procedures, introduced in Section 8, utilizing temperature measurements in the Weather dataset. As in Burdakis and Deligiannakis [2012], we perform L_2-norm as well as cosine (Cos) monitoring and measure the number of transmitted messages for different ε values (please recall that we enforce $|f(e^p(t)) - f(v(t))| \leq \varepsilon$), network scale configurations, and sliding-window sizes. We use the two most recent

readings in the VA predictor, namely $VA-2$. The adoption of the simple Decomposition to Local Constraints (DLC) approach provides no communication reduction in the case of Cos monitoring while allowing relatively small savings which range between 1% to 12% compared to the DLC Model 0 in L_2 approximate tracking. This happens due to the tight approximation requirements, that is, the magnitudes of the ε value that are placed while performing approximate Cos answers. The latter fact shows that the mere adoption of good predictors does not achieve significant message reduction when the δ parameter which determines the radius of the constructed hypersphere (see Inequality (7) and Figure 7) receives small values. However, it is important to note that the number of transmitted messages by the DLC approach is still always half the number of messages that the mere adoption of Model 1 and Model 0 (omitted in Figure 12) would require, respectively. This happens because, as noted in Section 8.1.1, the DLC framework does not require any information to be broadcasted back to the sites. Hence, one-way communication suffices during each synchronization process, which in turn halves the number of transmitted messages.

On the contrary, the application of CAA on the techniques of Section 8.1.2 (diamonded line approaching the horizontal axis in Figure 12) is capable of providing communication savings from 400 times to three orders of magnitude compared to DLC Model 1, which are even higher when compared to DLC Model 0 for both of the monitored functions. Hence, we have validated the claim of Section 8.1.2 regarding the flexibility of the CAA approach to simultaneously tune both the size and the position of the local constraints so as to improve approximate query answering performance.

10. CONCLUSIONS

In this article, we presented a thorough study regarding the generalization of the geometric approach of Sharfman et al. [2006, 2007b] by prediction models' adoption. After identifying the peculiarities exhibited by predictors upon their implementation in the aforementioned environment, we developed a solid theoretic framework composed of conditions rendering predictors more likely to lighten the communication burden. We proposed algorithms incorporating these conditions and expanded on their relaxed versions, along with extensive theoretical analysis on their expected benefits. We also directed our study on approximate $f(v(t))$ tracking scenarios and expanded our algorithms with enrichments tailored for efficiently fulfilling that particular task. Our ongoing efforts in this area explore the choice of optimal reference points, as in Sharfman et al. [2008] that, together with probing only a representative subset of the sites, may loosen the conditions of strict containment.

REFERENCES

B. Babcock and C. Olston. 2003. Distributed top-k monitoring. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'03)*. 28–39.

S. Burdakis and A. Deligiannakis. 2012. Detecting outliers in sensor networks using the geometric approach. In *Proceedings of the 28th International Conference on Data Engineering (ICDE'12)*. 1108–1119.

G. Cormode and M. Garofalakis. 2005. Sketching streams through the net: Distributed approximate query tracking. In *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB'05)*. 13–24.

G. Cormode and M. Garofalakis. 2007. Streaming in a connected world: Querying and tracking distributed data streams. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'07)*. 1178–1181.

G. Cormode and M. Garofalakis. 2008. Approximate continuous querying over distributed streams. *ACM Trans. Database Syst.* 33, 2.

G. Cormode, M. Garofalakis, S. Muthukrishnan, and R. Rastogi. 2005. Holistic aggregates in a networked world: Distributed tracking of approximate quantiles. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'05)*. 25–36.

G. Cormode, S. Muthukrishnan, and K. Yi. 2011. Algorithms for distributed functional monitoring. *ACM Trans. Algor.* 7, 21:1–21:20.

G. Cormode, S. Muthukrishnan, and W. Zhuang. 2007. Conquering the divide: Continuous clustering of distributed data streams. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE'07)*. 1036–1045.

A. Das, S. Ganguly, M. Garofalakis, and R. Rastogi. 2004. Distributed set-expression cardinality estimation. In *Proceedings of the 13th International Conference on Very Large Data Bases (VLDB'04)*. Vol. 30. 312–323.

A. Deligiannakis, Y. Kotidis, and N. Roussopoulos. 2004. Compressing historical information in sensor networks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'04)*. 527–538.

A. Deligiannakis, Y. Kotidis, and N. Roussopoulos. 2007. Dissemination of compressed historical information in sensor networks. *The VLDB J.* 16, 4, 439–461.

M. Garofalakis, J. Gehrke, and R. Rastogi. 2002. Querying and mining data streams: You only get one look a tutorial. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'02)*. 635.

M. Garofalakis, D. Keren, and V. Samoladas. 2013. Sketch-based geometric monitoring of distributed stream queries. *Proc. VLDB Endow.* 6, 10, 937–948.

N. Giatrakos, A. Deligiannakis, M. Garofalakis, I. Sharfman, and A. Schuster. 2012. Prediction-based geometric monitoring over distributed data streams. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'12)*. 265–276.

N. Giatrakos, Y. Kotidis, A. Deligiannakis, V. Vassalos, and Y. Theodoridis. 2013. In-network approximate computation of outliers with quality guarantees. *Inf. Syst.* 38, 8, 1285–1308.

R. Gupta, K. Ramamritham, and M. Mohania. 2013. Ratio threshold queries over distributed data sources. *Proc. VLDB Endow.* 6, 8, 565–576.

L. Huang, M. Garofalakis, J. Hellerstein, A. Joseph, and N. Taft. 2006. Toward sophisticated detection with distributed triggers. In *Proceedings of the SIGCOMM Workshop on Mining Network Data (MineNet'06)*. 311–316.

L. Huang, X. Nguyen, M. Garofalakis, and J. M. Hellerstein. 2007. Communication-efficient online detection of network-wide anomalies. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM'07)*. 134–142.

A. Jain, J. M. Hellestein, S. Ratnasamy, and D. Wetherall. 2004. A wakeup call for internet monitoring systems: The case for distributed triggers. In *Proceedings of the Hot Topics in Networks Workshops (HotNets'04)*.

R. Keralapura, G. Cormode, and J. Ramamirtham. 2006. Communication-efficient distributed monitoring of thresholded counts. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'06)*. 289–300.

D. Keren, I. Sharfman, A. Schuster, and A. Livne. 2012. Shape sensitive geometric monitoring. *IEEE Trans. Knowl. Data Engin.* 24, 8, 1520–1535.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. 2004. RCV1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.* 5, 361–397.

Z. Liu, B. Radunovic, and M. Vojnovic. 2012. Continuous distributed counting for non-monotonic streams. In *Proceedings of the 31st Symposium on Principles of Database Systems (PODS'12)*. 307–318.

S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. 2005. TinyDB: An acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.* 30, 122–173.

C. Olston, J. Jiang, and J. Widom. 2003. Adaptive filters for continuous queries over distributed data streams. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'03)*. 563–574.

G. Sagy, D. Keren, I. Sharfman, and A. Schuster. 2010. Distributed threshold querying of general functions by a difference of monotonic representation. *Proc. VLDB Endow.* 4, 46–57.

G. Sagy, I. Sharfman, D. Keren, and A. Schuster. 2011. Top-k vectorial aggregation queries in a distributed environment. *J. Parallel Distrib. Comput.* 71, 2, 302–315.

I. Sharfman, A. Schuster, and D. Keren. 2006. A geometric approach to monitoring threshold functions over distributed data streams. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'06)*. 310–312.

I. Sharfman, A. Schuster, and D. Keren. 2007a. Aggregate threshold queries in sensor networks. In *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS'07)*. 1–10.

I. Sharfman, A. Schuster, and D. Keren. 2007b. A geometric approach to monitoring threshold functions over distributed data streams. *ACM Trans. Database Syst.* 32, 4.

I. Sharfman, A. Schuster, and D. Keren. 2008. Shape sensitive geometric monitoring. In *Proceedings of the 27th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'08).* 301–310.

K. Yi and Q. Zhang. 2013. Optimal tracking of distributed heavy hitters and quantiles. *Algorithmica* 65, 1, 206–223.

Q. Zhang, J. Liu, and W. Wang. 2008. Approximate clustering on distributed data streams. In *Proceedings of the 24th International Conference on Data Engineering (ICDE'08).* 1131–1139.

Received August 2013; revised January 2014; accepted March 2014