ICT, FET Open

LIFT ICT-FP7-255957

Using Local Inference in Massively Distributed Systems

Collaborative Project

D 1.1

Algorithms for Safe Zone Construction
1.10.2010 – 30.09.2011

Contractual Date of Delivery:	30.09.2011
Actual Date of Delivery:	30.09.2011
Author(s):	Assaf Schuster, Daniel Keren, Minos Garofalakis
Institution:	Technion
Workpackage:	WP 1
Security:	PU
Nature:	R
Total number of pages:	6

Project coordinator name: Michael May

Project coordinator organisation name: **Revision:** 3

Fraunhofer Institute for Intelligent Analysis
and Information Systems (IAIS)
Schloss Birlinghoven, 53754 Sankt Augustin, Germany
URL: http://www.iais.fraunhofer.de

Abstract:

This document is the LIFT deliverable of WP1 for the first review period (01.10.2010 – 30.09.2011). The document contains an overview on algorithms for safe zone construction and then presents the work achieved during the last reporting period of the LIFT project.

Revision history

Administration Status			
Project acronym: LIFT			**ID:** ICT-FP7-255957
Document identifier:	D 1.1 Algorithms for Safe Zone Construction (01.10.2010 – 30.09.2011)		
Leading Partner:	Technion		
Report version:	3		
Report preparation date:	20.09.2011		
Classification:	PU		
Nature:	REPORT		
Author(s) and contributors:	Assaf Schuster (Technion), Daniel Keren (HU), Minos Garofalakis (TUC)		
Status:	-	Plan	
	-	Draft	
	-	Working	
	-	Final	
	x	Submitted	

Copyright

Project funded by the European Community under the Information and Communication Technologies Programme
Contract ICT-FP7-255957

Report for Project "LIFT", September 2011. WP1: Theory, Algorithms and Optimisation

A major goal of the LIFT project is to reduce the global testing and monitoring of massively distributed systems to the checking of local constraints, thus minimizing communication overhead. During the first year, research in this direction concentrated on the following objectives:

1. Generalizing previous work on geometric monitoring.
2. Applications of the mathematical and algorithmic techniques developed so far to solve a problem in static distributed databases.
3. Developing the general theory of the safe-zone approach, and studying its complexity.
4. Practical considerations for solving the safe-zone problem (hierarchical clustering and geometric considerations).
5. Concrete examples of monitored functions **(MEDIAN – CRETE TEAM)**

We next elaborate on the above items, but first, a very short summary of the goals and concepts involved:

- We assume that given is a distributed system, where each node locally holds a dynamic data vector.
- A global function is given which is defined on the aggregate of the local vectors. There are no restrictions provided on this function, i.e. it does not have to be linear, monotonic, convex etc. This function can be defined directly on the local vectors or also on some parameters derived from them. We refer to it as the *monitored function.*
- A threshold is provided, and the system must issue an alert when the possibility exists that the value of the monitored function crosses this threshold.
- The monitoring algorithm assigns to each node a subset of the data space, referred to as its *safe-zone.* It is guaranteed that as long as each local vector is inside its safe-zone, the monitored function did not cross the threshold. It is desired that the safe-zones be simple while reducing the number of *false alarms* (see next item).
- A *false alarm* occurs when a local vector breaches its safe-zone, but the monitored function did not cross the threshold. The main challenge is to reduce the number of false alarms, since each safe-zone breach results in communication.
- In case of a safe-zone(s) breach, it is desirable to "balance" the local vector(s) causing the breach with minimal communication, or instead to conclude that the global function crossed the threshold and re-synchronize the system.

1 Generalizing previous work on geometric monitoring

We follow the framework and definitions used throughout our previous work on monitoring. Here we list only the most basic definitions; for further details, please see the attached papers. Each node N_i holds a dynamic data vector v_i. Given are a general function defined on the average vector, $f(\frac{v_1+\ldots+v_n}{n})$, and the system-wide condition which has to be monitored is whether $f(\frac{v_1+\ldots+v_n}{n}) \leq T$. In [4] we explain how this definition covers many cases of practical interest, where the local vectors v_i are possibly augmented by some parameters. Denote the "admissible set" of $f()$ by $S = \{v|f(v) \leq T\}$. Thus we must find local conditions, defined independently for each i, which guarantee that $\frac{v_1+\ldots+v_n}{n} \in S$. These conditions are themselves represented by geometric containment, i.e. $v_i \in S_i$ for some sets S_i. For the monitoring algorithm to be correct, it must hold that $v_i \in S_i \rightarrow \frac{v_1+\ldots+v_n}{n} \in S$ (in other words, the *Minkowski average* of the S_i, $\frac{S_1\oplus\ldots\oplus S_n}{n}$, must be a subset of S). For the algorithm to be efficient, the sets S_i must be "large", in a sense to be made precise later.

Initial work on geometry-based monitoring, [6, 5] , introduced the definition of local constraints via spherical "bounding regions" . These constraints are defined as follows: given some global reference vector p, the i-th node constructs the ball supported by p and v_i, and checks whether it is contained in S. If this holds at each node, it is guaranteed that $\frac{v_1+\ldots+v_n}{n} \in S$, and no communication is required. Thus the set of all points such that the ball supported by them and p is contained in S forms a "safe-zone"; as long as all the local vectors are in their respective safe-zones (which are translates of each other, i.e. they have identical shape but occupy different locations), the system-wide condition is guaranteed to hold. This observation is the departure point for a far more general treatment of the monitoring problem, centered around a broader definition of safe-zones, which allows to define safe-zones which are 1) optimal, and 2) different nodes are assigned different safe-zones, which match their data distribution.

The first extension of the basic geometric paradigm is developed in [2]. In this paper we prove that every safe-zone defined as above is *convex* (this is regardless of the properties of the monitored function $f()$). On the other hand, since every convex set is closed under averaging, it follows that every convex subset of S can serve as a safe-zone. Therefore, it makes sense to seek an *optimal* convex subset of S and use it as a safe-zone. There are a few possible definitions for "optimal"; generally, a good safe-zone is one in which the local vectors will remain for a long time (this is important since every breach of the safe-zone results in communication), and also one which is relatively simple to define. The latter requirement follows from the need of every node to continuously test whether its dynamic data vector is in the safe-zone, and – especially in the case of thin nodes (i.e. battery-operated sensors) – it is desirable that this testing process will consume as little resources as possible.

In [2] we have experimented with safe-zones consisting of convex polyhedra, and compared them with the basic geometric method in [6, 5]. Data was restricted to lie in two-dimensional Euclidean space, obtained from wind measurements of the El-Nino system in two directions. Using the optimized convex subsets as safe-zones improved performance over previous work.

As compared to the bounding volume approach, the method in [2] has the advantage that it searches for an optimal safe-zone. The drawback is that an (offline) difficult optimization has to be solved – finding an optimal convex subset of the potentially high-dimensional, complicated set S. We have made some progress in this direction for the problem of monitoring the median function, which is very important in many applications (robust analysis, sketching, ranking, finding percentiles and heavy-hitters over distributed streams).

2 Application to distributed databases

This research, recently presented in VLDB [4], deals with *threshold queries* of general functions over a distributed database. The goal of a threshold query is to detect all objects whose score exceeds a given threshold. This type of query is used in many problems in data mining, event triggering, and top-k selection. Often, threshold queries are performed over distributed data, where an object's score is computed by aggregating the value of each attribute, applying a scoring function over the aggregation, and thresholding the function's value. However, joining all the distributed relations to a central database might incur prohibitive overheads in bandwidth, CPU, and storage accesses. Efficient algorithms required to reduce these costs exist only for monotonic aggregation threshold queries and certain specific scoring functions. Thus, this problem is very closely related to the LIFT paradigm of minimizing communication overhead in massively distributed systems. In [4] we presented a novel approach for efficiently performing general distributed threshold queries. First, a solution for monotonic functions was presented, which applies and develops the geometric ideas presented in the bounding volumes approach; then, we introduced a technique to solve for other functions by representing them as a difference of monotonic functions. Experiments with real-world data demonstrated the method's effectiveness in achieving low communication and access costs.

3 Theory and Complexity Analysis of the Safe-Zone Problem

Here we define the general problem of safe-zones, and study its complexity. The material in this section is contained in a soon to be submitted paper (included).

The main novelty in the geometry-based monitoring paradigm we're currently pursuing is that it enables each node to have its uniquely shaped safe-zone; this, as opposed to the work in [6, 5, 2], in which every node is assigned an identically shaped safe-zone. In this general formulation, and using the terminology of Section 1, we demand that $\frac{S_1 \oplus \ldots \oplus S_n}{n} \subset S$; this constraint guarantees that the monitoring algorithm is *correct*, i.e. every violation of the global constraint will be captured by at least one node. To guarantee *efficiency*, which is this case means minimal communication, we maximize the overall probability of the local data to remain within their respective safe-zones S_i. Formally, the optimization problem for the safe-zones is defined by

$$\text{Maximize} \int_{S_1} p_1 dv_1 \int_{S_2} p_2 dv_2 \ldots \int_{S_n} p_n dv_n$$

$$\text{Subject to: } \frac{S_1 \oplus S_2 \ldots \oplus S_n}{n} \subset S$$

where p_i is the probability distribution function (p.d.f henceforth) of the data at the i-th node. The p.d.f can be estimated from measurements, and its accuracy boosted by a dynamic model and prior knowledge. To the optimization paradigm above, one may add a penalty term on the complexity of the safe-zones; as noted in Section 1, simple safe-zones are preferable, as they allow to test the local conditions faster. Hence, as common in the data modeling community, one may replace the target function $\int_{S_1} p_1 dv_1 \int_{S_2} p_2 dv_2 \ldots \int_{S_n} p_n dv_n$ with $\int_{S_1} p_1 dv_1 \int_{S_2} p_2 dv_2 \ldots \int_{S_n} p_n dv_n + \lambda \sum_{i=1}^{n} L(S_i)$, where $L()$ is a measure of the safe-zone's complexity, or length of description, and λ a weight factor.

Note that the geometric constraint and the target function which needs to be maximized have to reach a "compromise"; figuratively speaking, the Minkowski average constraint forces the safe-zones to be small, while the probability increases as the safe-zones become larger. This tradeoff is central in the solution of the optimization problem.

The geometric interpretation of the optimization problem is next summarized and a descriptive example (from [3] in the attached) provided, which illustrates the advantage of using differently shaped safe-zones at different nodes.

- The optimal safe-zones try to match the shape of the p.d.f (probability density functions) at each node, so as to maximize the probability of the local data to fall in its safe-zone. For example, a distribution which is wide along the x-axis and narrow along the y-axis should be assigned an safe-zone which is wide horizontally and narrow vertically.

- The Minkowski average of the safe-zones should tightly approximate S (from the inside). If it fills a relatively small part of S, this means that the safe-zones can be enlarged and the value of the optimized function increased (Fig. 1).

Figure 1: Left: left and right denote samples of two-dimensional data tuples at two nodes. The p.d.f at both nodes are Gaussian (normal) distributions. S, which must contain the Minkowski average of the two safe-zones, is the dotted ellipse in the middle. The point cloud in the middle is a sample of the global data tuples, obtained by averaging the data at both nodes. The depicted safe-zones (polygons outlined in black) fit the local data well, alas they are illegal, since their Minkowski average (continuous dark line in the middle) is not inside S. Right: two legal safe-zones are depicted. The safe-zones attempt to fit the data, while satisfying the constraint. This figure demonstrates the trade-off between the fit of the safe-zones to the data and the necessity to maintain their Minkowski average inside S. Note that the Minkowski average "sticks" to S; this is because the safe-zones must be as large as possible (where by "large" it is meant that the integral of the p.d.f over them is large). If the Minkowski average does not "stick" to S, it is possible to enlarge the safe-zones and hence the integral of the p.d.f over them, thus they would not have been optimal.

3.1 Complexity Issues

This formulation of the safe-zone problem by means of optimization allows to study its complexity. It turns out that if no assumptions are made, the problem is NP-Hard even in the simplest possible scenario (two nodes, one-dimensional data). This follows by reduction to the *bi-clique* problem, which is in fact known to be not just NP-Hard but inapproximable. In this reduction, the set S is discrete and disconnected; however, the problem is still hard for more than two nodes when S is a one-dimensional interval, and for two nodes it is hard even when S is restricted to be convex and the dimension is ≥ 4. Thus, unless further restrictions are imposed, then – from the point of view of complexity theory – it is not possible to solve nor approximate the optimal solution, and efficient heuristics are required. We have pursued some such heuristics and, in addition to the general problem, a major effort in future work will concentrate on specific classes of problems for which optimal solutions can be found or approximated.

4 Practical Algorithms for Solving the Safe-Zone Problem

In addition to studying the complexity of the optimal safe-zone problem, we have pursued its solution for real data and commonly used functions. We applied safe-zones which were defined as polyhedra in Euclidean space (dimensions of the data vectors, over which the functions are computed, ranged from 2 to 5). The data and monitored functions are described in length in [3]. After many experiments, we concluded that, from the practical point of view, the parameter which most strongly hinders the solution is the number of nodes n, since, typically, the number of safe-zones parameters which need to be optimized over is proportional to n. To solve this problem, we applied a hierarchical clustering scheme, which allows to solve the problem recursively for very small (two or three) nodes; these nodes are composed of the union of clusters of the original nodes, where the clustering is based on a similarity measure between the nodes' p.d.f.'s.

4.1 Hierarchical Clustering

The following, summarized from [3], illustrates the hierarchical clustering approach by depicting how it proceeds for a small number of nodes. In the experiments, we applied it for up to 240 nodes.

We start by partitioning the entire set of nodes into a small number of clusters, which can be thought of as "super nodes", each containing the union of data of the nodes in the respective cluster. We fit safe-zones to these "super nodes", and continue recursively, by partitioning each "super node" into clusters, and so on. This process constructs (top-down) a tree of super nodes. Its leaves can be either individual nodes, or node clusters which are uniform enough that they do not require further partitioning into smaller clusters, and can all be assigned safe-zones with identical shapes.

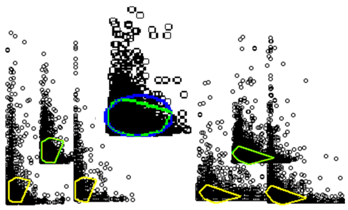

Figure 2: Hierarchical clustering example

To illustrate the hierarchical clustering algorithm, we present (Fig. 2) an example with four nodes (data is taken from [1]). Each of the four nodes' data is represented by a scatter diagrams at the bottom row. The safe-zones fit at the nodes are pentagons. The nodes are first separated into two clusters (or "super nodes") depicted in the middle row. Each of these "super nodes" is represented by sampling and averaging from the two nodes which it represents, and the entire data (top row) is represented by sampling and averaging from the two "super nodes"; it is the root of the cluster tree. S is the blue ellipse in the root (the root node was scaled for better view). An ellipse corresponds to monitoring a quadratic function (see also Section **??**). The two safe-zones corresponding to the "super nodes" are in green (as is their Minkowski average, depicted inside S). The yellow safe-zones are the ones computed in the original nodes; note that the Minkowski sum of the two left(right) ones is constrained to lie inside the left(right) safe-zone in the middle row. Thus, assigning four safe-zones at the nodes was achieved by only solving optimization problems involving two safe-zones. This leads to a considerably faster solution than when solving for all four nodes simultaneously (typically, the time complexity of solving a general optimization problem increases very rapidly with the number of parameters, so optimizing three times over two pentagons is much faster than a single optimization over four pentagons), and also allows parallelization. This example also demonstrates a typical situation – nodes with similar data distributions reside in the same clusters, and are assigned similar safe-zones; when many nodes are present, it is usually possible to prune the cluster tree and assign the same safe-zone to all nodes in a cluster, given that it is uniform enough. This, too, can save a great deal of computation.

5 MEDIAN – CRETE TEAM

6 Survey of Included Publications

The attached publications cover the theory and applications of the safe-zone approach developed in the project's first year. A short summary follows:

1. In [2], the first extension of the basic geometric approach to monitoring is described. After proving that the spherical bounding volumes defined in [6, 5] yield convex regions at the zones, and that, vice-versa,

convex subsets of the function's admissible region always define legal safe zones, we define a notion of optimality for such convex subsets, and compute them for some examples of two-dimensional data. Herein lies the main difference between the previous work in which the safe zones are explicitly defined and the new approach which seeks to find optimal safe zones (at the price of an off-line optimization problem). This work is still limited in the sense that all nodes are assigned safe zones which are translates of one set.

2. In [4], the bounding volume approach is applied to solve a general threshold problem over static, distributed databases, where the thresholded function does not have to satisfy a condition such as linearity, convexity, or monotonicity. First, the problem is solved for monotonic functions. Using bounding volumes, we show how local filtering can prune out a high percentage of the objects. Then general functions are treated by representing them as a difference of monotonic functions.

3. In [3] the problem of safe zones is studied in a more general framework, where each node can be assigned a save zone tailored to the distribution of its data. The complexity of this general problem is studied and it is proved to be NP-Hard in most interesting cases. A few steps are taken towards a practical solution, especially the idea of hierarchical clustering which allows to reduce the case of many nodes to distinct problems with two nodes each, while barely sacrificing accuracy.

4. ADD CRETE TEAM

References

[1] The European air quality database, http://dataservice.eea.europa.eu/dataservice/metadetails.asp?id=1079.

[2] D. Keren, I. Sharfman, A. Schuster, and A. Livne. Shape sensitive geometric monitoring. *IEEE TKDE, to appear*.

[3] Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, Izchak Sharfman, and Assaf Schuster. Simple and efficient safe-zones for distributed monitoring, in submission.

[4] G. Sagy, D. Keren, I. Sharfman, and A. Schuster. Distributed threshold querying of general functions by a difference of monotonic representation. *PVLDB*, 4(2):46–57, 2010.

[5] Izchak Sharfman, Assaf Schuster, and Daniel Keren. A geometric approach to monitoring threshold functions over distributed data streams. In *SIGMOD Conference*, pages 301–312, 2006.

[6] Izchak Sharfman, Assaf Schuster, and Daniel Keren. Shape sensitive geometric monitoring. In *PODS*, pages 301–310, 2008.