Distributed Data-Aware Representative Clustering for Geosensor Networks Data Collection

Ilka A. Reis[1,2], Gilberto Câmara[2], Renato Assunção[1], Antônio Miguel V. Monteiro[2]

[1]Departamento de Estatística – Universidade Federal de Minas Gerais (UFMG)
31270-901– Belo Horizonte – MG – Brazil

[2] National Institute for Space Research (INPE)
12227-010 – São José dos Campos – SP – Brazil

{ilka,gilberto,miguel}@dpi.inpe.br, assuncao@est.ufmg.br

Abstract. *Distributed clustering algorithms play an important role in energy-efficient data collection proposals for geosensor networks. The available data-aware clustering algorithms build clusters around nodes' representatives, which represent their associated nodes individually. We propose to build clusters around clusters' representatives, which are able to produce data summaries that are better estimates to their associated nodes' data. We present the Distributed Data-aware Representative Clustering (DARC) algorithm. We have concluded the DARC builds more homogeneous clusters and produce data summaries that estimate the nodes' data with the smallest error, if compared with the usual data-aware clustering proposals.*

1. Introduction

Geosensor networks comprise small electro-mechanical devices that sample spatio-temporal fields, collecting data and sending them to a remote base station by wireless communication. The main goal of a geosensor network is to keep the network's database updated while saving the limited nodes' energy as much as possible.

Since the wireless communication is the main consumer of the nodes' energy, an alternative to reduce the energy consumption is to limit the nodes communication to a local neighborhood, building clusters of nodes. Then, clustering algorithms have gained an important role on the energy-efficient data collection in geosensor networks. They are the basis for the cluster-based data routing protocols (for example, [Heinzelman et al. 2000] and its variations) and some schemes of spatial and spatio-temporal data suppression (for example, [Kotidis 2005] and [Tulone and Madden 2006], respectively).

A cluster-based data routing protocol groups the neighboring nodes around a cluster head, which aggregates the data of the cluster members and sends the summary to the base station. In addition to localize the nodes communication, this strategy reduces the data volume traveling through the network.

A scheme for spatio-temporal data suppression uses the temporal correlation among the readings of a same node and the spatial correlation among the observations of neighboring nodes to build the expected behavior for the nodes' data. The base station and the nodes agree on this behavior. The nodes send data to the base station only if these data differ from their known expected behavior. To deal with the spatial part of the suppression scheme, an alternative is to cluster the nodes around a head node.

To meet the energy constraints of a geosensor network, a clustering algorithm must group the nodes using only local messages (*distributed clustering*). The distributed clustering algorithms for sensor networks can be divided into two categories: *ordinary clustering* (for example [Heinzelman et al. 2000]) and *data-aware clustering* (for example [Kotidis 2005; Tulone and Madden 2006]). The difference between ordinary and data-aware clustering proposals is on the definition of the nodes' neighborhood. Ordinary clustering only considers the geographical proximity to define the nodes' neighbors, whereas data-aware clustering constrains this definition, considering also the similarity among the data the nodes sense [Reis et al. 2007].

Distributed clustering algorithms have two main tasks: to choose the head nodes and to associate the neighboring non-head nodes to the chosen heads. In the data-aware proposals, a non-head node joins the most similar neighbor head. If there is not a head in the neighborhood of a non-head node, it remains alone (a solitary node). The differences between the data-aware proposals are in the first task (heads choice). Despite of adopting different methods to choose the heads, the usual proposals have the same goal: to find a head node to represent each associated node *individually* (a *representative node* [Kotidis 2005]), acting as a *nodes' representative* during the data collection.

We look for a distributed data-aware clustering proposal that produces *clusters' representatives*, that is, head nodes that are the result of an *agreement* among neighboring nodes. This agreement considers the interest of all participating nodes. By the interest of a node, we mean "to join its most similar neighbor". In the current proposals, the head choice considers the interests of the nodes individually. Then, the chosen head cannot be considered a cluster's representative.

We believe a cluster built around a cluster's representative produces *more homogeneous* data than a cluster built around a nodes' representative. As a result, a cluster's representative calculates data summaries that are better estimates for the data of its associated nodes [Reis et al. 2007].

The main goal of this paper is to present a distributed data-aware clustering algorithm that builds clusters around clusters' representatives: the *Distributed Data-Aware Representative Clustering* (DARC). In addition, we evaluate our hypothesis on the homogeneity of these clusters.

The DARC algorithm promotes a "head election" among neighboring nodes. They exchange information about their most similar neighbor. Then, a node chooses as its head the most often choice among its neighbors, including its own choice. This "election" is the result of the agreement among neighboring nodes and provides a cluster's representative.

Our primary motivation to propose a novel distributed clustering algorithm is to provide a method to deal with failure issues inherent to data suppression schemes [Silberstein et al. 2007]. Since the resultant clusters are homogeneous, the data of a node that fails to deliver its message can be better estimated using the data of a non-failing node in its cluster. Moreover, our proposal can provide the support for the spatial part of a scheme for spatio-temporal data suppression. In this scheme, the head uses cluster summaries to estimate the data its cluster members suppress. Then, we need to build more homogeneous clusters.

2. Related Work

One of the first proposals to build clusters of sensor nodes in a localized fashion has been LEACH [Heinzelman et al. 2000]. The authors have proposed a simpler distributed clustering algorithm as part of a cluster-based data routing protocol. In the LEACH's clustering algorithm, each node "elects" itself as a cluster head according to a user-defined probability. The chosen nodes broadcast a message communicating their head status. A non-head node listens to the heads' messages and chooses the nearest one as its head node. After associating itself to a cluster head, the non-head node just sends data to its cluster head, which aggregates the data of its cluster's members and sends the summary to the base station. The clustering procedure of LEACH only considers the geographical proximity of the nodes in its neighborhood definition. Then, it is classified as *ordinary spatial clustering* [Reis et al 2007]. We consider the LEACH's algorithm as one of the simplest and least costly proposals for distributed clustering. Then, we use the results of the LEACH's clustering algorithm as a basis for comparison.

Our work also relates to the data-aware clustering proposals of Kotidis [2005] and Tulone & Madden [2006]. Kotidis [2005] has proposed an algorithm to select a small set of *representative nodes* (a *snapshot*) as part of a scheme of spatial data suppression to answer queries. Nodes monitor their neighbors' data messages and estimate the coefficients of a linear model to predict their neighbor's data. To define their neighborhood, nodes broadcast their sensed values and listen to the broadcast of their neighbors. Using the estimated model, a node predicts their neighbors' data and compares them with the received data. If the predicted and the real data of a neighbor differ by less than a threshold θ_{SNAP}, this neighbor enters to the candidates list of the node. After completing their candidates lists, nodes broadcast them and listen to their neighbors' lists. A node chooses as its representative the neighbor node with the longest candidates list. Once the representative nodes are chosen, only they answer the queries. In the snapshot maintenance, non-representative nodes continuously monitor their representatives' data. Whenever they differ from the data the non-representative predicts, the node looks for another representative, repeating the initial steps.

Representative nodes also appears in the spatial version of PAQ, a scheme for temporal data suppression using time series models [Tulone and Madden 2006]. PAQ's algorithm simplifies the Kotidis' proposal. It evaluates the similarity of two nodes' data only comparing their difference to a similarity threshold θ_{PAQ}. Furthermore, nodes do not exchange their list of similar neighbors to choose their representatives. Once the node has its list of similar neighbors, it includes its own ID in this list and chooses as its representative (the head) that node with the lowest ID. If a node is a head, it broadcasts a message communicating its status. A non-head receives a head message and checks if the head ID belongs to its list of heads. If so, it sends a joining request to the head node. Otherwise, it keeps listening to the heads' messages until the joining period ends. After that, if a node did not receive messages of the heads in its list, it remains alone and looks for a head in the next time period (clusters maintenance). As in the Snapshot algorithm, only the representative nodes (heads) send data to the base station.

Our DARC algorithm has been inspired in PAQ's and Kotidis' algorithms to build snapshots. We have adopted the simple evaluation of the nodes' similarity of PAQ and adapted the "neighbors' conversation" in the Kotidis' proposal. In DARC

algorithm, neighboring nodes exchange information about their most similar neighbor. Then, a node chooses as its head the most often choice among its neighbors. This transforms the "neighbors' conversation" into "neighbors voting" and the heads choice in a real "heads election". In addition, we propose an adjusting time period, which gives to the head nodes without a cluster (solitary nodes) a last chance to get a cluster at the end of the nodes association phase. To build the initial clusters, PAQ's algorithm spends up to two local messages per node and Kotidis' algorithm spends up to six messages, whereas DARC algorithm spends up to four messages per node. To maintain the clusters, DARC spends from zero to three messages per node, whereas PAQ's and Kotidis'algorithm spend up to two and six messages per node, respectively.

The goal of the representatives in Kotidis' and PAQ's proposals is to represent each associated node *individually*. To represent a single associated node, the representative does not need the data of the other associated nodes. In our proposal, the representative nodes (the heads) are *clusters' representatives*. They compute the *average* of the data of *all* nodes in the cluster and the resulting value *estimate* these nodes' data. This estimation procedure allows for spatial suppression as well as for local monitoring of the cluster area.

3. Our Proposal for Distributed Data-Aware Clustering

From now on, we reserve the term "similar neighboring nodes" or just "neighbors" to denote those geographical neighbors that collect similar data. As "geographical neighbors", we mean those nodes that can communicate to each other.

In this paper, we define the similarity of two values v_i and v_q as $d_i = | v_i - v_q |$. The value v_i is considered to be similar to v_q if $d_i \leq \theta_{(S)i}$, where $\theta_{(S)i}$ is a similarity threshold. In DARC algorithm, $\theta_{(S)i} = \theta \times s_i$, where s_i is standard deviation of the measures of the node that evaluates the similarity between its data and its neighbor's data. This definition for $\theta_{(S)i}$ allows for standardizing the difference between two sensed data. This makes easier the choice of the similarity parameter θ, since θ represents the maximum number of standard deviations that separate two similar values. For instance, we consider as similar two sensed data apart from each other at most four standard deviations, that is, θ=4. The value of s_i is estimated during a learning phase.

3.1. The Distributed Data-Aware Representative Clustering (DARC) Algorithm

The main idea of DARC algorithm is to get an agreement among neighboring nodes to choose one of them as their cluster head. This agreement results from the exchange of local messages among neighboring nodes. In the first phase, the non-head nodes sends three local messages and the head nodes have to send one more local message. In the clusters maintenance, nodes send from zero to three local messages. As in Kotidis' SNAP algorithm, the clock synchronization is necessary.

The clusters' building begins after the learning phase. The total time period for clusters building is divided into four time periods:

1) Talking time (Δ_{TT}): nodes exchange messages with their sensed values.
2) Association time (Δ_{TA}): nodes choose their heads and decide their status (head or non-head). Heads broadcast their status and all nodes listen to the messages.
3) Joining time (Δ_{TJ}): non-head nodes send join messages and heads listen to them.

4) Adjusting time (Δ_{TAD}): Head nodes without a cluster have a last opportunity to join other heads. All head nodes keep listening to the joining messages, while non-head nodes switch their radios to the sleep mode until the end of the adjusting time.

At the end of the initial clustering, nodes have one of three conditions: head, non-head or solitary node. A head node has at least one associated non-head node. A non-head belongs to one cluster and a solitary node does not belong to any cluster.

We explain our clustering proposal using the example in the Figure 1. Figure 1a presents a sensor network, the nodes' ID (inside the circles) and the value they sense (v) in a time period t. The edges represent the radio links among the nodes. The similarity threshold is θ=4 and $s_i = 1$, for i = 1,2,3,4,5,6,7,8,9.

Each node discovers its geographical neighbors, refines this neighborhood discarding those nodes with non-similar data (Figure 1b) and chooses the most similar neighbor to be its head candidate (neighbors inside the rectangles). Note the node ID_6 is within the radio range of node ID_1, but it is not part of the neighbors' list of ID_1. This is because their data are not similar, since $(v_1 - v_6) > \theta$. Nodes communicate the chosen heads to the neighbors and receive their choices. Then, the nodes build the list $\{CH^{(list)}\}$, which contains their own chosen heads and the choices of their neighbors that belong to the list $\{N_i\}$ (Figure 1c). The $\{CH^{(list)}\}$ of node ID_4, for example, has only its chosen head (ID_5), since the heads chosen by its neighbors (ID_1 and ID_8) does not belong to its neighbors list. The agreement among nodes' choices occurs when each node refines its list $\{CH^{(list)}\}$ keeping only the most "popular" node(s) among the chosen heads (Figure 1c). If there is not a most often node in the original $\{CH^{(list)}\}$, the refined $\{CH^{(list)}\}$ is the neighbors list $\{N_i\}$. If the node's ID belongs to the its refined $\{CH^{(list)}\}$, it sets its status to head, broadcasts its status and execute its tasks as a head. In Figure 1c, nodes ID_1, ID_2, ID_5, ID_8 and ID_9 set their status as heads. The non-head nodes (ID_3, ID_4, ID_6 and ID_7) listen to the heads' messages and build two lists of nodes: $\{CH^{(cand)}\}$, which has heads belonging to the refined $\{CH^{(list)}\}$ list (inside the ellipses) and $\{CH^{(wait)}\}$, which has the other neighbor heads (Figure 1d) . At the end of clustering period, non-head nodes choose the most similar head in the $CH^{(cand)}$ list. If $CH^{(cand)}$ is empty, they choose the first head in the waiting list $CH^{(wait)}$. If $CH^{(wait)}$ is empty, they become solitary nodes and keep this status until the maintenance phase.

The head nodes receive the joining requests of the non-head nodes and the announcements of neighbor heads, keeping the ID of the head nodes in a waiting list $CH^{(wait)}$. If a head node does not receive any joining request, it uses the list $CH^{(wait)}$ to join another head in its neighborhood. In the Figure 1 example, node ID_1 does not receive any joining message, since it is not chosen as head by any node. Then, in the adjusting time period, it uses its $CH^{(wait)}$ to join the node ID_5. If the $CH^{(wait)}$ is empty, the node changes its status to a solitary node and keeps this status until the maintenance phase, when it tries to join a cluster. Figure 1e presents the resulting clusters. The initial nine nodes are grouped into four clusters and there is no solitary node. The strategy of keeping a waiting list and having an adjusting time period avoids a large number of solitary nodes. If we apply the PAQ's grouping algorithm [Tulone and Madden 2006] to the network of Figure 1a, we get only one cluster and five solitary nodes. Since PAQ has not an adjusting time period, it produces a larger number of solitary nodes

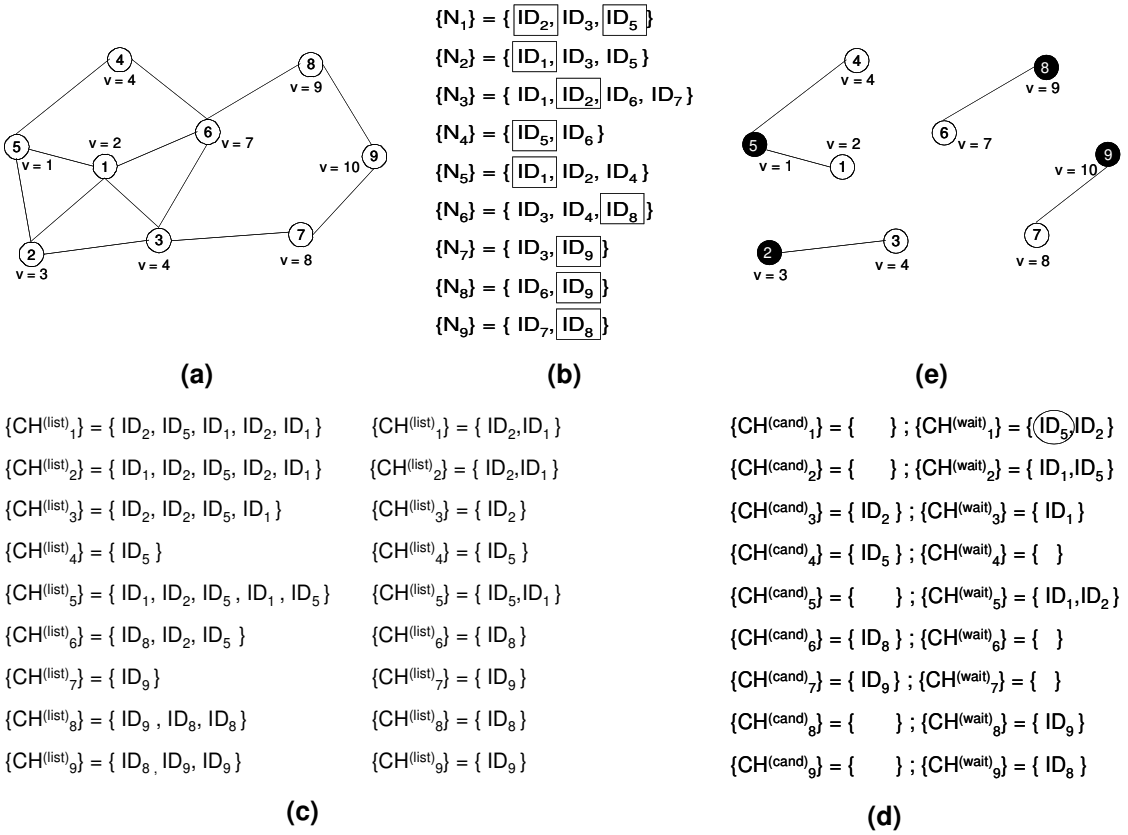

Figure 1. (a) Nodes' neighborhood and sensed data, v; (b) Neighbors lists and nearest neighbors (inside the rectangles); (c) Lists of head candidates and chosen heads (inside the ellipses); (d) Candidates and waiting lists ; (e) Final clusters (heads in black).

Once DARC algorithm builds the initial clusters, the nodes start the clusters maintenance phase. The goal of this phase is to adapt the initial clusters to data dynamics and to allow for the heads rotation. We omit its description here for briefness.

4. Assessing the Performance of the Clustering Algorithms

We have simulated datasets according to a Gaussian random field using a grid of 10000 cells (100 x 100). We refer to these data as *original data*. These datasets are characterized by *zones*, which are groups of cells with similar values. The zones' size relates to the spatial autocorrelation and is controlled by a scale parameter. The higher scale value, the larger the zones size. To each value of the scale parameter (5, 10, 15, 20, 30 and 40), we have simulated 1000 spatial datasets with the same mean (100) and variance (10). To sample the original data, we have deployed a geosensor network with 100 nodes in a regular fashion. The data collected by a sensor is the value of the cell in which it was placed. We refer to this sample as *geosensor data*.

The nodes have been grouped according to four clustering algorithms: our proposal (DARC), Kotidis' proposal (SNAP), PAQ and LEACH (for comparison). We have set the radio range equal to 20, which represents the double of the distance between two adjacent nodes in the regular grid. The idea is to localize the nodes communication to save energy. Nodes could transmit using low power during the cluster

building phase to get clusters with closer neighbors. Then, they save the entire radio range for an emergency, such as a long time period without communication with local neighbors.

The data-aware clustering proposals we consider here define different similarity measures to constrain the geographical neighborhood of the nodes. To make these algorithms comparable, we have adjusted their similarity thresholds. Since DARC uses a statistical property (the standard deviation) of the nodes' data to define their similarity threshold $\theta_{(S)i}$, we have adopted an equivalent procedure to define the similarity thresholds of the other proposals. For PAQ, we have set $\theta_{(PAQ)i} = \theta_{(S)i}$. The similarity measure of SNAP is the prediction error of a regression model. Then, we have set $\theta_{(SNAP)iq} = se_i$, where se_i is the standard error for an individual prediction using the linear regression model that node i have estimated to predict the data of node q, considering a 95% confidence level. The expression for se_i depends on s_i and is easily found in texts about linear regression models. To estimate s_i and to build the SNAP's correlation models, we have run a learning phase of 100 time periods.

For each clustering algorithm, we have calculated the number of resulting clusters (n_C) and the number of solitary nodes (n_S). To each cluster k, we have calculated the cluster size (n_k), the *Median Absolute Relative Error* of the cluster average ($MARE_k$) and the *Coefficient of Variation* (CV_k), which are defined by the expressions

$$MARE_k = \underset{i=1,\ldots,n_k}{median}\left(\frac{|CM_k - v_{ik}|}{v_{ik}}\right) \text{ and } CV_k = \frac{Sd_k}{CM_k}$$

, where v_{ik} is the data sensed by the node i of the cluster k ; CM_k and Sd_k are the average and the standard deviation of the data sensed by the members of the cluster k, respectively. $MARE_k$ measures the prediction error of cluster average CM_k as an estimate for the data of the cluster k members, whereas CV_k measures the internal homogeneity of the cluster k.

4.1. The results

For each spatial scale and given a clustering proposal, we have calculated the median of the $MARE_k$ values over the 1000 simulated datasets. A similar procedure was adopted to summarize the CV_k values. Figure 2 presents the summaries.

As expected, the values for MARE and CV of all clustering proposals decrease as the spatial correlation (zones size) gets higher.

Comparing the performance statistics, only the DARC algorithm has improved the clusters internal homogeneity and the prediction error of the cluster average if compared with the LEACH's ordinary clustering.

The SNAP algorithm has produced the smallest number of solitary nodes as well as the smallest clusters (2 and 2, in median, respectively). On the other extreme, PAQ has left the largest number of nodes without a cluster (19, in median) and built the largest clusters (6 nodes, in median). In the middle, DARC has built clusters with 4 nodes, leaving 3 nodes without a cluster (median values). Considering the network configuration and the radio range we have adopted, the maximum number of nodes in a cluster is 13. Having few but large clusters is important to save energy, since it minimizes the number of the nodes with the most costly tasks: the heads and the solitary

nodes, in this order. Whereas PAQ has built few clusters (13, in median) but left many nodes alone, SNAP has clustered almost all nodes (36 clusters, in median) at the cost of having set many heads. The DARC algorithm has built 26 (in median), which we consider to be a trade-off between the number of clusters and their size.

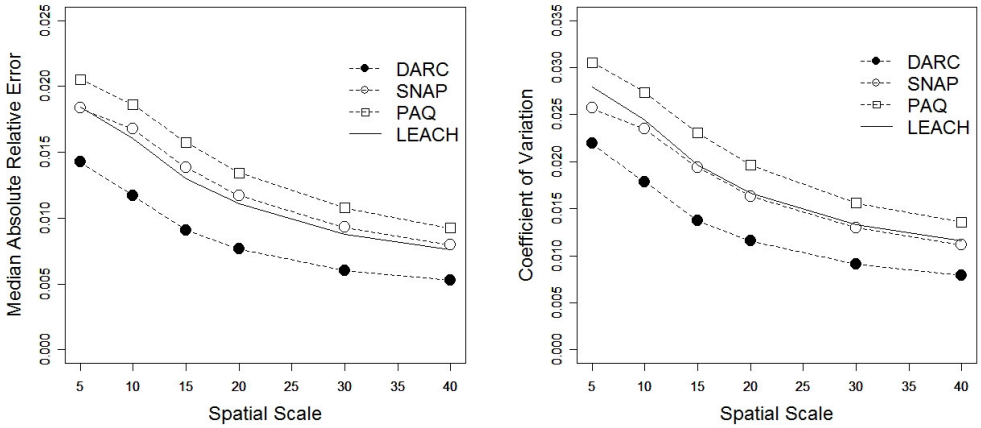

Figure 2. Performance statistics: MARE values and CV values.

5. Final Remarks

DARC's clusters have had the largest internal homogeneity and have produced averages with the smallest prediction errors. Furthermore, DARC has shown it can cluster almost all nodes without overloading an excessive number of them with the head tasks. About the clustering costs, DARC builds the initial clusters spending from three to four local messages. To maintain the clusters, nodes spend from zero to three local messages. In the SNAP algorithm [Kotidis 2005], for instance, the nodes must monitor their neighbors' data to estimate the regression models during clusters building and maintenance, besides they spend up to six messages to build the clusters.

DARC algorithm can be adapted to be part of a cluster-based data routing protocol, building data-aware clusters instead of ordinary clusters as in LEACH, for example.

References

Heinzelman, W. B., A. Chandrakasan and H. Balakrishnan (2000). "Energy-Efficient Communication Protocol for Wireless Microsensor Networks." 33rd. Hawaii International Conference on System Science.

Kotidis, Y. (2005). "Snapshot Queries: Towards Data-Centric Sensor Networks". 21st International Conference on Data Engineering (ICDE'05)

Reis, I. A., G. Câmara, R. M. Assunção, et al. (2007). "Data-Aware Clustering for Geosensor Networks Data Collection". XIII Brazilian Symposium of Remote Sensing (SBSR).

Silberstein, A., G. Puggioni, A. Gelfand, K. Munagala, and J. Yang, "Suppression and Failures in Sensor Networks: A Bayesian Approach", in Very Large DataBases '07, 2007.

Tulone, D. and S. Madden (2006). "PAQ: Time Series Forecasting For Approximate Query Answering In Sensor Networks." Lecture Notes in Computer Science (3868): 21--37.