Towards Distributed Clustering of Streaming Sensors

Pedro Pereira Rodrigues[1,2], João Gama[1,3], and Luís Lopes[2,4]

[1] LIAAD - INESC Porto L.A.
Rua de Ceuta, 118 - 6 andar, 4050-190 Porto, Portugal
[2] Faculty of Sciences of the University of Porto, Portugal
[3] Faculty of Economics of the University of Porto, Portugal
[4] CRACS - INESC Porto L.A.
pprodrigues@fc.up.pt, jgama@fep.up.pt, lblopes@dcc.fc.up.pt

Abstract. Clustering streaming sensors is the task of clustering streaming data series produced by sensors on a wide sensor network. Considering the dynamic behavior usually enclosed in streaming data, clustering streaming sensors should be addressed as an online and incremental procedure, in order to enable faster adaptation to new concepts and produce better models through time. However, centralized clustering strategies tend to be inapplicable as usual techniques have quadratic complexity on the number of sensors, and sensor networks grow unbounded. Thus, distributed clustering techniques which operate on sensor networks must be developed. In this paper we try to clarify why clustering of streaming sensors should be addressed in a distributed fashion, its requirements and their implications for future developments. We also propose a general setup for this procedure and illustrate it with a toy example.

Keywords: Distributed clustering, distributed data streams, sensor networks.

1 Introduction

Data flows continuously from streams at high speed, producing examples over time, which would make a traditional data gathering process create databases with tendentiously infinite length. Traditional database management systems are not designed to directly support the continuous queries required by these applications. Also, data gathering and analysis have become ubiquitous, in the sense that our world is evolving into a setting where all devices, as small as they may be, will be able to include sensing and processing ability. Thus, if data is to be gathered centrally, this scenario also points out to databases with tendentiously infinite width. The application of streaming procedures to centralized databases becomes nearly inapplicable, since usual clustering strategies have quadratic complexity in the number of variables [17]. New distributed techniques must be defined to deal with this new ubiquitous streaming setting. Clustering streaming sensors is the task of clustering streaming data series produced by

sensors on a wide sensor network. In this paper we address the research area of developing distributed algorithms to solve this problem. The main idea behind this task is the following: some (or all) of the sensors enclosed in the network should perform some kind of processing over the data gathered by themselves or/and by their neighbors, in order to achieve an up-to-date clustering structure definition of the entire sensor network.

1.1 Sensor Data and Networks

Sensors are usually small, low-cost devices capable of sensing some attribute of a physical phenomenon. These devices are most of the times interconnected in a distributed network which, due to the ubiquitous setting, creates new obstacles to the common data mining tasks. Most works on clustering analysis for sensor networks actually concentrate on clustering the sensors by their geographical position and connectivity, mainly for power management and network routing purposes [5]. However, in this work, we are interested in clustering techniques for data produced by the sensors, instead.

In terms of hardware development, the state-of-the-art in sensors is well represented by a class of multi-purpose sensor nodes called *motes* [7], which were originally developed at UC Berkeley and are being deployed and tested by several research groups and start-up companies. In most of the current applications [7], the sensor nodes are controlled by module-based operating systems such as TinyOS [1] and are programmed using arguably somewhat *ad-hoc* languages such as nesC [11]. Sensor networks are composed of a variable number of sensors (depending on the application), which have several features that put them in an entirely new class when compared to other wireless networks, namely: (a) the number of nodes is potentially very large and thus scalability is a problem, (b) the individual sensors are prone to failure given the often challenging conditions they experiment in the field, (c) the network topology changes dynamically, (d) broadcast protocols are used to route messages in the network, (e) limited power, computational, and memory capacity, and (f) lack of global identifiers [2]. Sensor network applications are, for the most part, data-centric in that they focus on gathering data about some attribute of a physical phenomenon. Routing can be based on the data-centric approach. Two main approaches are used: (a) sensors broadcast advertisements for the availability of the data and wait for interested nodes, or; (b) sinks broadcast interest in the data and wait for replies from the sensors. The queries for data are usually done using *attribute-based naming*, that is, using the attributes of the phenomenon being measured. The data is usually returned in the form of streams of simple data types without any local processing.

2 Connections with Previous Research

The main question is how can a distributed system develop and learn the global clustering structure of streaming sensors data, even though communication between sensors is limited (and even nonexistent in some extent). The handicap

on processing streams is the impossibility of total knowledge of each series data. One suitable solution to this problem are approximate algorithms. The handicap is reinforced in ubiquitous settings as, for a given processing unit, total knowledge of the complete set of sensors' data is also improbable. Hence, approximate algorithms must be considered in this direction also.

2.1 Clustering Streaming Time Series

Clustering streaming time series has been already targeted by researchers in order to cope with the tendentiously infinite amount of data produced at high speed. Beringer and Hüllermeier proposed an online version of *k-means* for clustering parallel data streams, using a Discrete Fourier Transform approximation of the original data [3]. The basic idea is that the cluster centers computed at a given time are the initial cluster centers for the next iteration of *k-means*, applying a procedure to dynamically update the optimal number of clusters at each iteration. Clustering On Demand *(COD)* is another framework for clustering streaming series which performs one data scan for online statistics collection and has compact multi-resolution approximations, designed to address the time and the space constraints in a data stream environment [8]. It is divided in two phases: a first online maintenance phase providing an efficient algorithm to maintain summary hierarchies of the data streams and retrieve approximations of the sub-streams; and an offline clustering phase to define clustering structures of multiple streams with adaptive window sizes. Rodrigues et al. [18], proposed the Online Divisive-Agglomerative Clustering *(ODAC)* system, a hierarchical procedure which dynamically expands and contracts clusters based on their diameters. It constructs a tree-like hierarchy of clusters of streams, using a top-down strategy based on the correlation between streams. The system also possesses an agglomerative phase to enhance a dynamic behavior capable of structural change detection. The splitting and agglomerative operators are based on the diameters of existing clusters and supported by a significance level given by the Hoeffding bound [12]. However, if data is produced by sensors on a wide network, the proposed algorithms tend to deal with them as a centralized multivariate stream.

2.2 Distributed Clustering

Although few works were directly targeted at data clustering on sensor networks, some distributed techniques are obvious starters of this area of research. Distributed implementations of well-known algorithms may produce both valuable and impractical systems, so the path to them should be carefully inspected. Sensors' characteristics imply resource restrictions which narrow the possibilities for high-load computation while operating under a limited bandwidth. Resource-Aware Clustering [9] was already proposed as a stream clustering algorithm for example clustering that can adapt to the changing availability of different resources. However, although sensor networks usually operate with limited bandwidth, due to energy restrictions, the amount of data produced by these networks

can become unbounded due to the large number of sensors and their fast sensing abilities. This can turn out to be an important bottleneck and force some nodes to spend more energy on relaying information to the sink [16]. The key objective of sensor data processing is to maintain information incrementally, in such a way that the system can cope with high-speed production of data. Data stream mining on sensor networks needs to operate under a limited bandwidth, reducing the capability to represent and transmit the data mining models over the network [14], which creates an even thicker barrier to an efficient handling of the continuous flow of data. Furthermore, the detection and reaction to changes in the clustering structure must be adapted to the new distributed setting. While it may seem straightforward to adapt previously developed techniques [18], there is another change that must be monitored, with even more control: network topology changes. The network topology can be highly volatile, evolving with time due to, for example, sensor movement, broken links or sensor failures. On top of all these issues, the deployment of moving sensors is an emergent technique, used in numerous applications. In these contexts, requirements for distributed clustering systems become extreme.

On one side, the data clustering structure could be defined locally, possibly restricted by the network topology, in order to confine communications to nearby sensors. Afterwards, these local structures would be combined by top-level processing units to define a global clustering structure, as cluster ensembles [20] techniques would operate. On the other hand, sensors could be able to define representative data or summary information that would be used by any top-level process to define a single clustering structure, even if roughly approximated. Kargupta et al. presented a collective principal component analysis (PCA), and its application to distributed example cluster analysis [13]. However, these techniques still consider a centralized process to define the clusters, which could become overloaded if sensors were required to react to the definition of clusters, forcing the server to communicate with all sensors. Given the extent of common sensor networks, the old client-server model is essentially useless to help the process of clustering data streams produced on sensors.

3 Issues on Clustering Streaming Sensors

Common sensor networks data aggregation techniques are based on the Euclidean distance (physical proximity) of sensors to perform summaries on a given neighborhood [5]. However, the clustering structure definition of the series of data produced by the sensors is orthogonal to the physical topology of the network, as stressed in the example presented in Figure 1. Considering the main restrictions of sensor networks, the analysis of clusters of multiple sensor streams should comply not only with the requirements for clustering multiple streaming series [17], but also with the available resources and setting of the corresponding sensor network. New techniques to efficiently perform clustering of streaming sensors should be developed in distributed fashions, as massive sensor networks produce high levels of data processing and transmission, reducing the ability to

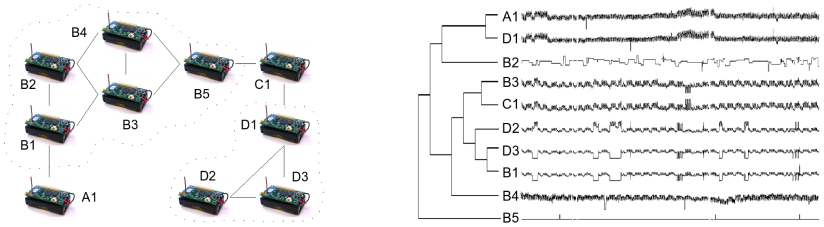

Fig. 1. Example of a mote sensor network (left) and a possible clustering definition of the series produced by each sensor (right). This example shows the orthogonality that is expected to exist between network topology and the sensors' data structure.

feedback the information to the system. We foresee advantages in this distributed approach: local processing reduces dimensionality of clustering techniques, while implying less communication to the server; exploiting relations between sensed data clusters and actual geographical clusters can provide deeper understanding of the network dynamics; faster answers, sensitive information preservation, easier deployment, possibility of self-organization and better adaptability. These are strong features for applications where centralized versions are inapplicable.

3.1 Increased Privacy Preservation

The privacy of personal data is most of the times important to preserve, even when the objective is to analyze and compare with other people's data. Anonymization is the most common procedure to ensure this but experience as shown that it is not flawless. This way, centralizing all information in a common data server could represent a more vulnerable setup for security breaches. If we can achieve the same goal without centralizing the information, privacy should be easier to preserve. Furthermore, the system could achieve a global clustering structure without sharing sensible information between all nodes in the network.

3.2 Sensor Network Message Forwarding

One of the highest resources consuming tasks in sensor networks is communication. For example, a network of wireless integrated network sensors (WINS) has to support large numbers of sensors in a local area with short range and low average bit-rate communication [16]. If a distributed clustering procedure is applied at each forwarding node, usual data aggregation techniques could be data-centric, in the sense that one node could decide not to transmit a message, or aggregate it with others, if it contains information which is quite similar to other nodes'.

3.3 Sensor Network Deployment Quality

When sensor networks are deployed in objective areas, the design of this deployment is most of the times subject to expert-based analysis or template-based

configuration. Unfortunately, the best deployment configuration is sometimes hard to find. Applying distributed clustering of sensors' data streams the system can identify sensors with similar reading profiles, while investigating if the sensors are in the same geographical cluster. If similar sensors, with respect to the produced data, are place in a dense, with respect to the geographical position, cluster of sensors, resources are being spoiled as less sensors would give the same information. These sensors could then be assigned to different positions in the network.

3.4 Domains of Application

Sensor networks are nowadays used in various applications for sensing, processing and monitoring physical measures or activities. Clustering the sensors by the data they produce can present specific advantages for some of them. In electricity supply systems, the identification of demand profiles (ex: industrial or urban) by clustering streaming sensors' data decreases the computational cost of predicting each individual sub-network load [10]. As thousands of sensors are naturally distributed in the electrical network, distributed procedures which would focus on local networks could prevent the dimensionality drawback. Also, a common problem in geoscience research is the monitoring of natural phenomena evolution. Sensor nodes can be densely deployed either very close or directly inside the phenomenon to be observed [21]. Clustering the data produced by different sensors is helpful to identify areas with similar profiles, possibly indicating actual water or wind streams. The Global Positioning System (GPS) is commonly used to monitor location, speed and direction of both people and objects. Identifying similar paths, for example, in delivery teams or traffic flow, is a relevant task to current enterprises and end-users [15]. However, the amount of data produced by each GPS receiver is so huge, and the allowed reply delay so narrow, that performing centralized clustering of GPS tracks is too expensive to perform. Finally, clustering medical sensor data (such as ECG, EEG, etc.) is useful to determine association between signals [19], allowing better diagnosis. Detecting similar profiles in these measures among different patients is one way to explore uncommon conditions. Mobile and embedded devices could interconnect different patients and physicians, without revealing sensitive information from patients while nevertheless achieving the goal of identifying similar profiles.

4 Distributed Clustering of Streaming Sensors

The main objective of a clustering system should be to be able to answer queries for the global clustering definition of the entire sensor network. If sensors are distributed on a wide area, with local sites being accessible from transient devices, queries could be issued at each local site, enabling fast answers to be sent to the querying device. However, current setups assume data is forwarded into a central server, where it is processed, being this server the main answering device. This setup forces not only the data but also the queries to be transmitted across the

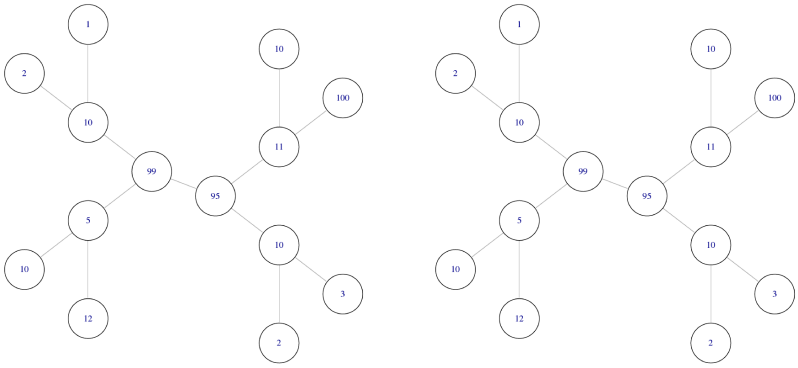

Fig. 2. Toy example of a sensor network, where each sensor s produces a stream $X_s \sim N(\mu_s, 0.5)$, link connections are represented by edges and vertex labels on the left plot indicate each sensor's concept (μ_s), while vertex labels on the right plot indicate a possible mean and standard deviation of actual data produced in such a conceptual network.

network into a sink. In this section we propose a general setup for distributed clustering of streaming sensors, taking into account the requirements and advantages previously enunciated. First, we should be able to have some type of sketch of the stream being produced by each sensor, in order to reduce the computation of similarities between sensors. Then, we believe each sensor should communicate only locally with its neighbors, in order to reduce the amount of data being forwarded throughout the network. Finally, we must include mechanisms to prevent redundant communication, while monitoring for structural changes in the clustering definition of the entire network. Throughout the entire section we will use a toy example to illustrate our proposal. Figure 2 presents that example of a sensor network, where each sensor s produces a stream of data $X_s \sim N(\mu_s, 0.5)$, with edges representing link connections and where vertex labels are the μ_s. In this toy example, we are only looking for a definition of K cluster centers, with $K = 2$ previously known by the system, assuming dissimilarity between sensors is computed by the distance between the means of each sensor. Although this simple example lacks some of the common characteristics of real-world scenarios, its extension is straightforward. Nevertheless, a sound definition and evaluation of a complete system is ongoing and should be presented in future work.

4.1 Sketching Streaming Sensors

Each sensor produces a stream of data, usually defined by one or more infinite time series. In the following, we consider that each sensor produces a single stream of data. As previously stated, we want to define a clustering structure for the sensors, where sensors producing streams which are alike are clustered together. Hence, we shall define a similarity measure for such streams. However, we do not ever have access to the complete time series, and we would like to

prevent the whole data to be transmitted in the network. We should consider approximate metrics, using simple sufficient statistics of the streams, or data synopsis, to achieve similar results. One way to summarize a data stream X_i is by computing its sample mean $\bar{x}_i$ and standard deviation s_i, assuming some kind of data distribution. This simple strategy, although extremely naive, is sufficient for our toy example, where we assume the dissimilarity between sensors to be the distance between their sample means. As we can see in Figure 2, and although in several real-world scenarios that is not true, we should assume the sample mean of each sensor to be non-correlated with its physical location. Each sensor produces data continuously. Given this, each sensor s is responsible to keep its own estimate of the sample mean $\hat{\mu}_s$ in a online fashion. In this small example we will consider that, at each new example x_s^t arriving at time t, each sensor's estimate is updated by doing

$$\hat{\mu}_s^t = \hat{\mu}_s^{t-1}(1 - \lambda_\mu) + x_s^t \lambda_\mu \tag{1}$$

where $\lambda_\mu \in [0, 1]$ is the update coefficient for the online computation of the sensor mean (a simply incremental version would consider $\lambda_\mu = \frac{1}{t}$). More complex strategies could include distribution distances based on the histograms of each sensor's data (e.g. relative entropy [4]), where each sensor would have to transmit the frequency of each data interval to its neighbors, or using approximations of the original data [3]. Overall, we should consider techniques that project each sensor's data stream into a reduced set of dimensions which suffice to extract similarity with other sensors. These estimates can be seen as the sensor's current overview of its own data, giving an indication of where in the data-space this sensor is included.

4.2 Local Approximations of the Global Clustering Definition

As each sensor s is able to sketch its own data in a dimensionally-reduced definition, it is also able to interact with its neighbor nodes η_s, in order to assess a local clustering of sensors. In our example, the distance between two sensors s_i and s_j can be defined as $d_{ij} = |\hat{\mu}_i - \hat{\mu}_j|$. Overall, each sensor should include incremental clustering techniques which operate with distance metrics developed for the dimensionally-reduced sketches of the data streams. Our goal is to have at each local site a global clustering structure of the entire sensor network. To achieve this, at each time t, each sensor s should send to its neighbors η_s its own estimate of the global clustering C_s^t, instead of sendind only its own sketch $\hat{\mu}_s^t$. Note that with this approach, each sensor keeps an approximate estimate of the global cluster centers C_s. This estimate can be seen as the sensor's current overview of the entire network which, together with its own sketch, gives an indication of where in the entire network data-space this sensor is included.

At a given point in time t, a given sensor s receives estimates of the global cluster centers which were gathered by their neighbors: $P_s^t = \{C_i^t \mid i \in \eta_s\}$. The idea behind this step is to aggregate all the locally defined centers and apply a clustering procedure on these centers, considering them as points for

the final clustering. This way, C_s^{t-1} are defined as initial centers for the iterative clustering procedure. Merging clustering definitions is a known technique which as been argued to give good results [6]. However, given the streaming setting where sensor networks usually operate, these procedures must also be executed online. If no information is sent by neighbors, sensor s would have an *almost static* view of the global network clustering. The only update that is possible in this scenario is to incrementally update the clustering estimate C_s with current sensor's sketch. This would be supported by two steps: defining to which cluster center $c_j \in C_s$ sensor s is assigned, $j = \text{argmin}_{i \in \{1,\dots,K\}} d(c_i, \hat{\mu}_s^t)$; and updating c_j using simple techniques such as $c_j^t = c_j^{t-1}(1 - \lambda_c) + \hat{\mu}_s^t \lambda_c$, where $\lambda_c \in [0, 1]$ is the update coefficient for the online adjustment of the cluster center. This way, next time this sensor transmits its estimate of the global clustering structure, it is already updated with its most recent sketch.

4.3 Communication Management

Communication is one of the most resource-consuming procedures of sensor networks [5]. If a central server is used to aggregate all data, each individual sketch must be forward through the network into a sink node. To enable each local site to have the global clustering structure of the entire network, the central server would have to reply with K values, largely increasing the global number of transmissions. If we transmit data only between neighbors, this would represent $2E$ communications of K values, where E is the number of links in the network, achieving an approximate clustering of the whole network at each node, with much less communication. On top of this, if the concept of the data being produced in the network is stable, then the clustering estimates will converge, and transmissions will become redundant. We should include mechanisms to allow each sensor to decide to which neighbors it is still valuable to send information. In our toy example, sensor s should transmit its estimate C_s to sensor $a \in \eta_s$ iff $D(C_s, C_a) > \tau$, where D is the Euclidean distance between ordered centers and τ is a suitably defined (possibly adaptive) threshold. However, the world is not static. It is possible that, with time, the sketches of each sensor will change, adapting to new concepts of data. On a long run, the communication management strategy could prevent the system from adapting to new data. Overall, sensors should include change detection mechanisms that would trigger if the data changes, either univariatedly at each sensor, or in the global interaction of sensor data.

4.4 Results on Toy Example

In order to see how the previous techniques would apply to a given problem, we used the scenario presented in Figure 2 and generated 20 different sets of streams of data accordingly. Each sensor's sketch is maintained as in Equation 1 with $\lambda_\mu = 1/t$ and communication is allowed after 10 examples (to stabilize the sample means) as described in Section 4.2, using a step of *k-means* as incremental

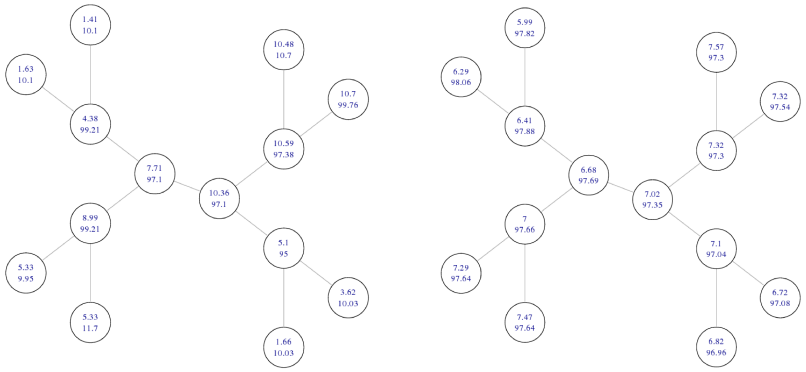

Fig. 3. Results of one run of the toy scenario. Plot shows the cluster centers estimates after 1 iteration (left) and 150 iterations (right). The local update of global clustering definition allows all nodes to achieve an acceptable result.

clustering of gathered centers. First, each sensor s sends its own sketch to all neighbors η_s, and computes the first cluster centers using a iteration over its neighbors' sketches. The following communications are each sensor's estimate of the hereby gathered global clustering centers. After 150 points, results are evaluated. Since we are forcing the concept to be stable, the system also stabilizes quite rapidly. We compare each local estimate with the real centers gathered using each sensor's real μ_s, in terms of Euclidean distance of the ordered centers. The objective (correct) cluster centers are $C = \langle 6.9, 98.0 \rangle$. Over all runs, the median average deviance from the actual centers was $D = 0.205$ for a centralized online k-means on the whole data, $D = 0.663$ for the proposed local system without any transmission control and $D = 0.758$ for the proposed system with a simple transmission control based on the distance between clustering definitions.

Figure 3 presents the resulting definition for one run, after 150 iterations, deviating from the actual centers by $D = 0.71 \pm 0.23$, using an average of 17 transmissions of 2 values per iteration. Although at first cluster centers estimates at each sensor are highly biased towards local data, the exchange of information between sensors allows the system to reduce the error very rapidly, so that, after only few iterations, the system is already stable. We should note that, without restricting transmission, the resulting structure is a bit better ($D = 0.66 \pm 0.26$) performing exactly 26 transmissions of 2 values per iteration. To have an idea about the quality of this results, a centralized online *k-means* clustering on all data would end-up with $D = 0.31$. The main idea is to have an estimate of the centers in all sensors. Considering the possible size of sensor networks, we believe that these techniques, although approximate, are much more feasible than centralizing all information. The path will start from here finding better analysis and procedures to achieve good results in real-world sensor networks, with tight error bounds.

5 Future Developments

Although the physical topology of the network may be useful for data management purposes, the main focus should be on finding similar sensors irrespectively to their physical location. Also, minimizing different resources (mainly energy) consumption is a major requirement in order to achieve high uptimes for sensors. On top of this, a compact representation of both the data and the generated models must be considered, enabling fast and efficient transmission and access from mobile and embedded devices. Even though processing may be concentrated on local computations and short-range communication, the final goal is to infer a global clustering structure of all relevant sensors. Hence, approximate algorithms should be considered to prevent global data transmission. Given this, when querying a given sensor for the global clustering, we allow (and known beforehand that we will have) an approximate result within a maximum possible error with a certain probability. Each approximation step (local sketch, local clustering update, merging different cluster definitions, etc.) should be restricted by some stability bound on the error [12]. These bounds should serve as balancing deciders in the trade-off between transmission management and resulting errors. In this paper we have shown how the task of clustering streaming sensors differs from previously studied traditional tasks in its neighborhood. We have tried to clarify to what extent the previous approaches are not suitable for the new task of clustering streaming sensors, pointing out the main restrictions implied by sensor networks characteristics. A general setup is proposed for distributed clustering of streaming sensors, and a toy example is used to illustrate the procedure. Future work will focus on clear algorithms for real-world applications of sensor networks, and their strict evaluation on ubiquitous scenarios. A first step is being already done using online histograms as sketching process, and distance between distributions as dissimilarity measure among sensors.

Acknowledgments The work of P.P. Rodrigues is supported by the Portuguese Foundation for Science and Technology (FCT) under the PhD Grant SFRH/BD/29219/2006. The authors thank FCT's Plurianual financial support attributed to LIAAD and CRACS and the participation of Project ALES II under FCT's Contract POSC/EIA/55340/2004 and Project CALLAS under FCT's Contract PTDC/EIA/71462/2006.

References

1. The TinyOS Documentation Project. Available at `http://www.tinyos.org`.
2. I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey on Sensor Networks. *IEEE Communications Magazine*, 40(8):102–114, 2002.
3. J. Beringer and E. Hüllermeier. Online clustering of parallel data streams. *Data and Knowledge Engineering*, 58(2):180–204, August 2006.
4. M. Berthold and D. Hand. *Intelligent Data Analysis - An Introduction*. Springer Verlag, 1999.

5. H. Chan, M. Luk, and A. Perrig. Using clustering information for sensor network localization. In *First IEEE International Conference on Distributed Computing in Sensor Systems*, pages 109–125, 2005.
6. G. Cormode, S. Muthukrishnan, and W. Zhuang. Conquering the divide: Continuous clustering of distributed data streams. In *Proceedings of the 23nd International Conference on Data Engineering (ICDE 2007)*, pages 1036–1045, 2007.
7. D. E. Culler and H. Mulder. Smart Sensors to Network the World. *Scientific American*, 2004.
8. B.-R. Dai, J.-W. Huang, M.-Y. Yeh, and M.-S. Chen. Adaptive clustering for multiple evolving streams. *IEEE Transactions on Knowledge and Data Engineering*, 18(9):1166–1180, September 2006.
9. M. M. Gaber and P. S. Yu. A framework for resource-aware knowledge discovery in data streams: a holistic approach with its application to clustering. In *Proceedings of the ACM Symposium on Applied Computing*, pages 649–656, 2006.
10. J. Gama and P. P. Rodrigues. Stream-based electricity load forecast. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2007)*, volume 4702 of *Lecture Notes in Artificial Intelligence*, pages 446–453, Warsaw, Poland, September 2007. Springer Verlag.
11. D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesC Language: A Holistic Approach to Network Embedded Systems. In *PLDI'03*, pages 1–11. ACM Press, 2003.
12. W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.
13. H. Kargupta, W. Huang, K. Sivakumar, and E. L. Johnson. Distributed clustering using collective principal component analysis. *Knowledge and Information Systems*, 3(4):422–448, 2001.
14. H. Kargupta, B.-H. Park, S. Pittie, L. Liu, D. Kushraj, and K. Sarkar. MobiMine: Monitoring the stock market from a PDA. *SIGKDD Explorations*, 3(2):37–46, 2002.
15. A. Moreira and M. Y. Santos. Enhancing a user context by real-time clustering mobile trajectories. In *International Conference on Information Technology: Coding and Computing (ITCC'05)*, volume 2, page 836, Los Alamitos, CA, USA, 2005. IEEE Computer Society.
16. G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. *Communications of the ACM*, 43(5):51–58, May 2000.
17. P. P. Rodrigues and J. Gama. Clustering techniques in sensor networks. In *Learning from Data Streams - Processing Techniques in Sensor Networks*, pages 125–142. Springer Verlag, 2007. Chapter 9.
18. P. P. Rodrigues, J. Gama, and J. P. Pedroso. Hierarchical clustering of time-series data streams. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):615–627, May 2008.
19. D. M. Sherrill, M. L. Moy, J. J. Reilly, and P. Bonato. Using hierarchical clustering methods to classify motor activities of copd patients from wearable sensor data. *Journal of Neuroengineering and Rehabilitation*, 2(16), 2005.
20. A. Strehl and J. Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, December 2002.
21. J.-Z. Sun and J. Sauvola. Towards advanced modeling techniques for wireless sensor networks. In *Proceedings of the 1st International Symposium on Pervasive Computing and Applications*, pages 133–138. IEEE Computer Society, 2006.