1

Distributed Reasoning

Pedro Rodrigues[1] and João Gama[1,2]

[1] LIAAD-INESC TEC, University of Porto
[2] Faculty of Economics, University Porto
pprodrigues@med.up.pt; jgama@fep.up.pt

Summary. This paper discusses the problem of learning a global model from local information. We consider ubiquitous streaming data sources, such as sensor networks, and discuss efficient learning distributed algorithms. We present the generic framework of distributed sources of data, an illustrative algorithm to monitor the global state of the network using limited communication between peers, and an efficient distributed clustering algorithm.

1.1 Introduction

Data are distributed in nature. Nowadays, detailed data for almost any task are collected over a broad area, and streams in at a much greater rate than ever before. In particular, advances in miniaturization, the advent of widely available and cheap computer power, and the explosion of networks of all kinds gave life to inanimate things. Simple objects that surround us are gaining sensors, computational power, and actuators, and are changing from static, inanimate objects into adaptive, reactive systems. Sensor networks and digital social networks are present everywhere (Kargupta et al., 2004).

Examples of network data include smart grids consisting of millions of automated electronic meters. The meters would generate an overwhelming amount of distributed data that can be handled with emergent techniques: data streams management and processing approaches. A key characteristic of smart grids is the *intelligent layer* that analysis the data produced by these meters allowing companies to develop powerful new capabilities in terms of grid management, planning and customer services for energy efficiency. The development of the market with a growing share of load management incentives and the increasing number of local generators will bring new difficulties to grid management and exploitation. Present monitoring systems suffer for the lack of machine learning technologies that can modify the behavior of monitoring systems based on the sequence patterns arriving over time. From a data mining point of view, a smart grid forms a network (eventually decomposable) of distributed sources of high-speed data streams. The dynamics of

data are unknown; the topology of network changes over time, the number of meters tends to increase and the context where the meter acts evolves over time. This way, several data mining tasks are involved: prediction, cluster analysis (profiling), event and anomaly detection, correlation analysis, etc. All these characteristics constitute challenges and opportunities for applied research in distributed data mining. The requirements of near real-time analysis for multiple time horizons and multiple space aggregations make these analyses an even harder research challenge.

In this work we focus on *clustering*, one of the most used data mining techniques. The goal in cluster analysis is the assignment of a set of observations (or objects) into groups so that observations in the same group are similar in some sense.

The paper is organized as follow. In Section 1.2 we present the distributed network framework and an illustrative example about distributed reasoning. In Section 1.3, we present a distributed clustering algorithm for sensor networks. In the context of this work, a cluster is defined to be a set of sensors. The key characteristic of the proposed algorithm is that each sensor processes locally their own data, and communicate with neighbours in order to learn a global view of the network. The last section concludes the paper by presenting the lessons learned.

1.2 Network Data Model

The goal of our study are networks of interconnected nodes. Nodes, or sensors or peers, are sensing the environment measuring some quantity of interest. Individually, each peer has a local and limited information about the environment. If sensors communicate, the network might have a global perspective of the environment. Figure 1.1 illustrates this context.

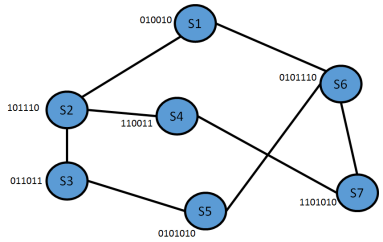

Fig. 1.1. A network of interconnected nodes. Circles represent sensors, edges represent communication paths.

1.2.1 The framework

Network topology is the organizational hierarchy of the interconnected nodes. Different network topologies can affect throughput, but reliability is often more critical.

A common structure is the *star network*, where all nodes are connected to a special central node, the coordinator. This is the typical layout found in a wireless sensor networks. Another popular layout is the *mesh network*, where each node is connected to an arbitrary number of neighbours in such a way that there is at least one traversal from any node to any other. The main purpose of a *mesh network* is fault tolerance.

Routing is the process of selecting network paths to carry network traffic. Some popular routing schemes are: *unicast*: delivers a message to a single specific node; *broadcast*: delivers a message to all nodes in the network; *anycast*: delivers a message to a group of nodes, typically the ones nearest to the source.

In data-mining problems, a user runs queries over the data produced by the sensors. A query is defined over the data produced by all the sensors:

$$Query = Q(\bigcup_{i=0}^{n} S_i)$$

We can consider two types of queries:

1. One-shot queries: What is the current state of the network?
2. Continuous queries: Track and monitor the state of network at any time.

Continuous queries are of particular interest because they are used for monitoring purposes, understanding dynamics, detect anomalies and changes.

In the network data model, data is vertically distributed. Answering continuous queries, requires specific characteristics of the algorithms. Following Du et al. (2005); Zhu, Setia, and Jajodia (Zhu et al.), the requirements for processing continuous queries are:

- Single pass: process each observation once;
- Small space: constant space;
- Small processing time;
- Reduced communications.

Local approaches are the most efficient ones (Giannella et al., 2004). They preserve privacy and security issues but require some sort of synchronization between peers (May and Saitta, 2010).

1.2.2 An illustrative example

In this section, we present an illustrative application of ubiquitous reasoning. The problem consists of monitoring data produced in a sensor network. The

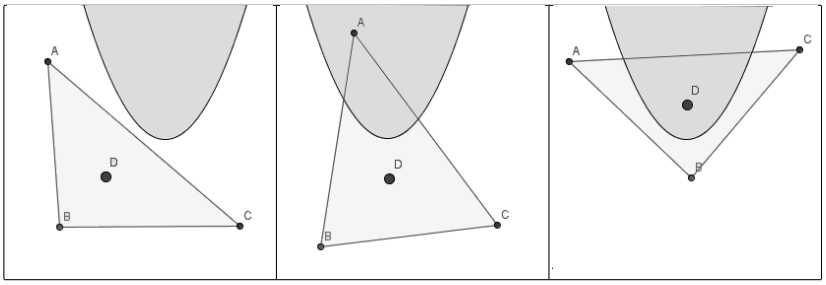

Fig. 1.2. The vector space: the gray dots (A,B,C) corresponds to the sensor's measurements; and the black dot (D) to the aggregation vector. The gray region corresponds to the alarm region. The left and central figures illustrates a normal air condition. The right figure presents an alarm condition, where none of the sensors is inside the alarm region.

sensors monitor the concentration of air pollutants. Each sensor maintains a data vector with measurements of the concentration of various pollutants (CO_2, SO_2, O_3, etc.). A function on the average of the data vectors determines the Air Quality Index (AQI). The goal consists of trigger an alert whenever the AQI exceeds a given threshold. The problem involves computing a function over the data collected in all sensors. A trivial solution consists of sending data to a central node. This might be problematic due to huge volume of data collected in each sensor and the large number of sensors.

Sharfman, Schuster, and Keren (2007) present a distributed algorithm to solve this type of problems. They present a geometric interpretation of the problem. Figure 1.2 illustrate the instance space. Each axis corresponds to one pollutant. For visualization purposes, we represent only 2 pollutants. The gray dots corresponds to the sensor's measurements, and the black dot to the aggregation vector, the AQI index. The gray region corresponds to the alarm region. The goal is detect whenever the AQI index is inside the gray region. In Figure 1.2 we present 3 examples. The first one, all sensors and the AQI index are outside the alarm region. In the second plot, the AQI index is outside the alarm region, although one of the sensors is inside the alarm region. The third plot, illustrate the case where the AQI index is inside the alarm region, although all sensors are outside the alarm region. These examples illustrate that information of individual sensors is not enough to make a decision about the global state of the network. Sensors need to share information to reach a correct decision.

The method is based on local computations with reduced communications between sensors. The base idea is that the aggregated function is always inside the convex-hull of the vectors space (see Figure 1.3 A and B). Suppose that all points share a reference point. Each sensor can compute a sphere with diameter the current measurement and the reference point. If all spheres are in the normal region, the aggregated value is also in the normal region. This holds,

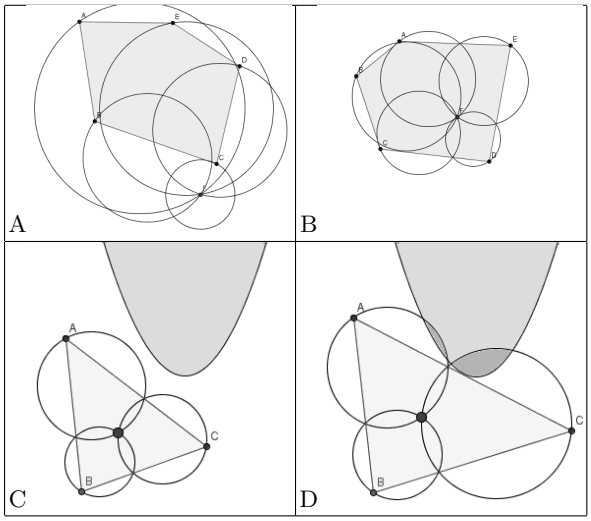

Fig. 1.3. The bounding theorem: the convex-hull of sensors is bounded by the union of spheres. Sensors only need to communicate their measurements when the spheres are non-monochromatic.

because the convex-hull of all vertex is bounded by the union of the spheres (see Figure 1.3 C and D). In the case that a sphere is not monochromatic, the node triggers the re-calculation of the aggregated function. Sensors broadcast their current measurements, and a new common point is computed.

The algorithm guarantees that any alarm is detected and no false alarms are signalled. The algorithm only uses local constraints. Mostly only local computations are required and this minimizes the communications between sensors.

1.3 Clustering Distributed Data Sources

Clustering is the most popular technique for data understanding. The basic idea behind clustering streaming data sources is to find groups of sources that behave similarly through time, which is usually measured in terms of the distance between the data series or the data distribution. Let X be a sensor node producing observations x_i at each time step i. The goal of an incremental clustering system for streaming data sources is to find (and make available at any time i) a partition $C(i)$ of data sources, where data sources in the same cluster tend to be more alike than data sources in different clusters (Rodrigues et al., 2008; Gama, 2010; Rodrigues et al., 2011). We propose a local algorithm to perform clustering of sensors on ubiquitous sensor networks, based on the moving average of each node's data over time. *L2GClust* has two main characteristics. On one hand, each sensor node keeps a sketch of its own data. On

Algorithm 1: The Monitoring Threshold Functions Algorithm (sensor node).

```
begin
    Broadcast Initial position ;
    Compute an initial reference point ;
    foreach new measurement do
        Compute the sphere with diameter defined by the current
        measurements and the reference point and check its colour;
        if sphere non monochromatic then
            Broadcast the actual measurement;
            Recompute a new reference point;
        if new messages with current measurements from other sensors
        received then
            Recompute the reference point;
```

the other hand, communication is limited to direct neighbours, so clustering is computed at each node. The moving average of each node is approximated using memoryless fading average, while clustering is based on the furthest point algorithm applied to the centroids computed by the node's direct neighbours. This way, each sensor acts as data stream source but also as a processing node, keeping a sketch of its own data, and a definition of the clustering structure of the entire network of data sources.

In this work we search for a definition of k clusters of sensor nodes, with k previously known by the system. Although this simple example lacks some of the common characteristics of real-world scenarios (e.g. unknown number or clusters or unbalanced data), its extension is straightforward. If the number of clusters to find is unknown, each node could search for a clustering with different number of clusters. As only centroids are transmitted and used as single points (as if operating with ensembles of clusters), there's no need to know how many points come from each node; all centroids that are received are included in the clustering as single points. For unbalanced data (in terms of the assignment of nodes to clusters) we believe that the convergence would take longer but deeper analysis is required in future work.

1.3.1 Local data stream sketches

As previously stated, we consider that each sensor produces a univariate stream of data, and we want to define a clustering structure for the sensors, where sensors producing streams, which are alike, are clustered together. Hence, we should consider techniques that project each sensor's data stream into a reduced set of dimensions that suffice to extract similarity with other sensors. These estimates can be seen as the sensor's current view of its own data, giving a sign of where in the data-space this sensor is included (Ro-

drigues et al., 2010). One-way to summarize a data stream x is by computing its sample mean $\hat{\mu}_x$ and standard deviation $\hat{\sigma}_x$. Our approach is to keep track of the moving average of each sensor, as an estimate of the sample mean of most recent data.

Memory-less fading average.

Each sensor produces data continuously. Given this, each sensor s is responsible of keeping its own estimate of the sample mean ($\hat{\mu}_s$) in a online fashion. Moving averages are usually easy to compute, if we can keep a small buffer of data points (Rodrigues et al., 2010). However, in such resource-demanding scenarios, this is seldom the case. Nonetheless, sum-based statistics computed on sliding windows can be approximated by weighting the sums using fading statistics (Gama et al., 2013). The *α-fading sum* $S_{x,\alpha}(i)$ of observations from a stream x is computed at time $\forall i > 0$, as: $S_{x,\alpha}(i) = x_i + \alpha \times S_{x,\alpha}(i-1)$, where $S_{x,\alpha}(0) = 0$. In the computation, α ($0 \ll \alpha < 1$) is a constant determining the forgetting factor of the sum. This way, the *α-fading average* at observation $\forall i > 0$ is then computed as: $M_{x,\alpha}(i) = \frac{S_{x,\alpha}(i)}{N_\alpha(i)}$, where $N_\alpha(i) = 1 + \alpha \times N_\alpha(i-1)$ is the corresponding *α-fading increment*, with $N_\alpha(0) = 0$. An important feature of the α-fading increment is that: $\lim_{i \to +\infty} N_{\alpha<1}(i) = \frac{1}{1-\alpha}$. Each value of α, which should be close to 1 (e.g. 0.999), will converge to sliding windows of different sizes. This way, at each observation i, $N_\alpha(i)$ gives an approximated value for the weight given to recent observations used in the α-fading sum.

1.3.2 Local clustering of stream sources

The goal is to have at each local site an approximation of the global clustering structure of the entire sensor network. Each sensor should include incremental clustering techniques which operate with distance metrics developed for the dimensionally-reduced sketches of the data streams. Also, and although in several real-world scenarios this is not true, we should not assume the sample mean of each sensor to be correlated with its physical location and connectivity, as the matching between data clusters and physical clusters is a promising strategy for sensor network comprehension, so we should not bias the clustering solution (Rodrigues et al., 2010). Given the simple sketch definition, the dissimilarity between two sensors x and y is the absolute distance between their sample means, $d(x, y) = |\hat{\mu}_x - \hat{\mu}_y|$.

Neighbourhood interaction.

Each sensor x is not only able to sketch its own data in a dimensionally-reduced definition (the fading average $M_{x,\alpha}$), but it is also able to interact with its neighbouring nodes η_x. The main characteristic of our approach is that, at each new observation i produced by sensor x, instead of sending its own sketch $M_{x,\alpha}$ to its neighbours η_x, the node sends its own estimate of the

global clustering $C_x(i)$. Note that, with this approach, each node needs to keep an estimate of the global cluster centers $C_x(i) \approx C_g(i)$. This estimate can be seen as the sensor's current view of the entire network which, together with its own sketch, gives a sign of where in the entire network data-space this sensor is included.

At first observations, each sensor node x has only access to its own sketch $M_{x,\alpha}(i)$. However, with neighbour nodes broadcasting their approximations of the global clustering structure $C_y(i), \forall y \in \eta_x$, node x suddenly has access to several data points which are believed by other nodes to be the real cluster centers. Let $P_x(i)$ be the complete set of clustering definitions $\{C_j(i) \,|\, j \in \eta_x\}$ received by node x between observations x_{i-1} and x_i. The set of points used in the clustering step includes: $\hat{\mu}_x$, the node's own sketch; $C_x(i-1)$, the node's approximation of global cluster centers (computed before observation x_i); and $P_x(i)$, the centroids sent by node's direct neighbours. Therefore, $C_x(i)$ is computed by clustering the set of points $\{M_{x,\alpha}(i)\} \cup C_x(i-1) \cup P_x(i)$.

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 clustering. This way, next time this sensor uses or transmits its estimate $C_x(i)$ of the global clustering structure, it is already updated with its most recent sketch and neighbours' information.

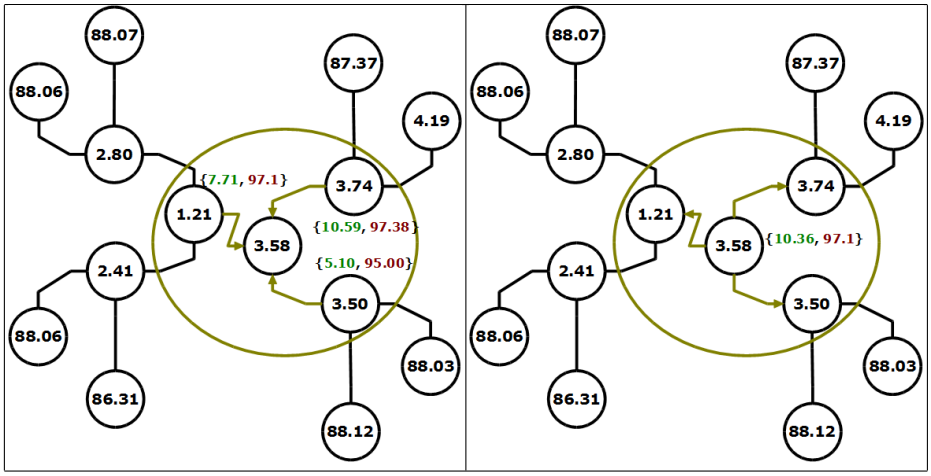

Fig. 1.4. The two main local steps in L2GClust. In the left figure, each node receives data from direct neighbours. Each node recomputes their centroids and send the new centroids to the neighbour nodes (right figure).

Furthest-point clustering.

In the general task of finding k centers given m points, there are two major objectives: minimize the *radius* (maximum distance between a point and its closest cluster center) or minimize the *diameter* (maximum distance between two points assigned to the same cluster) (Cormode et al., 2007). The *Furthest Point* algorithm (Gonzalez, 1985) gives a guaranteed 2-approximation for both the *radius* and *diameter* measures. It begins by picking an arbitrary point as the first center, c_1, then finding the remainder centers c_i iteratively as the point that maximizes its distance from the previously chosen centers $\{c_1, ..., c_{i-1}\}$. After k iterations, one can show that the chosen centers $\{c_1, c_2, ..., c_k\}$ represent a factor 2 approximation to the optimal clustering (Cormode et al., 2007).

This strategy gives a guaranteed definition of the cluster centers, computed by finding the center k_i of each cluster after attracting remainder points to the closest center c_i. Since we are applying clustering to cluster centroids, we are in fact merging clustering definitions, a known technique which has been argued to give good results (Cormode et al., 2007).

1.4 Conclusions

In this paper, we have discussed the problem of learning global models from distributed local information. We have presented a clustering algorithm for data streams generated on wide sensor networks producing high speed data, from a dynamic (time-changing) environment. The algorithms run locally in each node of the network, processing their own data and communicating aggregated data to its neighbours. This is an important characteristic in several applications, because it preserves user's privacy. A good characteristic of the proposed systems is the ability to adapt to resource-restricted environments: system granularity can be defined given the resources available in the network's processing sites. The proposed algorithms reduce both the dimensionality and the communication burdens, by exploiting limited computational resources at each local sensor.

Acknowledgements

This work was supported by Sibila and Smartgrids research projects (NORTE-07-0124-FEDER-000056/59), financed by North Portugal Regional Operational Programme (ON.2 O Novo Norte), under the National Strategic Reference Framework (NSRF), through the Development Fund (ERDF), and by national funds, through the Portuguese funding agency, Fundação para a Ciência e a Tecnologia (FCT), and by European Commission through the project MAESTRA (Grant number ICT-2013-612944).

References

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

Du, W., J. Deng, Y. Han, P. Varshney, J. Katz, and A. Khalili (2005). A Pairwise Key Predistribution Scheme for Wireless Sensor Networks. *ACM Transactions on Information and System Security 8*(2), 228–258.

Gama, J. (2010). *KnowledgeDiscovery from Data Streams*. Data Mining and Knowledge Discovery. Atlanta, US: Chapman & Hall CRC Press.

Gama, J., R. Sebastião, and P. P. Rodrigues (2013). On evaluating stream learning algorithms. *Machine Learning 90*(3), 317–346.

Giannella, C., K. Liu, T. Olsen, and H. Kargupta (2004). Communication efficient construction of decision trees over heterogeneously distributed data. In *Proceedings of the Fourth IEEE International Conference on Data Mining*, pp. 67–74. IEEE Press.

Gonzalez, T. F. (1985). Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science 38*(2/3), 293–306.

Kargupta, H., A. Joshi, K. Sivakumar, and Y. Yesha (2004). *Data Mining: Next Generation Challenges and Future Directions*. AAAI Press and MIT Press.

May, M. and L. Saitta (Eds.) (2010). *Ubiquitous Knowledge Discovery*. LNAI 6202, Springer.

Rodrigues, P. P., J. Gama, J. Araújo, and L. M. B. Lopes (2011). L2gclust: local-to-global clustering of stream sources. In W. C. Chu, W. E. Wong, M. J. Palakal, and C.-C. Hung (Eds.), *SAC*, pp. 1006–1011. ACM.

Rodrigues, P. P., J. Gama, and L. Lopes (2010). Knowledge discovery for sensor network comprehension. In A. Cuzzocrea (Ed.), *Intelligent Techniques for Warehousing and Mining Sensor Network Data*, pp. 118–134. Information Science.

Rodrigues, P. P., J. Gama, and L. M. B. Lopes (2008). Clustering distributed sensor data streams. In *European Conference on Machine Learning and Knowledge Discovery in Databases*, Volume 5212 of *Lecture Notes in Computer Science*, Antwerp, Belgium, pp. 282–297. Springer.

Sharfman, I., A. Schuster, and D. Keren (2007). A geometric approach to monitoring threshold functions over distributed data streams. *ACM Transactions Database Systems 32*(4), 301–312.

Zhu, S., S. Setia, and S. Jajodia. LEAP: efficient security mechanisms for large-scale distributed sensor networks. In *CCS '03*, pp. 62–72. ACM Press.