Location Based Distributed Spectral Clustering for Wireless Sensor Networks

Gowtham Muniraju[1], Sai Zhang[1], Cihan Tepedelenlioğlu[1], Mahesh K. Banavar[2],
Andreas Spanias[1], Cesar Vargas-Rosales[3] and Rafaela Villalpando-Hernandez[3]
SenSIP Center, [1]Arizona State University, [2]Clarkson University and [3]Tecnologico de Monterrey

***Abstract*—A distributed spectral clustering algorithm to group sensors based on their location in a wireless sensor network (WSN) is proposed. For machine learning and data mining applications in WSN's, gathering data at a fusion center is vulnerable to attacks and creates data congestion. To avoid this, we propose a robust distributed clustering method without a fusion center. The algorithm combines distributed eigenvector computation and distributed K-means clustering. A distributed power iteration method is used to compute the eigenvector of the graph Laplacian. At steady state, all nodes converge to a value in the eigenvector of the algebraic connectivity of the graph Laplacian. Clustering is carried out on the eigenvector using a distributed K-means algorithm. Location information of the sensor is only used to establish the network topology and this information is not exchanged in the network. This algorithm works for any connected graph structure. Simulation results supporting the theory are also provided.**

***Index Terms*—Wireless sensor network, spectral clustering, distributed K-means, machine learning, consensus.**

I. Introduction

A wireless sensor network (WSN) consists of a large number of low-cost, multi-functional sensors with power, bandwidth, and memory constraints, operating in remote environments with sensing and communication capabilities [1]. Typical applications of WSN's include physiological and environmental monitoring, precision agriculture, factory instrumentation, and inventory tracking [2]. In WSN's, the position of the sensors need not be predetermined, which allows for random deployment in different configurations [3]. For instance, sensors deployed on volcanic mountains to obtain seismic data can be in a concentric circular configuration. In environmental monitoring applications, the data collected from the sensors and the location of the sensors are highly correlated. In such applications, it is often necessary to perform distributed location-based clustering of the deployed sensors.

Clustering is a process of grouping a set of unlabeled observations or records into groups of similar observations [4]. To handle large datasets, parallel implementations of K-means and expectation maximization (EM) algorithms have been proposed in [5], [6]. However, these approaches are not feasible in WSN's due to power and bandwidth constraints [7]. In [8], the EM algorithm for mixture of probabilistic principle component analyzers is extended to a summing variant and then transformed into the distributed EM algorithm. Reference [9] presents a so-called distributed K-means++ algorithm for initializing centroids and then develops distributed K-means and fuzzy c-means algorithms. A distributed K-means algorithm based on weight-entropy regularization is proposed in [10]. Authors of [11], propose a deterministic, and a probabilistic approach for distributed clustering by using consensus based formulations and distributed optimization techniques and extends this to identify outliers. Reference [12] proposes a distributed spectral clustering algorithm using diffusion strategies to exchange data in the network, and then apply matrix completion and distributed gradient descent.

Classical clustering algorithms such as K-means and EM algorithms suffer from several drawbacks. The log likelihood functions may have several local minima, requiring these algorithms to have multiple restarts to obtain the desired results [13]. The K-means algorithm is also sensitive to initialization. However, clustering algorithms such as spectral clustering [14] and density based spatial clustering of applications with noise (DBSCAN) [15] address these aforementioned problems. DBSCAN does not require prior knowledge of the number of clusters, but it is very sensitive to input parameters, radius ϵ and minimum number of points inside the ϵ-sphere. Spectral clustering makes use of spectral graph theory to cluster data based on the connectivity rather than the compactness in the data. References [13], [14] develop spectral clustering for a centralized implementation. Although centralized computation is accurate, distributed computation has benefits in several areas such as power management, fault tolerance, cost of implementation and memory management.

To the best of our knowledge, a fully distributed spectral clustering algorithm to cluster the sensors based on the sensor's location has not been addressed before. The proposed method computes the eigenvector corresponding to the second smallest eigenvalue of the graph Laplacian using the power iteration method and then clusters the eigenvector using the K-means algorithm in a distributed way. Unlike [12], we assume the graph induced by the communication radius and the location of sensors as the similarity graph. The location information of sensors is only used to establish the network topology.

The authors from Arizona State University are funded in part by the NSF award ECSS 1307982, NSF CPS award 1646542 and the SenSIP Center, School of ECEE, Arizona State University.

978-1-5386-1663-5/17/$31.00 ©2017 IEEE

Simulation results illustrate that spectral clustering performs better than K-means, for node configurations as in Figure 1.

The rest of the paper is organized as follows. In Section II, we present the system model. The mathematical background is provided in Section III. The proposed distributed spectral clustering method is presented in Section IV. Simulation results and conclusion are in Sections V, and VI, respectively.

Vectors are denoted by boldface lower-case, and matrices by boldface upper-case letters. The symbol $||\cdot||$ denotes l_2-norm for real vectors and spectral norm for symmetric matrices. The symbol $|\cdot|$ denotes absolute value for a real or complex numbers and cardinality for sets. Vector $\mathbf{1}$ represents a $N\times 1$ column vector of all ones, $[1,1\dots1]^T$. $\lambda_n(\mathbf{A})$ denotes the n^{th} smallest eigenvalue of a symmetric matrix $\mathbf{A}$ and $\mathbf{u}_n(\mathbf{A})$ denotes the corresponding eigenvector.

II. System Model and Problem Statement

We consider a network of N nodes an in [16], where the i^{th} node is located at (x_i, y_i) on a 2-D plane. The communication among the nodes is modeled as an undirected graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges connecting the nodes. Two nodes can communicate with each other if they are within a Euclidean distance of ϵ. The set of neighbors of node i is denoted by $\mathbb{N}_i=\{j|\{i,j\}\in\mathcal{E}\}$. The degree of the i^{th} node, denoted by d_i, is the number of neighbors of the i^{th} node. The degree matrix $\mathbf{D}$ is a diagonal matrix that contains the degrees of the nodes. The connectivity structure of the graph is characterized by the adjacency matrix $\mathbf{A}=\{a_{ij}\}$ defined by, $a_{ij}=1$ if $\{i,j\}\in\mathcal{E}$ and $a_{ij}=0$, otherwise. The graph Laplacian $\mathbf{L}$ is a $N\times N$ positive semi-definite matrix defined by $\mathbf{L}=\mathbf{D}-\mathbf{A}$. The smallest eigenvalue of the graph Laplacian is $\lambda_1(\mathbf{L})=0$. For a connected graph $\lambda_i(\mathbf{L})>0, i=2,\dots N$. The eigenvalue $\lambda_1(\mathbf{L})=0$, is associated with the eigenvector $\mathbf{1}$, composed of ones. Connectivity of the graph is captured by the smallest non-zero eigenvalue $\lambda_2(\mathbf{L})$ called *algebraic connectivity* and the corresponding eigenvector is called the *Fiedler vector*.

As mentioned, we assume that every sensor can communicate with other sensors within a radius of ϵ, which induces a graph topology called the similarity graph. Our goal is to cluster the sensors in a distributed way, based on their position without sharing the location information in the network. The reason we consider distributed spectral clustering over the existing distributed approaches such as K-means, EM or Gaussian mixture models [8]–[11] is due to its effectiveness for node configurations as in Figure 1.

III. Mathematical Background

A. Distributed average consensus

Distributed average consensus is a well studied method of computing the average of initial measurements distributively [17]–[19]. In this paper, distributed average consensus based formulations are used to compute the eigenvector corresponding to the algebraic connectivity of the graph Laplacian. Consider $\mathbf{x}^t=[x_1^t, x_2^t, \dots x_N^t]^T$ to be the state values of the nodes at time $t\geq 0$ and the initial state of the i^{th} node is x_i^0. The sample mean of all the state values is calculated in a distributed way as follows,

$$x_i^{t+1}=x_i^t+\alpha\sum_{j\in\mathbb{N}_i}a_{ij}(x_j^t-x_i^t), \tag{1}$$

where $t\geq 0$ is the time index and α is the step size, satisfying $0<\alpha<\frac{1}{\lambda_N(\mathbf{L})}$. Equation (1) can be written as $\mathbf{x}^{t+1}=\mathbf{W}\mathbf{x}^t$, where $\mathbf{W}=\mathbf{I}-\alpha\mathbf{L}$. Convergence of average consensus is proved in [18], [19].

B. Power Iteration method

We use matrix transformations and the power iteration method to compute the eigenvector corresponding to the second smallest eigenvalue of the graph Laplacian, $\mathbf{u}_2(\mathbf{L})$. The centralized power iteration method can be used to compute the largest eigenvalue and the corresponding eigenvector of a positive semi-definite matrix. The eigenvector corresponding to the largest eigenvalue of matrix $\mathbf{Z}$, is computed as

$$\mathbf{u}^{t+1}=\frac{\mathbf{Z}\mathbf{u}^t}{||\mathbf{Z}\mathbf{u}^t||}, t\geq 0 \tag{2}$$

where $\mathbf{u}^{(0)}$ is a initial random vector from a continuous distribution and $t\geq 0$ is the time index. As $t\to\infty$, $\mathbf{u}^t$ converges to the eigenvector of the largest eigenvalue of $\mathbf{Z}$.

However, the power iteration method on $\mathbf{L}$ results in the eigenvector corresponding to the largest eigenvalue of $\mathbf{L}$, but we are interested in computing $\mathbf{u}_2(\mathbf{L})$. Hence we use the idea of computing the Fiedler vector of $\mathbf{L}$ from [20], which involves matrix transformation, deflation and power iteration methods. The Fiedler vector of the graph Laplacian, $\mathbf{u}_2(\mathbf{L})$ can be calculated as follows: The graph Laplacian $\mathbf{L}$ is transformed into a positive semi-definite matrix $\mathbf{W}=\mathbf{I}-\alpha\mathbf{L}$, which satisfies $\mathbf{W}=\mathbf{W}^T$ and $\mathbf{W}\mathbf{1}=\mathbf{1}$ and $0<\alpha<\frac{1}{\lambda_N(\mathbf{L})}$. Note that the α used in the previous expression, and in Equation (1) can take different values within the bounds, but in our work we assume them to be the same to avoid an extra input parameter in the proposed algorithm. The λ's of $\mathbf{W}$ and $\mathbf{L}$ are related by

$$\lambda_n(\mathbf{L})=\frac{1-\lambda_{N+1-n}(\mathbf{W})}{\alpha}. \tag{3}$$

The matrix $\mathbf{W}$ is deflated to remove the largest eigenvalue and its corresponding eigenvector.

$$\mathbf{Z}=\mathbf{W}-\frac{1}{N}\mathbf{1}\mathbf{1}^T=\mathbf{I}-\alpha\mathbf{L}-\frac{1}{N}\mathbf{1}\mathbf{1}^T. \tag{4}$$

The eigenvector associated with the $\lambda_2(\mathbf{L})$, $\lambda_{N-1}(\mathbf{W})$ and $\lambda_N(\mathbf{Z})$ are equal, which can be computed by Equation (2).

$$\mathbf{u}_2(\mathbf{L})=\mathbf{u}_{N-1}(\mathbf{W})=\mathbf{u}_N(\mathbf{Z}). \tag{5}$$

This method relies on the choice of scalar α and the degree of deflation. The distributed version is discussed in Section IV-B.

C. K-means clustering method

K-means is a well known clustering method to cluster the data set into K groups. In our work, K-means algorithm is used to cluster the Fiedler vector of $\mathbf{L}$. In centralized K-means, K centroids $\boldsymbol{\mu}^t = [\mu_1^t, \mu_2^t, ..\mu_K^t]$ are randomly initialized. The algorithm performs cluster assignment and centroid update iteratively. In cluster assignment step, data points $\mathbf{y} = [y_1, y_2, \dots y_N]$ are assigned to a cluster $C_{\hat{k}}$ as,

$$y_i \in C_{\hat{k}}, \quad \text{if} \quad \hat{k} = \underset{k \in \{1,\dots K\}}{\operatorname{argmin}} ||y_i - \mu_k^t||^2. \tag{6}$$

Each y_i is assigned to one of the K clusters. After the points are assigned to a cluster, the centroid is updated by,

$$\mu_k^{t+1} = \frac{1}{|C_{\hat{k}}^t|} \sum_{y_i \in C_{\hat{k}}^t} y_i. \tag{7}$$

Stopping criteria of the algorithm is based on the successive changes in the centroid's position.

IV. Distributed Spectral Clustering

Distributed spectral clustering method involves: (i) defining the similarity graph for the network, (ii) distributively estimating the Fiedler vector of the graph Laplacian of the similarity graph [21] and (iii) clustering the Fiedler vector using the distributed K-means algorithm. Algorithm 1, explains the distributed spectral clustering method for K clusters. The input parameters, N and K are assumed to be known. The Fiedler vector of the graph Laplacian is computed distributively using the power iteration method, refer Lines (5)-(9) of the algorithm 1. The distributed K-means algorithm, Line (10), is implemented with the Fiedler vector as the input to cluster the dataset into K groups. We now further elaborate on the description of Algorithm 1.

A. Similarity Graph

In the centralized spectral clustering method, a similarity graph is constructed using the input dataset and clustering is performed on the eigenvectors of the graph Laplacian of the similarity graph [14]. The similarity graph can be constructed using various metrics such as ϵ-neighborhood, k-nearest neighbors and kernel methods. We are effectively adopting the ϵ - neighborhood method because all nodes whose pairwise Euclidean distance is less than ϵ are assumed to be connected. Note that in our setting, the similarity graph does not require an explicit construction and it is induced naturally by the communication radius ϵ and the location of the nodes.

B. Distributed Power Iteration method

To compute the eigenvectors, Equations (2) and (4) have to be computed in a distributed way. Every node generates a state value u_i^0, from a continuous distribution over $(-1, 1)$. Let g_i^t be an intermediate value at node i, at time $t \geq 0$. The iterative algorithm in Lines (5)-(9), compute the numerator of Equation (2), $\mathbf{g} = \mathbf{Zu}$ in a distributed way. Consider the equation in Line (7), where u_{avg}^t is the distributed average of $\mathbf{u}^t$, obtained via consensus in Line (6), calculated locally using Equation (1). The step size α is selected to satisfy $0 < \alpha < \frac{1}{\lambda_N(\mathbf{L})}$. Next step is to normalize the eigenvector $\mathbf{g}^t = [g_1^t, \dots g_N^t]$ to obtain u_i^{t+1}. The numerator in the Line (8) is already computed in Line (7). To compute denominator, the distributed average consensus is used over $({g_i}^t)^2$, followed by a square root. Lines (7) and (8) are computed iteratively until convergence. When t is large, all the nodes in the network converge to a reasonable estimate of the Fiedler vector of $\mathbf{L}$.

Algorithm 1 Distributed spectral clustering

```
 1: Distributed Power Iteration
 2: Input: location co-ordinates $(x_i, y_i)$, $N, \alpha$, $\mathbf{A}$
 3: Initialization
 4:   every node generates $u_i^0 = \mathrm{rand}(-1, 1), \mathbf{u} = [u_1, \dots u_N]$
 5: repeat $\{i = 1 : N\}$
 6:   $u_{avg}^t = \mathrm{avgconsensus}(\mathbf{u}^t)$
 7:    $g_i^t = u_i^t - \alpha \sum_{j \in \mathbb{N}_i} (u_i^t - u_j^t) - u_{avg}^t$
 8:    $u_i^{t+1} = \frac{g_i^t}{||\mathbf{g}^t||}$
 9: until convergence.
10: Distributed $K$-means
11: Input: $\mathbf{u} = [u_1, u_2, \dots u_N]$, $K$
12: every node generates $\boldsymbol{\mu} = [\mu_1, \dots \mu_K]$ from $\mathrm{rand}(-1, 1)$
13: repeat$\{i = 1 : N, k = 1 : K\}$
14:  compute {at every node}
15:      $\rho_{ki} = |u_i - \mu_k|$
16:     cluster assignment :
17:     assign $clusterindex = \underset{\mathrm{k}}{\operatorname{argmin}}(\rho_{ki})$
18:     update centroid:
19:     $\mathcal{U}_k = \{u_i|(i \in clusterindex = k\}$
20:     $\mu_k = \mathrm{avgconsensus}(\mathcal{U}_\mathrm{k})$
21:     centroid information exchange:
22:     flood $(0, \dots, \mu_k, \dots, 0)$
23:     update$(0, \dots, \mu_k, \dots, 0) \leftarrow (\mu_1, \dots, \mu_k, \dots, \mu_K)$
24:   go to: compute
25: until convergence
```

C. Distributed K-means

In this section, we explain the distributed implementation of the K-means algorithm [9]. To achieve this, Equations (6) and (7) must be implemented distributively. In our work, the input data for clustering is a $N \times 1$ eigenvector $\mathbf{u}_2(\mathbf{L})$, so the task reduces to a 1-D clustering. Every node generates a vector of centroids $\boldsymbol{\mu} = [\mu_1, \dots \mu_K]$ from a continuous distribution over $(-1, 1)$, since the range of the Fiedler vector lies within $(-1, 1)$, Figure 4. Each node computes the Euclidean distance ρ_{ki}, between u_i and the centroids μ_k, refer to Line (15) of the algorithm. Every node computes the minimum of ρ_{ki} and the label corresponding to the minimum of the distances will be the cluster to which the node belongs, refer Line (17). The nodes in the same cluster compute distributed average consensus over the state values, ie., u_i's and update their centroid, Line (20).

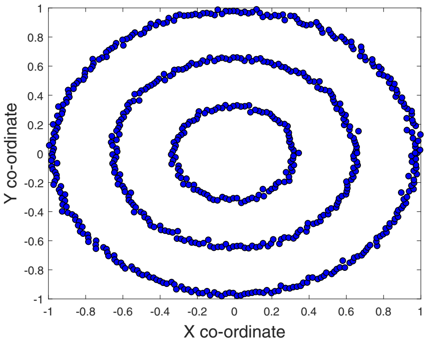

Figure 1: Synthetic data of 2-D sensor locations.

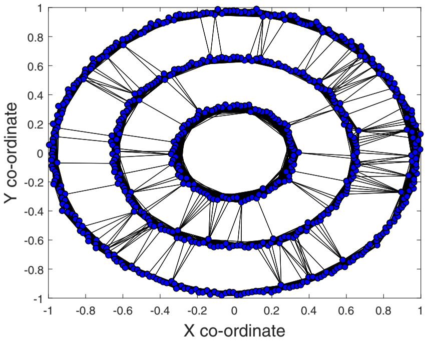

Figure 2: Similarity graph, $\epsilon = 0.3$.

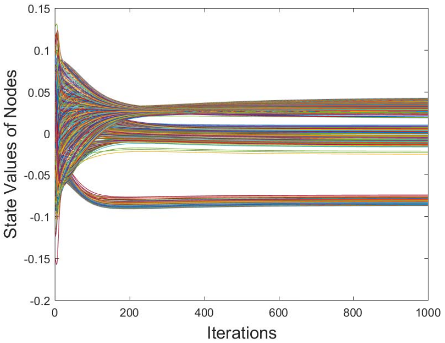

Figure 3: Convergence of nodes to the Fiedler vector.

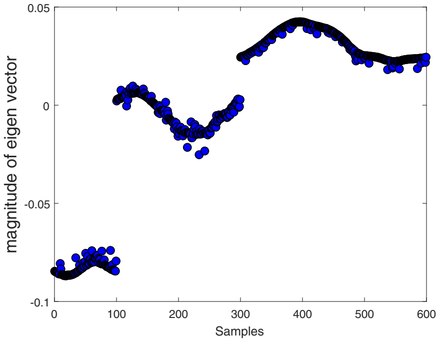

Figure 4: Fiedler Vector computed by Algorithm 1, $\alpha = 0.02$.

The total number of nodes in each cluster can be found by using distributed node counting methods [22].

After the first iteration, nodes only have knowledge of their own cluster's centroid. To obtain the centroid information of other clusters, a flooding protocol can be used [23]. To exchange the centroid information, all the nodes flood a $K \times 1$ vector in the network with the k^{th} entry being the centroid of the cluster to which the node belongs and the rest set to 0, Line (22). Once the entire network is flooded, every node starts updating the knowledge of other cluster's centroids. The $K \times 1$ centroid vector is updated by replacing 0's with the μ_k's, Line (23). Now all nodes have the centroid information of all the clusters and the cluster assignment and centroid update steps are repeated until convergence.

V. Simulations and Applications

We assume $N = 600$ nodes and $K = 3$ clusters in the input data as in Figure 1. The concentric circles dataset is used in [13] to validate the performance of the spectral clustering algorithm. This dataset is generated by adding noise from $\mathcal{N}(0, 0.01)$ to the circles of radius 0.3, 0.6 and 0.9 respectively, and normalized to ensure that the dataset lies in $(-1, 1) \times (-1, 1)$. The adjacency matrix is formed by creating a link between pairs of nodes whose distance is less than $\epsilon = 0.3$, as in Figure 2. In our setting, $\epsilon = 0.3$ is the smallest radius required to establish connectivity in the network. For real-time applications in WSN's, the unit of ϵ and the input dataset is identical. We have implemented the distributed eigenvector computation, Section IV-B by choosing $0 < \alpha < \frac{1}{\lambda_N(\mathbf{L})}$, where $\lambda_N^{-1}(\mathbf{L}) = 0.024$, closer to the upper-bound as $\alpha = 0.02$ for faster convergence. All nodes generate a value from a continuous distribution over $(-1, 1)$ as their initial state value and reach consensus to a value in the eigenvector of the algebraic connectivity of $\mathbf{L}$ as in Figure 3. Clustering is performed on the eigenvector by using the distributed K-means algorithm. The cluster centroids for $K = 3$ are initialized uniformly over $(-1, 1)$. The cluster assignment and update centroid steps are repeated until convergence. In Figure 5, the result of the distributed spectral clustering algorithm is displayed. Figure 6 shows the result of the K-means algorithm applied on the same dataset as in Figure 1. We observe that distributed spectral clustering provides more acceptable results than distributed K-means, because spectral clustering takes into account the connectivity in the dataset, which is captured in the Fiedler vector of $\mathbf{L}$.

A. *Intelligent Monitoring* & *Control of Solar PV Arrays*

At SenSIP, we have a large testbed [24] consisting of solar panels equipped with sensors whose purpose is to validate algorithms for monitoring and controlling photo-voltaic systems. This cyber physical system generates analytics from sensors that are attached on the panels. Each panel has a cluster of sensors, namely, voltage, current, temperature and irradiance. The work described in this paper can be applied to data labeling and fault localization in a large-scale system which has high correlation between the location data and measurements of the

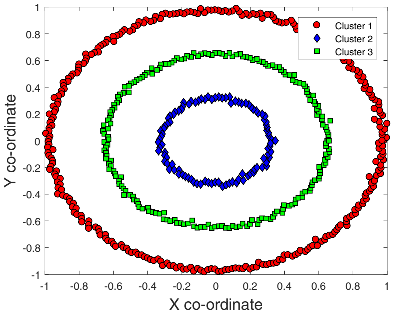

Figure 5: Result of distributed spectral clustering, $K = 3$.

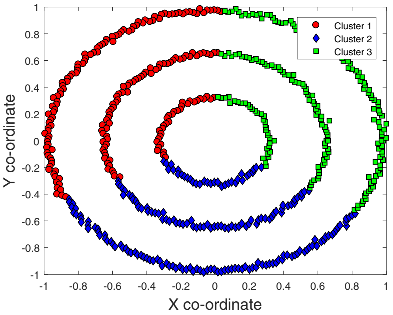

Figure 6: K-means clustering on the dataset in Fig. 1, $K = 3$.

sensors [25]. For instance, the solar panels installed over a large area form a distributed network that can be clustered into different groups based on their locations. These groups consist of PV modules affected by shading, temperature, cloud cover and irradiance. Our assumption of high correlation between location and measurements is valid for a large solar array. The algorithm we developed will be able to identify clusters within a very large utility-scale array [26] that have similar performance because of similar conditions, namely shading, temperature and irradiance, which can be used to help localize faults [27] and under performing modules. The advantage of our algorithm is that the location information or any data measurements need not be shared in the network.

VI. Conclusion

We have designed and implemented a spectral clustering method in a distributed way without any fusion center in the network, by combining the distributed eigenvector computation and distributed K-means clustering methods, to cluster the input dataset into K groups. The power iteration method is implemented distributively, to compute the Fiedler vector. All nodes converge to a value in the Fiedler vector of the graph Laplacian. Clustering is carried out on the Fiedler vector using the distributed K-means algorithm. The location information of the sensor is only used to establish the network topology and this information is not exchanged in the network. Simulation results illustrate that the distributed spectral clustering algorithm performs better than the K-means algorithm as the eigenvector of graph Laplacian is a better feature space to cluster than the input dataset.

References

[1] X. Zhang, C. Tepedelenlioglu, M. Banavar, and A. Spanias, "Node Localization in Wireless Sensor Networks," *Synthesis Lectures on Comms.*, vol. 9, no. 1, pp. 1–62, Morgan & Claypool Publishers, 2016.

[2] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with Wireless Sensor Networks," in *IEEE, ICASSP*, 2001.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Comms. mag.*, vol. 40, 2002.

[4] U. Shanthamallu, A. Spanias, C. Tepedelenlioglu, and M. Stanley, "A Brief Survey of Machine Learning Methods and their Sensor and IoT Applications," *IEEE Proc. 8th Int. Conf. IISA*, Larnaca, August 2017.

[5] W. Y. Chen, Y. Song, H. Bai, C. J. Lin, and E. Y. Chang, "Parallel Spectral Clustering in Distributed Systems," *IEEE Trans. on PAMI*, 2011.

[6] Y. Zhang, Z. Xiong, J. Mao, and L. Ou, "The Study of Parallel K-Means Algorithm," in *World Congress on ICA, IEEE*, 2006.

[7] J. B. Predd, S. B. Kulkarni, and H. V. Poor, "Distributed learning in Wireless Sensor Networks," *IEEE Signal Process. Magazine*, July 2006.

[8] H. Yin, C. Zhang, and Y. Ji, "Distributed clustering using distributed mixture of probabilistic PCA," in *Int. Conf. on FSKD, IEEE*, Aug 2014.

[9] J. Qin, W. Fu, H. Gao, and W. X. Zheng, "Distributed k -Means Algorithm and Fuzzy c -Means Algorithm for Sensor Networks Based on Multiagent Consensus Theory," *IEEE Trans. on Cybernetics*, 2017.

[10] J. Zhou, Y. Zhang, Y. Jiang *et al.*, "A Distributed K-means clustering algorithm in Wireless Sensor Networks," in *ICCSS, IEEE*, 2015.

[11] P. A. Forero, V. Kekatos, and G. B. Giannakis, "Robust clustering using outlier-sparsity regularization," *IEEE Trans. on SP.*, vol. 60, 2012.

[12] S. Scardapane, R. Altilio, M. Panella, and A. Uncini, "Distributed spectral clustering based on Euclidean distance matrix completion," in *IEEE Int. Joint Conf. on Neural Networks*, 2016.

[13] A. Y Ng, M. I. Jordan, Y. Weiss *et al.*, "On spectral clustering: Analysis and an algorithm," in *NIPS*, vol. 14, 2001.

[14] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Springer, 2007.

[15] M. Ester, H. P. Kriegel *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *KDD*, vol. 96, 1996.

[16] S. Zhang, C. Tepedelenlioglu, M. K. Banavar, and A. Spanias, "Max consensus in sensor networks: Non-linear bounded transmission and additive noise," *IEEE Sensors J.*, vol. 16, no. 24, pp. 9089–9098, 2016.

[17] S. Dasarathan, C. Tepedelenlioglu, M. K. Banavar, and A. Spanias, "Robust Consensus in the Presence of Impulsive Channel Noise." *IEEE Trans. Signal Process.*, vol. 63, no. 8, pp. 2118–2129, 2015.

[18] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, 2007.

[19] L. Xiao, S. Boyd *et al.*, "Distributed average consensus with least-mean-square deviation," *J. of Parallel & Dist. Comp., Elsevier*, 2007.

[20] P. Di Lorenzo and S. Barbarossa, "Distributed estimation and control of algebraic connectivity over random graphs," *IEEE Trans. on SP.*, 2014.

[21] T. Xiang and S. Gong, "Spectral clustering with eigenvector selection," *Pattern Recognition, Elsevier*, vol. 41, 2008.

[22] S. Zhang, C. Tepedelenlioglu, M. Banavar, and A. Spanias, "Distributed node counting in wireless sensor networks in the presence of communication noise," *IEEE Sensors Journal*, vol. 17, pp. 1175–1186, 2017.

[23] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *IEEE Int. Conf. on Mobile computing and Networking*, 1999.

[24] S. Rao, D. Ramirez, H. Braun, J. Lee, C. Tepedelenlioglu, A. Spanias *et al.*, "An 18 kw solar array research facility for fault detection experiments," in *IEEE, MELECON*, Limassol, 2016.

[25] M. Banavar, H. Braun, S. T. Buddha, A. Spanias *et al.*, "Signal processing for solar array monitoring, fault detection, and optimization," *Synth. Lect. on Power Elec.*, vol. 7, pp. 1–95, Morgan & Claypool Publishers, 2012.

[26] A. Spanias, "Solar Energy Management as an Internet of Things (IoT) Application," *IEEE Proc. 8th Int. Conf. IISA*, Larnaca, August 2017.

[27] S. Rao, S. Katoch, P. Turaga, A. Spanias, C. Tepedelenlioglu, A. Ayyanar, H. Braun, J. Lee, U. Shanthamallu, and M. Banavar, "A Cyber-Physical System Approach for Photovoltaic Array Monitoring and Control," *IEEE Proc. 8th Int. Conf. IISA*, Larnaca, August 2017.