Adaptive Data-centric Clustering with Sensor Networks for Energy Efficient IoT Applications

Sanat Sarangi
TCS Innovation Labs
Tata Consultancy Services
Mumbai, India
sanat.sarangi@tcs.com

Srinivasu Pappula
TCS Innovation Labs
Tata Consultancy Services
Hyderabad, India
srinivasu.p@tcs.com

***Abstract*—A wireless sensor network (WSN) typically involves deploying multiple nodes in an area to measure environmental parameters. WSNs are getting enveloped within the realm of IoT which significantly increases their scale of deployment. The end-objective of deploying a sensor network is to get valuable data about a region irrespective of the physical configuration used for measurement. We propose an Adaptive Data-centric Clustering algorithm for Sensor networks (ADCS), a hierarchical algorithm where user-specific data requirements are factored into the clustering decisions. Specifically, similarity in parameter variations are used as a criteria for optimization. We have deployed an eKo-based sensor network in north-eastern India to measure environmental parameters as part of a precision agriculture application. Data from this network is used to develop models to rigorously compare the performance of three variants of ADCS: ADCS-DB, ADCS-KM and ADCS-AG and arrive at useful recommendations for deployment planning.**

Keywords–sensor networks, internet of things, data-centric clustering, unsupervised learning, adaptive algorithms

I. Introduction

A Wireless Sensor Network (WSN) consists of one or more nodes with onboard sensors for bespoke real-time precision control and monitoring applications. WSN derives its strength from redundancy in terms of number of nodes. A typical deployment often has multiple nodes placed close to each other which end-up measuring the same parameter given that monitoring happens at regular intervals. While the advantage is robustness against loss of data in the event of node deaths, most practical applications would prefer to use the optimal number of nodes and the appropriate sensing strategy for these nodes keeping the energy and cost factors in view. The sensing strategy could depend on a higher semantic-level objective defined by the end-user such as measuring the variations in one or more environmental parameters for a given area where the physical network organises itself appropriately to meet this objective.

We propose a data-centric clustering algorithm ADCS to perform clustering with sensor networks in order to maximise their lifetime while meeting certain service level agreements. The significance of this is further amplified with sensor networks gradually merging with the larger fabric of modern IoT systems covering multiple sensor networks. An important IoT application area is precision agriculture which involves outdoor deployment of solar-powered sensor networks for monitoring ambient parameters. Effective energy utilisation is needed in such a scenario, for example, to maximise the network lifetime where nodes are unable to charge during night or due to cloudy conditions. We use the characteristics of sensor data obtained from our field deployment of eKo sensor nodes [1] in a tea estate to discuss how sensors could be smartly tasked with ADCS, and compare the performance results for different configurations.

II. Literature Survey

Clustering in sensor networks has traditionally been used to increase the network lifetime by minimising communication energy in the transfer of messages. LEACH [2] uses probabilistic clustering at the node level followed by one-hop transmission of the message to the clusterhead. In PEGASIS [3], the messages move through chains with fusion of information at the intermediate levels. TEEN [4] has a hierarchical clustering structure which involves thresholding at the first and all subsequent levels. Similarly, there are other algorithms that focus on other aspects of clustering such as location [5], [6], QoS [5], [6] and mobility [7]. Most of these algorithms however work with the physical rather than semantic characteristics of the data.

In data-centric routing algorithms, routing tables with addresses are not maintained on each node. Some early work where data is considered as part of the routing decisions includes directed diffusion [8], rumor routing (gossiping) [9] and SPIN [10]. Directed diffusion involves flooding the network which is costly [11]. Rumor routing [9] reduces the message overload in the network by achieving a compromise between queries and sensor data. The data-driven protocols presented so far are not resource-aware. This is addressed by SPIN [10] protocol which tries to optimise on two fronts: information with each node and its residual energy. Each node running SPIN assigns a high-level *meta-data* label which is used to perform negotiations before any data is transmitted.

More recently, mining big data to derive intelligence has gained sufficient interest with sensor networks and IoT. The characteristics of sensor data and its appropriateness as a source of big data is presented in [12]. The typical characteristics of WSN data is discussed in more detail in [13]. The 6V's of Big Data are highlighted and sensor networks are presented

DOI 10.1109/LCN.2016.68

as a data source with non-ending streams which is not the typical substrate for standard data mining techniques. Special data mining and fusion techniques such as complementary, redundant and cooperative fusion are discussed.

III. Motivation

Precision monitoring of ambient parameters such as temperature, humidity, soil moisture and leaf wetness, to name a few, are useful for a number of applications in agriculture such as development of accurate pest and disease forecasting models, irrigation models, and finding exceptional events on the farm. With closed loop control ability, higher temporal and spatial resolution, and improved accuracy, WSNs offer numerous advantages over conventional monitoring sources. As part of the digital farming initiatives of TCS, we have deployed sensor networks for disease forecasting with potato [14] and subsequently for tea gardens in West Bengal. Spatially co-located nodes have nearly similar measurement patterns, and the typical end-objective of the deployments are more about knowing the parameter variations for a region rather than the details of the physical setup used to measure them. Such requirements can be used to task the nodes to cooperatively monitor parameters in an optimal manner such that the network lifetime is maximised. The broad interest is to develop adaptive algorithms that enable optimal cooperative sensing based on features of interest.

With ADCS, we propose a smart auto-adaptive clustering strategy to measure parameter variations in a region by abstracting the tasks for the individual sensor nodes. Tolerance levels to parameter variations are used as inputs for unsupervised clustering to identify regions of interest. The focus with ADCS is on data-centric clustering with redundant fusion where the aim is to minimise the overall energy of the network. As an example, we use temperature as one of the sensed parameters to discuss the multi-level unsupervised clustering with ADCS which allows it to adapt to different scenarios.

IV. ADCS: Adaptive Data-centric Clustering for Sensor Networks

We present the generic ADCS algorithm and then discuss how it is (a) adapted to a given sensing scenario based on observed data characteristics, and (b) customised with various unsupervised clustering models. The variants of ADCS thus obtained are then compared to evaluate the most effective algorithm for the energy optimisation objective.

A. The ADCS Algorithm

Let us assume a region R covered by a set of S sensor nodes where each node $s_i \in S$ measures a parameter regularly at a given interval. Let V_i denote a set of n measurements $\{v_1, ..., v_n\}_i$ from sensor node s_i. For any two $s_i, s_j \in S$ where the data points of V_i have the same trend as V_j, the location and field characteristics of the two nodes determine whether they could be considered as measuring the same or different trends for a given parameter. For example, if s_i, s_j are geographically separated by a large distance, the similarity in observed values could be a temporal coincidence. On the other hand, if the nodes are placed near each other, a difference in elevation or overhead vegetation (such as a camouflage) would govern whether the observed trends remain the same or change over time. There could be several such constraints which help determine the extent of similarity and hence the corresponding clustering decision. Further, the conditions around the nodes may change over time so the clustering decision would need to change at regular intervals. For clarity, we discuss ADCS with clustering at two levels, L0 and L1, where L0 is at the data level and L1 is at the location level. The principle is generic enough to extend to any number of levels according to the number of criteria being considered.

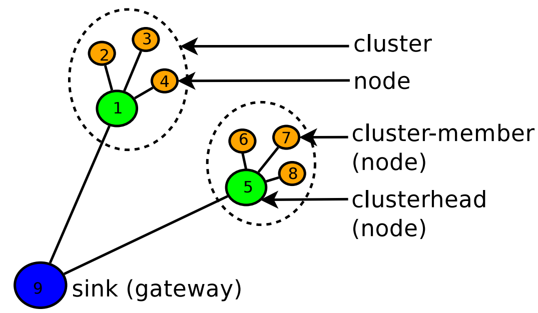


Fig. 1: A configured wireless sensor network with cluster nodes, cluster members and sink

Let us assume an n-node WSN deployment terminating at a sink which is the gateway for the WSN with the external world and executes the ADCS algorithm. Clustering with ADCS starts with a request to all nodes to send their data. The network nodes send their data to sink by transmitting directly or routing it through other nodes in the network. The data is processed by ADCS to arrive at a clustering decision which is then conveyed back to the network nodes to arrange them in clusters (Fig. 1). Each cluster in the network consists of a clusterhead (nodes 1 and 5) and one or more cluster-members (nodes 2-4 and 6-8). A clusterhead is a local aggregation point for data from cluster members and transmits directly to the sink.

During L0 clustering, ADCS identifies nodes at similar data-levels. This involves computational overhead in terms of calculation of features from the reported measurements as discussed later in Sec. V-B. The feature-set for L0 clustering is denoted by f. The order complexity of the construction of f is dependent on how the component features of this feature-set are calculated. We note that while having similar data characteristics, clusters identified in this manner may be geographically scattered. ADCS then performs the L1 clustering which groups nodes within certain physical separation of each other. We use maximum allowable distance between any two nodes in a cluster as the basis for L1 clustering. The end-result is a set of clusters where nodes in each cluster are at a similar data level and are also geographically close to each other.

Effectively, we create with ADCS an adaptive clustering framework which combines unsupervised learning with a two-level hierarchical data fusion and transfer mechanism to achieve energy efficiency. The framework is generic enough to allow a variety of models to be used. We present the background on unsupervised learning models chosen for ADCS which helped us derive the three variants: ADCS-DB, ADCS-KM, and ADCS-AG. Further, as noted earlier, redundant fusion is performed at the level of each cluster wherein we assume that only one cluster member senses the concerned parameter and transmits to its clusterhead at any given point of time, and this role is rotated between cluster members to evenly distribute energy consumption across all nodes. Numerous such optimisations could be performed within the framework.

B. *Unsupervised Clustering Models for ADCS*

We choose three unsupervised clustering algorithms DBSCAN [15], K-Means [16], and Agglomerative [17] (Ward Hierarchical) as candidates for both L0 and L1 clustering with ADCS. The input requirements and clustering scale for these algorithms seem particularly suited to WSN scenarios. Assuming the number of sensor nodes to be n, the prediction in DBSCAN [15] is for very large n and medium number of clusters m. DBSCAN is a good candidate for uneven cluster-size requirements and uneven geometry. It takes a neighbourhood parameter (ϵ) as a clustering input which is the maximum acceptable distance between any two neighbours. K-Means is good for very large n and medium number of clusters. It is useful for even cluster size requirements for a flat geometry. Agglomerative Clustering [17] is again meant for large n and number of clusters. The ability of these algorithms to handle large number of samples makes them effective candidates for ADCS. Three variants of ADCS, one with each algorithm, are thus obtained: ADCS-DB, ADCS-KM and ADCS-AG.

V. EVALUATION OF SENSOR DATA FOR CLUSTERING DECISIONS

We have installed a set of eKo nodes labelled 2, 3 and 4 in Bagdogra, West Bengal for monitoring temperature and humidity conditions which are useful for pest and disease forecasting in crops [14]. The nodes are placed within a distance of 50 meters with respect to each other and measure temperature, humidity and dew-point at regular intervals. We have 10815 samples for temperature, humidity and dew point for node 2 from Nov, 2011 to April, 2015. For node 3, we have 10086 samples from Nov, 2011 to July, 2015. For node 4, it is 7010 samples from Nov 2011 to Jan 2015.

A. *Data Characteristics for Clustering Assumptions*

During the 3.5 years of continuous run, the nodes experienced downtime periods due to lack of power, bad weather and other factors. Further, insufficient battery levels also drove down the quality of sensed data on some occasions to negative levels. For the purpose of analysis, we consider data for a period where uninterrupted measurements were received from all nodes. Figs. 2(a)–2(c) give the distributions of temperature, humidity and dew-point for nodes 2, 3, and 4 for a continuous period of 15 days from 1st Sep 2014 - 15th Sep 2014. We note that dew-point is a derived value. The sinusoidal pattern is indicative of the diurnal variation on each measurement.

We examine the frequency distribution of nodes 2, 3 and 4 for a sequence of 400 samples (Fig. 3). We note that the frequency characteristics of all three signals are similar with the maximum frequency at 10 Hz and a peak amplitude between 3.0 and 4.0. This is due to the geographical proximity of the nodes. Each node samples the ambient parameters every 15 min to ensure sufficient granularity in measuring minor variations during the course of a day. For effective information-level clustering we need to establish the degree of similarity between the measured signal sequences from the sensor nodes.

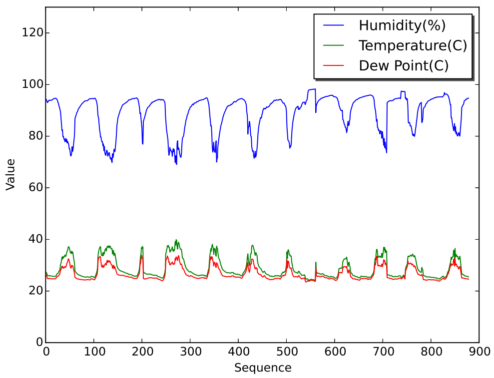
B. *Feature-set Construction for Clustering with ADCS*

From Sec IV-A, a typical measurement signal sequence $\{v_1, ..., v_n\}_i$ for a parameter of a node s_i can be considered as a vector $[v_1, ..., v_n]_i \in \mathbb{R}^n$ that needs to be compared with other signals for similarity. In order to meaningfully achieve this with an unsupervised clustering model, it is often essential to reduce the dimensionality of this input vector to make it suitable for classification. Some of the best and popular unsupervised learning models use the Euclidean distance between points in upto $\mathbb{R}^3$ as a measure for classification as classification results in higher dimensions do not remain meaningful.

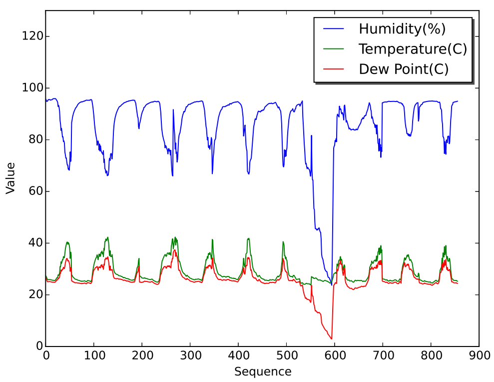
We reduce the signal sequence expressed in $\mathbb{R}^n$ by transforming it into a value in $\mathbb{R}^2$ that represents the sequence. From Sec. V-A, we use the highest frequency component of the signal and the amplitude for this frequency as the transformed feature-set $f = [\arg\max \mathrm{fft}(V_i), \max \mathrm{fft}(V_i)]$ representing each n-point vector V_i. The order complexity for construction of f for k nodes is $O(k)$ where k is the number of nodes. Ideally, feature-set used for classification need not be restricted to signal sequence from a single node. We note that joint characteristics involving more that one node, for example, RMSE and correlation coefficient of signals for node pairs may also be considered. This however requires pairwise examination of k signals which increases the complexity of f for the k nodes to $k(k+1)/2$ or O(k^2). Such conditions are kept outside the current scope of work.

VI. PERFORMANCE EVALUATION OF ADCS

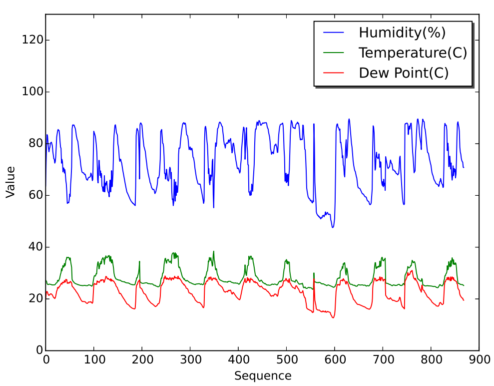
Simulation with Python is used to evaluate the performance of all variants of ADCS. A deployment area of size 50 m x 50 m is assumed where one of the diagonal coordinates is $\{(0, 0), (50, 50)\}$ and the sink is at a distant (100, 100). This is the typical scenario in which advantages associated with clustering and two hop communication from the node to the sink via a clusterhead is best realised [2]. The number of nodes randomly deployed in the area is 100 and they are uniformly



(a) Node 2 parameters over 15 days



(b) Node 3 parameters over 15 days



(c) Node 4 parameters over 15 days

Fig. 2: Node parameters over 15 days

distributed. We use our experimental results for information-level distribution of nodes for L0 clustering. The objective of the evaluation is to compare and contrast the performance of ADCS-DB, ADCS-KM and ADCS-AG for a common realistic observation set from sensors.

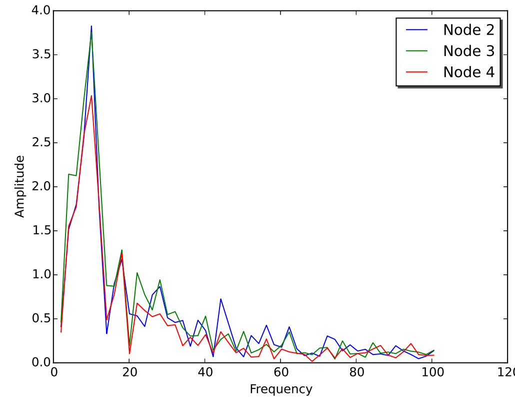


Fig. 3: Frequency distribution of temperature

Physical Parameters	Value
Number of nodes	100
Rounds of clustering	20
Node distribution area	50 m x 50 m with diagonal points (0,0) and (50,50)
Position of sink	(100,100)
Energy Parameters	**Value**
E_{elec}	50 nJ/bit
E_o	100 pJ/bit/m^2
k	2000 bits

TABLE I: Simulation Parameters

A. Clustering Parameters

To keep our evaluation close to the nature of data generated from a testbed (Sec. V), the measurement sequence for each node is assumed to be obtained from a sinusoidal sequence parametrised by the elements in the feature vector f of the form (x, y) (Sec. V-B). With this assumption for information-level (L0) clusters, we assume from Sec. V-B that the set of feature vectors $F = \{f_1, ..., f_{100}\}$, one for each of the 100 nodes, are obtained from a set of Gaussian isotropic clusters centred at a set of points $P = \{(x_1, y_1)...(x_m, y_m)\}$

Algorithm	Clustering Mod.	Parameter
ADCS-DB	DBSCAN (L0) DBSCAN (L1)	epsilon ϵ (0.6) min no. of cluster members (2) [both L0 and L1]
ADCS-KM	K-Means (L0) K-Means (L1)	no. of clusters (5) no. of clusters (5)
ADCS-AG	Aggl. (L0) Aggl. (L1)	no. of clusters (5) no. of clusters (5)

TABLE II: ADCS and its variants with unsupervised clustering methods

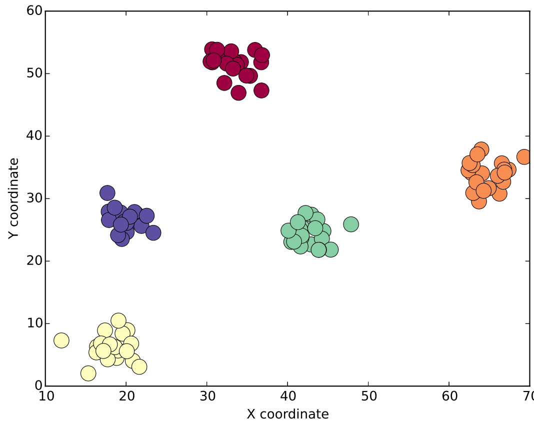


Fig. 4: Distribution of measurement feature-set associated with each node for a given round of clustering

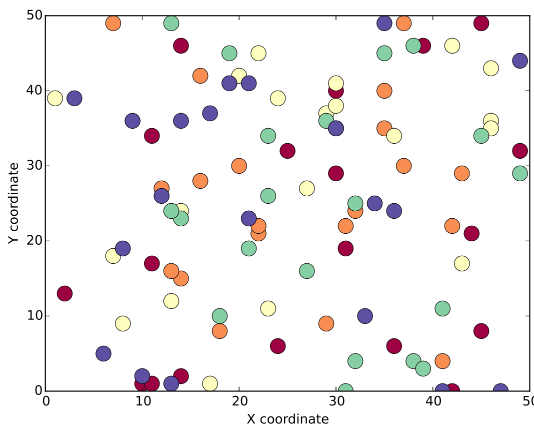


Fig. 5: Position of nodes for a given round of clustering with colour indicating their feature-set cluster

with a standard deviation s. The value of s is dependent on the tolerance level for the span that we wish to have for clustering at the data level (L0). For a randomly deployed 100-node sensor network with a communication energy model assumption similar to ours (Sec VI-B), it has been shown that the optimal number of clusters is 3–5 where the lowest energy consumption typically happens with 5 clusters [18]. So assuming $m = 5$ to facilitate comparison between ADCS variants, a typical set of values for P and s for a given round of clustering are $P = \{(17.1, 6), (21.1, 27), (33.0, 50), (43.1, 24), (65, 34)\}$ and $s = 2$ so that there is no overlap between clusters. Fig. 4 shows this distribution of P with each cluster in a different colour. Fig. 5 shows the (randomly distributed) location of each of the nodes.

Kmeans and Ward clustering require a default number of clusters to be specified. So, we set this to 5 for ADCS-KM and ADCS-AG at L0 and L1. DBSCAN on the other hand depends on the value of ϵ which is indicative of separation between neighbours within a cluster. We use the following assumptions to set the value of ϵ. If we divide 50 m x 50 m area (A) into 5 segments (each being one cluster), and assume each of these segments to be a square, the side of the square area would be 22 m ($\sqrt{(A/5)}$). Assuming the 100 nodes are equally distributed in these 5 clusters, there would be 20 nodes in each cluster. If these nodes were in a straight line (worst case), the linear separation between the nodes (neighbours) would be ≈1 m. So, we make a conservative assumption of 0.6 as the value of ϵ for clustering at both L0 and L1 levels. We also note that a value of 0.6 or lower for ϵ helps DBSCAN in accurately identifying the clusters created for dataset P. The parameter assumptions for each algorithm is given in Table II.

$$E_{Tx}(k, d) = E_{elec}k + E_0kd^2 \tag{1}$$

$$E_{Rx}(k) = E_{elec}k \tag{2}$$

B. Energy Parameters

Eqn. 1 and Eqn. 2 are the free-space path-loss equations used to calculate the energy consumed in transmission and reception of messages between (a) cluster-members and clusterheads and (b) clusterheads and the sink for a distance d [3]. Here, E_{elec} is the energy dissipated in the transmit and receive circuitry, E_o is the energy required to transmit a message, and k is the length of the message in bits.

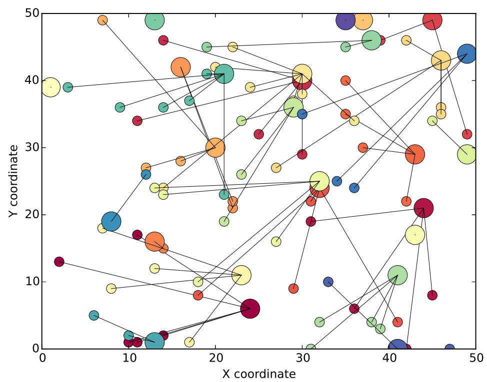
C. Simulation

Twenty rounds of clustering each with a different set of values for P are used to evaluate the performance of ADCS-KM, ADCS-AG and ADCS-DB. With each clustering round, the nodes in the network are organised into multiple clusters where nodes in each cluster have similar parameter values. We configure the simulation to have each cluster send its parameter value to the sink 5 times before the next round of clustering. To achieve this, nodes in each cluster take turns to send their measurements to the clusterhead, and the clusterhead sends it to the sink. Various results obtained from simulation are used to justify the use of the appropriate algorithm for a given scenario.

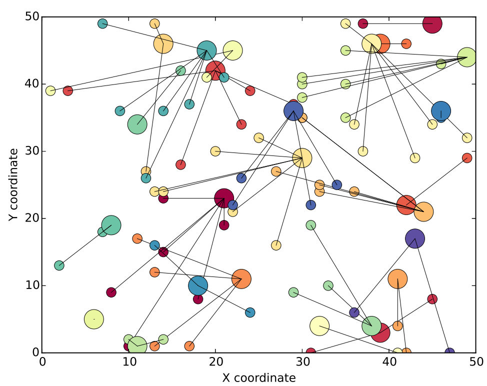
VII. Results and Discussion

We present the performance of clustering with ADCS-KM, ADCS-AG and ADCS-DB to highlight their suitability under different situations. We know from Table II that the parameters required by each algorithm to perform clustering could be potentially different. Therefore, we present as part of our performance results certain metrics which show that the parameter assumptions can be considered equivalent. The network configuration for a typical round of clustering for ADCS-DB, ADCS-KM and ADCS-AG are shown in Figs. 6(a)–6(c).

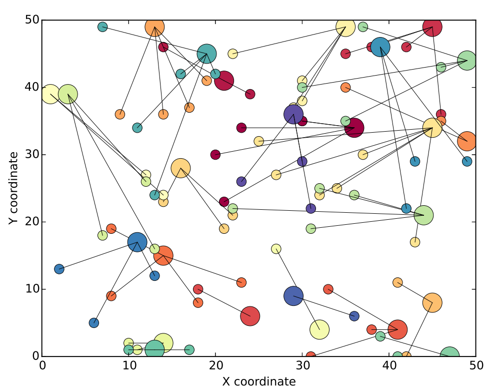
ADCS-DB uses an augmented form of DBSCAN where DBSCAN is followed by additional support logic to help overcome some constraints. When only DBSCAN is used, we note that 100% nodes are clustered at L0 and ≈70% nodes at



(a) ADCS-DB clusters after L0 and L1 clustering for a given round



(b) ADCS-KM clusters after L0 and L1 clustering for a given round



(c) ADCS-AG clusters after L0 and L1 clustering for a given round

Fig. 6: ADCS clusters after L0 and L1 clustering

L1 levels as DBSCAN is designed to leave out nodes that are unable to meet the ϵ criteria. So, we follow it up with support logic where each of the remaining 30% nodes (R) associate with one of the final post-L1 set of clusters (C) provided (a) a node $r \in R$ was in the same L0 cluster as the nodes in a cluster $c \in C$, and (b) the Euclidean distance between r and

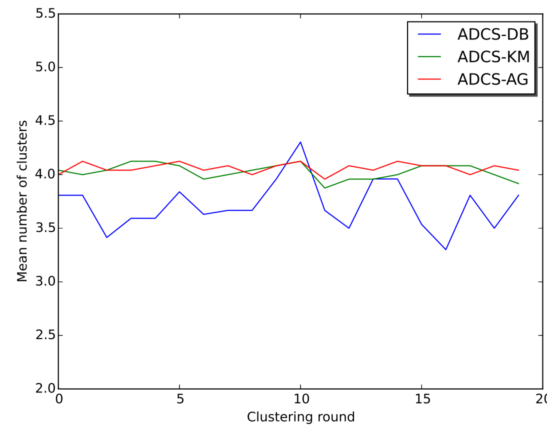


Fig. 7: Mean number of clusters after each round for 20 rounds (0-19) of clustering

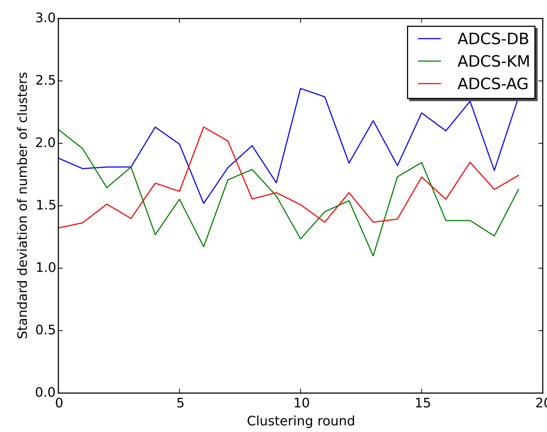
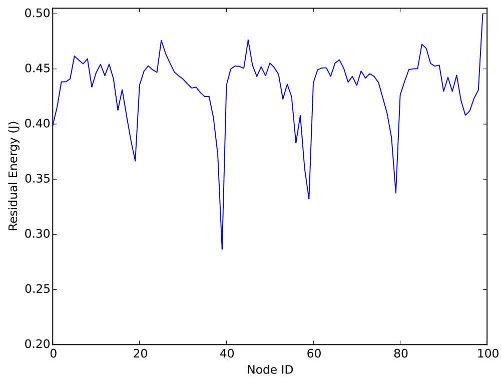


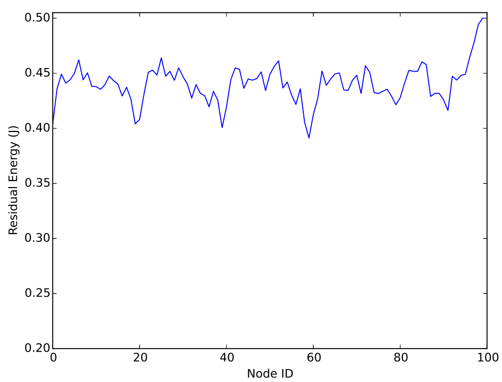
Fig. 8: Standard deviation of number of clusters after each round for 20 rounds (0-19) of clustering

the farthest node in c is less than a threshold T. We assume $T = 25\ m$ as this is the order of distance between clusterhead and cluster-members observed with ADCS-KM and ADC-AG.

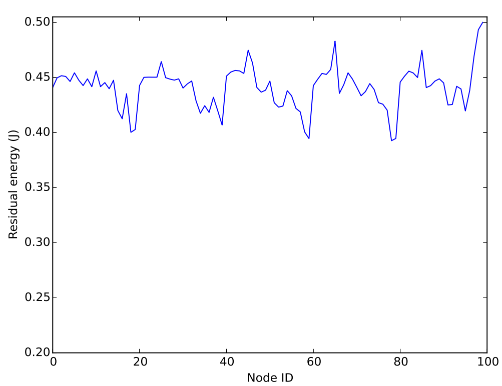
Fig. 7 and Fig. 8 show the mean and standard deviation of the number of clusters over all clustering rounds for all three algorithms. We note that the mean number of nodes per cluster hovers at 4 for both ADCS-KM and ADCS-AG while it oscillates between 3 and 5 for ADCS-DB. The standard deviation for ADCS-KM and ADCS-AG hovers at 1.5 while it increases to 2 for ADCS-DB. So, KMeans and Agglomerative clustering produce more balanced clusters. The net residual energy for each node after 20 rounds of clustering with ADCS-DB, ADCS-KM and ADCS-AG is given in Figs. 9(a)–9(c). The corresponding mean residual energy of the network in the form $\mu \pm \sigma$ is given as 46.05 ± 1.89, 46.37 ± 1.72 and 46.41 ± 1.70. ADCS-AG has the highest residual energy while ADCS-DB has the lowest. The gradient of residual energy over



(a) Residual energy with ADCS-DB after 20 rounds of clustering



(b) Residual energy with ADCS-KM after 20 rounds of clustering



(c) Residual energy with ADCS-AG after 20 rounds of clustering

Fig. 9: Residual energy after 20 rounds of clustering

all clustering iterations is shown in Fig. 10. The mean distance between a cluster member and its clusterhead is 26.38, 23.52 and 24.55 for ADCS-DB, ADCS-KM and ADCS-AG.

Our analysis shows that ADCS-AG is most effective in maximising the network lifetime followed closely by ADCS-KM and then ADCS-DB. We note that while the residual

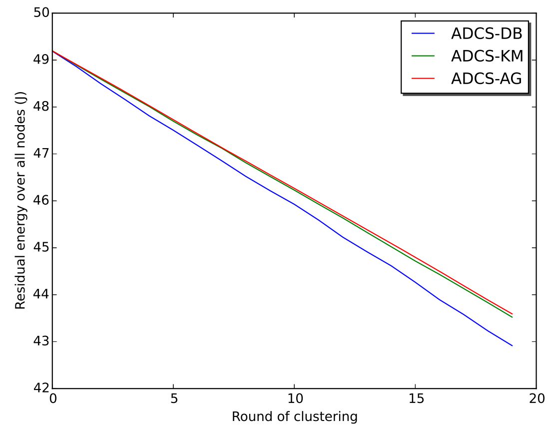


Fig. 10: Residual energy after each round for 20 rounds (0-19) of clustering

energy with ADCS-DB is comparatively lower, it offers a higher level of flexibility in specifying end-user requirements for clustering in terms of allowable distance between members of a cluster rather than the number of clusters. Therefore, the three algorithms together can help address different requirements for different scenarios within a common framework.

Since ADCS does not extend any of the algorithms discussed in Sec. II, a direct comparison with them would not be meaningful. For example, LEACH makes a heuristic assumption on the number of clusters regardless of the data associated with each cluster and has a cluster-level data aggregation strategy which is different from ADCS. TEEN uses predefined thresholds to decide message transmission policies through the clusters whereas ADCS tries to find similar nodes based on their measurement patterns which acts as a basis for cluster formation. SPIN focuses more on optimising the message-routing aspects and related overheads which is not the focus for ADCS. In future, we will develop suitable extensions and make appropriate assumptions for such algorithms so that we can compare them with ADCS. We hope to develop new features for ADCS in the process to address a wider range of application scenarios.

VIII. Using ADCS in IoT Applications

ADCS has been developed keeping the end-user IoT application scenarios in view. While ADCS itself is configurable with different clustering algorithms that it can support, its operational orchestration for different components is clearly defined to facilitate easy integration and deployment planning. We have developed an Agro-IoT digital farming platform called InteGra [19] which has sophisticated data acquisition, analysis and dissemination capabilities. One of the acquisition components of InteGra for sensor data is KwikSense which is able to acquire and store data in the OGC Sensor Observation Service-defined format. This interacts with gateways deployed in farms, each talking directly to a set of sensor nodes over a low-power network protocol such as ZigBee. As clustering related messages with ADCS are essentially for network management, ADCS at the gateway-node would, for example, help filter out such messages so that only the valuable data reaches the InteGra server. In a more complex scenario with a large number of sensor nodes associated with the gateway, the sensor nodes may organise themselves in clusters with the gateway as the sink.

IX. Conclusions

We have proposed an algorithm, ADCS, to carry out adaptive clustering with sensor networks which performs clustering both at data and location levels. We compare the performance of different variants of ADCS with various unsupervised clustering algorithms and compare the performance which helps arrive at recommendations to maximise the lifetime of the underlying sensor network and tradeoffs in terms of flexibility in specifying user requirements and performance. Finally, the placement of various components of ADCS is discussed in the context of an IoT platform to illustrate its impact and future scope.

X. Acknowledgements

We thank the staff of Kadambini Tea Estate, West Bengal who helped us immensely in the installation of the sensor nodes and data collection for the experiments.

References

[1] "eKO Sensor Node." [Online]. Available: http://www.memsic.com/wireless-sensor-networks

[2] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. 33rd Annual Hawaii Intl Conf. System Sciences, 2000*. IEEE, 2000, pp. 10–pp.

[3] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems," in *IEEE Aerospace Conference Proceedings, 2002*, vol. 3, 2002, pp. 3–1125–3–1130 vol.3.

[4] A. Manjeshwar and D. P. Agrawal, "TEEN: a routing protocol for enhanced efficiency in wireless sensor networks," in *Proc. 15th IEEE Intl Parallel and Distributed Proc. Symp.*, 2001, p. 30189a.

[5] M. E. azhari, R. Latif, and A. Toumanari, "Minimizing energy consumption in wireless sensor networks using solar powered sensors," *Materials and processes for energy: communicating current research and technological developments*, pp. 84–94, 2013.

[6] R. Sobti, "A Comparative Study on Network structure based Routing Protocol and its Variants in Wireless Sensor Networks: A Survey," *International Journal of Computer Applications*, vol. 117, no. 12, 2015.

[7] S. Sarangi and S. Kar, "Genetic algorithm based mobility aware clustering for energy efficient routing in wireless sensor networks," in *17th IEEE International Conference on Networks (ICON)*, Dec 2011, pp. 1–6.

[8] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proc. 6th ACM Annual Intl Conf. Mobile Computing and Networking*, 2000, pp. 56–67.

[9] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM, 2002, pp. 22–31.

[10] R. Sobti, "A Comparative Study on Network structure based Routing Protocol and its Variants in Wireless Sensor Networks: A Survey," *International Journal of Computer Applications*, vol. 117, no. 12, 2015.

[11] P. Hebden and A. R. Pearce, "Data-centric routing using Bloom filters in wireless sensor networks," in *Fourth International Conference on Intelligent Sensing and Information Processing, ICISIP'06*. IEEE, 2006, pp. 72–77.

[12] M. Alwadi and G. Chetty, "Energy Efficiency Data Mining for Wireless Sensor Networks Based on Random Forests," *International Journal on Data Mining and Intelligent Information Technology Applications*, vol. 4, no. 1, p. 1, 2014.

[13] M. M. Fouad, N. E. Oweis, T. Gaber, M. Ahmed, and V. Snasel, "Data Mining and Fusion Techniques for WSNs as a Source of the Big Data," *Procedia Computer Science*, vol. 65, pp. 778–786, 2015.

[14] P. A. Patil, J. Raval, B. G. Jagyasi, N. Warke, and P. P. Vaidya, "Demo Abstract: Lessons Learned from a WSN Deployment for Precision Agriculture," in *EWSN 2014: Posters and Demos*, 2014.

[15] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[16] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA., 1967, pp. 281–297.

[17] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American statistical association*, vol. 58, no. 301, pp. 236–244, 1963.

[18] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications,*, vol. 1, no. 4, pp. 660–670, 2002.

[19] "InteGra Digital Farming Platform," Barcelona, Feb. 2016, Mobile World Congress 2016. [Online]. Available: https://twitter.com/TCS_News/status/702512713478438913