ON-IN: An On-Node and In-Node Based Mechanism for Big Data Collection in Large-Scale Sensor Networks

Marwa Ibrahim[a,b,†], Hassan Harb[a,b,‡], Abbass Nasser[a,b,†,‡], Ali Mansour[a,†] and Christophe Osswald[a,†]
[a]*Lab-STICC, CNRS UMR 6285, ENSTA-Bretagne, Brest, France*
[b]*Computer Science department, American University of Culture and Education (AUCE), Beirut, Lebanon*
Emails:[†]{firstname.lastname@ensta-bretagne.fr}, [‡]{firstnamelastname@auce.edu.lb}

***Abstract*—Nowadays, data are collected everywhere from searches on Google to posts on social media. Thus, the era of big data is started. Among many feasible sources, Wireless Sensor Network (WSN) becomes one of the vibrant big data sources where a huge volume of data is generated from various sensor nodes in large-scale networks. Compared to traditional networks, WSN faces serious challenges especially in data management and conserving sensor energies. In this work, we propose a novel two phases big data processing mechanism, called ON-IN: on-node and in-node (between nodes). In the first phase, we introduce the Newton's forward difference method to reduce the amount of data generated at each sensor node. Meanwhile, in the second phase we perform a clustering technique, i.e. PKmeans (Pattern-Kmeans) algorithm, and aim to reduce the redundancy among data generated by neighboring nodes. Through both simulations and experiments on real telosB motes, we evaluated the efficiency of our proposed mechanism in terms of reducing data transmission and conserving sensor energies, compared to other existing techniques.**

***Index Terms*—Wireless sensor networks, Newton forward difference method, PKmeans, telosB mote, energy conservation.**

I. Introduction

THE world has witnessed the bursting effects of WSNs as a decisive element in any monitoring process whether in agriculture, medical care, environment or other fields. The large spread and usage of such networks is mainly because of three reasons: their low-cost implementation, their flexibility, and their precision in yielding accurate data. Unfortunately, big data acquisition and transmission energy cost are two major problems that must be handled in order to maximize the lifetime of a sensor. Therefore, data reduction techniques are becoming a fundamental operation to reduce the amount of transmitted data and consequently minimize the energy consumption.

Indeed, the reduction techniques can be either applied at the sensor level itself, e.g. on-node [1]–[5], or at intermediate nodes, e.g. in-node [6]–[9], along the path to the sink. On one hand, the authors of [1] propose an on-node mechanism using the concept of time series analysis in order to analyze the variations in sensed data, so as it can interpreted based on an autoregressive model of order p. On the other hand, the authors of [9] propose an in-node mechanism called Semi Distributed Heuristic Energy efficient Aggregation Tree (SDHEAT) for WSN. Mainly, SDHEAT is based on three concepts: heuristic tree formation, sensing priority and distributed nature. Finally, the authors of [10] a data reduction technique for both sensor and aggregator levels. First, they propose an on-node aggregation method to remove redundant data collected by the sensor. Then, an in-node data reduction called prefix frequency filtering (PFF) is introduced at the cluster-head (CH) level. PFF allows CHs to find similarities among data collected by neighboring nodes in the same cluster using a Jaccard similarity function.

In this paper, we propose a two phases data reduction mechanism dedicated to periodic large-scale sensor network applications. Our mechanism works on two phases: on-node and in-node. The final goal of our mechanism is to reduce data transmission, whether collected by the sensors or transmitted by intermediate nodes, e.g. cluster-head (CH).

The rest of paper is organized as follows. In Section II, we define terminologies and network design. Sections III and IV detail the on-node and the in-node models, respectively. Simulations and experiments are presented in Section V. Section VI concludes the paper.

II. Network Design and Preliminaries

A. Network Design

For energy conservation and scalability reasons, we are interested in the cluster-based architecture for WSN, where the whole network is divided into subareas termed as clusters within which a cluster-head is elected. The CH has a mission to gather data from its subordinate cluster members and sends them to the sink node. Fig. 1 presents the cluster-based network design considered in our study. Thus, our proposed mechanism consists of two phases: on-node and in-node. The on-node operation is performed by the sensors while the in-node is applied at the CH level. After data being periodically collected, the sensor nodes use a data prediction model in order to reduce the data size sent to its corresponding CH. Upon receiving the prediction model for all its sensors, the CH uses a clustering model in order to prevent sending similar prediction data generating by neighboring node to the sink node. Finally, the sink receives the subset of prediction data and try to recover data of the whole sensors.

978-9-0827-9703-9/19/$31.00

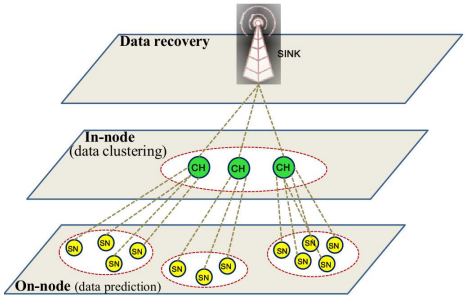


Fig. 1. A cluster-based network architecture

B. Problem Description and Notations

WSN is represented as a connected graph G =($\mathcal{N}$,E), where $\mathcal{N} = \{N_1, N_2, \ldots, N_n\}$ is a set of n (sensor) nodes and E is a set of edges. A sensor node collects data over a period of time (P) and subsequently transmits all the sensed data to the next hierarchy level (e.g. CH). Sensor networks supporting these kind of applications are known as periodic wireless sensor networks (PWSNs). Each period P is divided into a finite number of F time slots as follows: $P = [s_1, s_2, \ldots, s_F]$. At each slot s_j, each sensor node N_i captures a new data value v_{i_j}, and eventually forms a vector of sensed data during the period P as follows: $V_i = [v_{i_1}, v_{i_2}, \ldots, v_{i_F}]$. Mostly, each data vector V_i may contain redundant data (or similar data), especially when the monitored conditions vary slowly or when the frequency of sensing is high or the time slots are short.

III. Sensor Level: On-Node Prediction Model

In PWSN, the huge amount of collected data and its corresponding huge number of transmitted packets lead to two sensor problems: high level of energy consumption and sending unneeded/useless data to the sink. The first phase of our mechanism, e.g. on-node, is applied at the sensor level and aims to prevent sending similar data points sensed at each period P, based on a prediction model using the Newton's forward difference method.

A. Newton's Forward Difference Method

In the mathematical field of numerical analysis, a Newton forward difference is an interpolation polynomial for a given set of data points. It aims to estimate the value of a function ($y_i = f(x_i)$) for any intermediate value of the independent variables (x_i).

Definition 1: Forward Differences. Given a set of $k+1$ data points, $\{(x_0, y_0), (x_1, y_1), \ldots, (x_j, y_j), \ldots, (x_k, y_k)\}$. The differences $y_1 - y_0, y_2 - y_1, \ldots, y_k - y_{k-1}$ denoted by $\Delta y_0, \Delta y_1, \ldots, \Delta y_{k-1}$ respectively are called the first forward differences. Thus the first forward differences are defined by:

$$\Delta y_j = y_{j+1} - y_j, \text{ where } j \in [0, k-1] \tag{1}$$

Based on the above definition, we typically set up the forward difference table as:

x	y	Δy	$\Delta^2 y$	$\ldots$	$\Delta^{c-1} y$	$\Delta^c y$
x_0	y_0					
x_1	y_1	Δy_0	$\Delta^2 y_0$			
x_2	y_2	Δy_1	$\Delta^2 y_1$	$\ddots$	$\Delta^{c-1} y_0$	
$\vdots$	$\vdots$	Δy_2	$\vdots$			$\Delta^c y_0$
x_{k-2}	y_{k-2}	$\vdots$	$\Delta^2 y_{k-3}$		$\Delta^{c-1} y_1$	
x_{k-1}	y_{k-1}	Δy_{k-2}	$\Delta^2 y_{k-2}$	$\iddots$		
x_k	y_k	Δy_{k-1}				

Then, in order to find y-value of a new x-value, we use the Newton's Gregory forward interpolation formula:

$$y = f(x_0 + hu) = y_0 + u\Delta y_0 + \frac{u(u-1)}{2!}\Delta^2 y_0 + \ldots + \frac{u(u-1)(u-2)\ldots(u-c+1)}{c!}\Delta^c y_0 \tag{2}$$

This formula is particularly useful for interpolating the values of $f(x)$ near the beginning of the set of given values. h is called the interval of difference ($h = x_1 - x_0$) and $u = (x - x_0)/h$, where x is the value we want to find its corresponding y.

B. On-Node Prediction Algorithm

The data collected by the sensor nodes during a period, i.e. V_i, are mainly redundant. Thus, in order to prevent sending redundant data to the CH, we propose to integrate the Newton's forward difference method into the sensor processing to reduce the data transmission to the CH. The idea is to find the coefficients of Newton Gregory equation then send them to the CH instead of sending the whole raw data in V_i. Obviously, the data can be regenerated at any time based on the received equation.

The Newton Gregory polynomial needs $k+1$ data points to calculate the equation while the period contains F readings, where F is much bigger than $k+1$. Thus, we propose to select a subset of d data points, named as $\mathcal{D}_i$, from V_i to find the corresponding polynomial. $\mathcal{D}_i$ can be formed based on the following equation:

$$\mathcal{D}_i = \{(s_{1+j\times\lfloor F/d\rfloor}, v_{i_{1+j\times\lfloor F/d\rfloor}}), (s_F, v_{i_F})\} \tag{3}$$

where $v_{i_{1+j\times\lfloor F/d\rfloor}}$ are all readings collected at slot numbers $s_{1+j\times\lfloor F/d\rfloor}$ (such that $j \in [0, F]$ and $1+j\times\lfloor F/d\rfloor < F$) and v_{i_F} is the last reading in V_i.

After selecting the readings, the sensor computes the forward difference table in order to find the needed variables used in the Newton Gregory equation. Then, the sensor will send only the set of $\mathcal{V}_i = \{x_0, x_1, y_0, \Delta y_0, \Delta^2 y_0, \ldots, \Delta^c y_0\}$ which is necessary to recalculate the y values of all readings.

Finally, Algorithm 1 describes the on-node prediction model applied at each sensor node. Briefly, the algorithm takes the period size F as an input for the algorithm. After collecting data readings at each period (lines 1-5), the sensor node selects a set of readings, $\mathcal{D}_i$, from V_i (lines 6-10). Finally, the sensor calculates the final set that will send to its CH based on the forward difference method and Newton Gregory equation (lines 11-12).

Algorithm 1 On-Node Prediction Algorithm.

Require: Node number: N_i, period size: F.
Ensure: Sent set: $\mathcal{V}_i$.
1: $V_i \leftarrow \emptyset$
2: **for** $j = 1$ to F **do**
3: take reading value v_{i_j}
4: $V_i \leftarrow V_i \cup \{v_{i_j}\}$
5: **end for**
6: $\mathcal{D}_i \leftarrow \emptyset$
7: **for** $j = 1$ to F/d **do**
8: $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \{(s_{1+j\times\lfloor F/d\rfloor}, v_{1+j\times\lfloor F/d\rfloor})\}$
9: **end for**
10: $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \{v_{i_F}\}$
11: compute the forward difference table
12: find the variables of $\mathcal{V}_i$

IV. CH Level: In-Node Clustering Model

At a periodic basis, the CH will receive all variable sets coming from all member nodes. Indeed, the spatial-temporal correlation between neighboring sensors can produce a high redundancy between data sets that must be eliminated before sending final data to the sink. At the CH level, we propose to use a clustering approach in order to summarize data coming from the sensor, so that only useful information are sent to the sink.

A. Pattern-Kmeans Algorithm: PKmeans

Data clustering is one of the most important approaches used to classify data. Indeed, Kmeans has been considered as the most popular data clustering algorithms introduced in different domains. The idea behind Kmeans is to classify a number of datasets into K clusters, where the similarity among datasets in the same cluster is high. The process of Kmeans starts by randomly selecting K datasets as the centroids of the clusters, then each dataset is assigned to the nearest centroid using a distance function. After that, the new centroids of the clusters are recalculated and the process is iterated until no more changes in the cluster centroids. Unfortunately, this traditional Kmeans suffers from the computation complexity due the distance calculation, especially when the number of datasets is high and each one contains a large number of values (like the WSN case). This leads to affect the data latency which is an important challenge in WSN, especially in critical applications.

In the literature, one can find many enhancements of Kmeans in order to overcome the data latency problem [11]. In this paper, we propose a new version of Kmeans called Pattern-Kmeans (PKmeans) inspired from the work presented in [11]. PKmeans can largely reduce the computation time of Kmeans and is suitable to WSN applications. After receiving the variable sets from all sensors, PKmeans works based on the following steps:

- The CH regenerates the raw data for each sensor based on the Newton's Gregory equation.
- For each dataset $\mathcal{V}_i$, PKmeans calculates the following statistical parameters: $\mathcal{P}_i = \{Peak, RMS, CrestFactor, Kurtosis, ImpulseFactor, ShapeFactor\}$. Consider that $\mathcal{P}_{i_0}$ corresponds to $Peak$, $\mathcal{P}_{i_1}$ corresponds to RMS and so on.
- PKmeans selects randomly K sets among $\mathcal{P}_i$'s as the initial cluster centroids.
- To assign a dataset to a cluster, PKmeans calculates the Manhattan distance between $\mathcal{P}_i$ and all the cluster centroids.
- Like Kmeans, the process is continue until no more changes in the cluster centroids.

The parameters used in our pattern can be calculated as follows:

$$Peak = \frac{1}{2}\left(max(x_i) - min(x_i)\right) \qquad RMS = \sqrt{\frac{1}{F}\sum_{i=1}^{F}(x_i - \overline{x})^2}$$

$$CrestFactor = \frac{Peak}{RMS} \qquad Kurtosis = \frac{\frac{1}{N}\sum_{i=1}^{N}(s_i - \overline{s})^4}{RMS^4}$$

$$ImpulseFactor = \frac{Peak}{\frac{1}{N}\sum_{i=1}^{N}|s_i|} \qquad ShapeFactor = \frac{RMS}{\frac{1}{N}\sum_{i=1}^{N}|s_i|}$$

Algorithm 2 shows how PKmeans is working out. First, the CH calculates the parameters of $\mathcal{P}_i$ for each set sent $\mathcal{V}_i$ by the sensor. Then, it randomly selects K centroids as the initial centers of the clusters. After that, the Manhattan distance is calculated between every sensor pattern and the cluster centroids while the data sensor is assigned to the nearest one. A loop is done until no change in the cluster centroids. Finally, the nearest data set to the center in order to send to the sink as a representing of the cluster.

Algorithm 2 PKmeans Algorithm.

Require: Sensor datasets: $\mathcal{V} = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_n\}$, number of clusters: K.
Ensure: Set of clusters: $C = \{C_0, C_1, \ldots, C_{K-1}\}$.
1: **for** each set $\mathcal{V}_i \in \mathcal{V}$ **do**
2: // calculate the parameters of pattern $\mathcal{P}_i$
3: **for** each parameter $\mathcal{P}_{i_j} \in \mathcal{P}_i$ **do**
4: calculate $\mathcal{P}_{i_j}$
5: **end for**
6: **end for**
7: randomly choose K centroids G_i ($i \in [0, \ldots, K-1]$) for the clusters
8: $D_{MH} = 0$ // a defined variable for Manhattan distance
9: **repeat**
10: **for** each set $\mathcal{P}_i \in \mathcal{P}$ **do**
11: calculate $D_{MH}(\mathcal{P}_i, G_j)$ where $j \in [0, \ldots, K]$
12: consider $D_{MH}(\mathcal{P}_i, G_m) < D_{MH}(\mathcal{P}_i, G_{m^*})\ \forall\ m^* \in [0, \ldots, K] - [m]$
13: Assign $\mathcal{P}_i$ to the cluster C_m
14: **end for**
15: Update the centroid G_m of each cluster C_m
16: **until** clusters' centroids no longer changes
17: **return** C

V. Performance Evaluation

In order to evaluate the performance of our mechanism, both simulations and real experiments have been conducted.

A. Simulation Results

In this section, we show the results of a set of simulations conducted using the scalar dataset picked up from sensors deployed in the Intel Berkeley Research lab [12]. Table I shows the information about the deployed network. We implemented our technique based on Java simulator and we compare the results to those obtained with PFF [10].

TABLE I
Simulation parameters and their values.

Parameter	Value
Dimension of area	42×33 meters
Number of sensors	46
Observed conditions	temperature[1], humidity, light
Collected readings	2.3 millions
Slot interval	31 seconds

1) Raw Data vs Recovered Data: Fig. 2 shows the performance of on-node phase by recovering raw data collected by the sensors after applying the Newton Gregory equation (referred as NG in the figure). We fixed the period size to 10 readings and we varied the number of selected points (d) to 4, 6 and 8. The results show that on-node phase gives a high data accuracy level compared to the raw data. We can also notice that, the accuracy of the recovered data increases by increasing the number of selected points d. This is because, the accuracy of Newton Gregory formula increases when d increases.

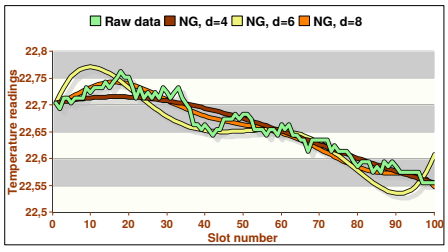
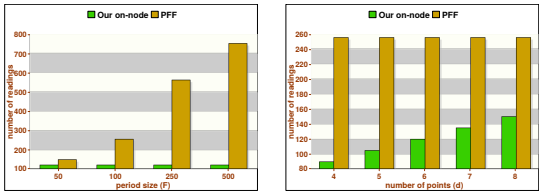


Fig. 2. Comparison between raw and Newton Gregory generated data, $F = 100$

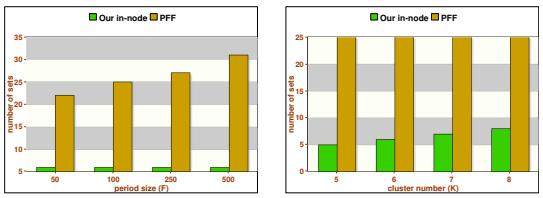
2) Sensor Data Transmission Ratio: This section studies the average number of readings sent from each sensor to the CH (Fig. 3). Compared to PFF technique that uses data aggregation approach, the figure shows that on-node phase gives better results in terms of eliminating redundancy and reducing data transmission to the CH. Subsequently, it reduces the amount of data transmission from 20% to 84% compared to PFF when varying F (Fig. 3(a)), and from 41% to 64% when varying d (Fig. 3(b)). This reduction is because, the sensor node only sends, using on-node phase, the Newton Gregory coefficients to the CH while in the PFF, it uses aggregation method to send a portion of collected data instead of the whole raw data. Therefore, on-node phase will highly minimize the energy consumption in the sensor and increase its lifetime.



(a) $d = 6$ (b) $F = 100$

Fig. 3. Number of readings periodically sent to the CH.

3) CH Data Transmission Ratio: Fig. 4 shows the CH data transmission ratio or the periodic number of sets sent to the sink after applying our in-node phase and PFF. Fig. 4(a) shows the effects of varying the period size F while Fig. 4(b) presents the effects of varying the number of clusters K (from 5 to 8). The obtained results show that in-node phase can successfully reduce the data transmission ratio at the CH, compared to PFF. We notice that in-node phase reduces up to 80% when varying F and K. This confirms the fact that data clustering is an efficient approach to find redundancy among datasets. Therefore, our mechanism can be considered as an energy-efficient technique for both sensor and CH levels.



(a) $K = 6$ (b) $F = 100$

Fig. 4. Periodic number of sets sent to the sink, $d = 6$.

B. Experiment Results

This section shows the results of real data experiments made in our laboratory. We deployed twenty telosB motes in order to collect temperature and humidity data where data are sent to a sink of type SG1000 [13], which it is connected to a laptop machine in order to retrieve and make statistics over the collected data. TelosB uses TinyOS and can be programmed based on nesC language. The sampling rate of all the sensors has been set to 1 reading per 30 seconds while the period size is set to 50 readings. Motes positions in our laboratory are shown in Fig. 5 with IDs ranging from 1 to 20 as well as an ID $= 0$ is assigned to the SG1000.

1) Raw Data vs Recovered Data: In this section, our objective is to show the relevance of on-node phase comparing among simulations and experiments. Similar to Fig. 2, Fig. 6 shows the difference between raw recovered data after applying on-node phase for both temperature and humidity sensors. As expected, the on-node algorithm allows to save a high accuracy level of recovered data. This can be noticed

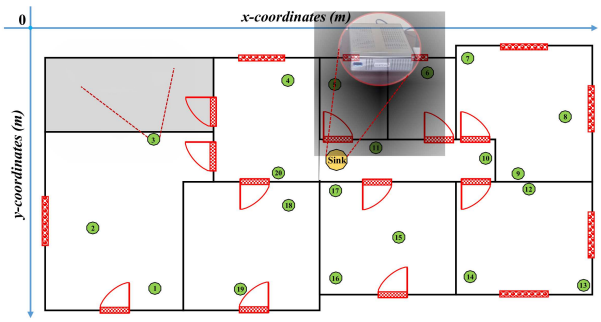


Fig. 5. Distribution of motes in our lab.

through the nearest distance between raw and recovered data at both sensors. Compared to the simulation results (Fig. 2), the experimentations conducted in our lab confirms the behavior of our on-node phase concerning the reducing of data transmission ratio while conserving a high level of information integrity.

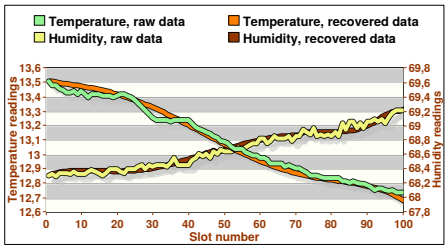


Fig. 6. Comparison between raw and recovered data, $d = 6, F = 100$.

2) Iteration Loop Number: Fig. 7 shows the number of iterations needed by PKmeans algorithm in order to find the final clusters. Obviously, more the number of iterations increases more the packet delivery time to the sink becomes. Thus, data latency will be highly affected. The results show that PKmeans needs approximately 4 iteration loops to finish, in both temperature and humidity. This value is likely acceptable compared to that needed by the traditional Kmeans. Therefore, PKmeans can be considered as an efficient data latency algorithm that seems very suitable to the WSN case.

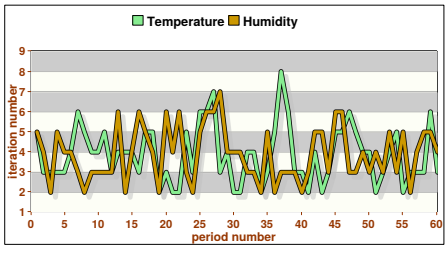


Fig. 7. Number of iterations in applying PKmeans, $F = 100, d = 6, K = 3$.

VI. Conclusion and Future Work

With a constant rise in the importance of WSNs in multiple fields, the need for development of new big data reduction mechanisms is being essential more and more each day. In this manuscript, we proposed an on-node and in-node (ON-IN) mechanism for reducing big data collected in sensor networks. The first phase of our technique focuses on reducing the data transmitted by sensors using the Newton's forward difference method. The second phase focuses on reducing the data generated by neighboring nodes using PKmeans algorithm. The proposed mechanism is evaluated using both simulations and experiments on telosb motes. Our results demonstrated that the proposed mechanism is better than other techniques in terms of data transmission and energy consumption.

As future work, many enhancements can be made on our mechanism. First, other interpolation methods can be developed and implemented on the sensor level to compare for better results. Another direction to follow and study concerns the reduction of the complexity of the PKmeans algorithm to further reduce its data latency.

References

[1] G. Krishna, S. K. Singh, J. P. Singh, and P. Kumar, "Energy conservation through data prediction in wireless sensor networks," in *Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT), Jaipur, India, March 26-27*, 2018, pp. 986–992.

[2] E. Zimos, J. F. Mota, E. Tsiligianni, M. R. Rodrigues, and N. Deligiannis, "Data aggregation and recovery for the internet of things: A compressive demixing approach," in *2018 IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain , April 15-18*. IEEE, 2018, pp. 1–6.

[3] C. Luo, F. Wu, J. Sun, and C. W. Chen, "Efficient measurement generation and pervasive sparsity for compressive data gathering," *IEEE Transactions on Wireless Communications*, vol. 9, no. 12, pp. 3728–3738, 2010.

[4] Q. Xu, R. Akhtar, X. Zhang, and C. Wang, "Cluster-based arithmetic coding for data provenance compression in wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 2018, p. 15 pages, 2018.

[5] S. Kim, C. Cho, K.-J. Park, and H. Lim, "Increasing network lifetime using data compression in wireless sensor networks with energy harvesting," *International Journal of Distributed Sensor Networks*, vol. 13, no. 1, pp. 1–10, 2017.

[6] H. Harb, A. Makhoul, D. Laiymani, and A. Jaber, "A distance-based data aggregation technique for periodic sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 13, no. 4, p. 32, 2017.

[7] S. Ozdemir, M. Peng, and Y. Xiao, "Prda: polynomial regression-based privacy-preserving data aggregation for wireless sensor networks," *Wireless communications and mobile computing*, vol. 15, no. 4, pp. 615–628, 2015.

[8] S. M. Al-Tabbakh, "Novel technique for data aggregation in wireless sensor networks," in *2017 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), Gafsa, Tunisia, October 20-22*. IEEE, 2017, pp. 1–8.

[9] D.-g. Zhang, T. Zhang, J. Zhang, Y. Dong, and X.-d. Zhang, "A kind of effective data aggregating method based on compressive sensing for wireless sensor network," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, p. 15, 2018.

[10] J. M. Bahi, A. Makhoul, and M. Medlej, "A two tiers data aggregation scheme for periodic sensor networks." *Ad Hoc & Sensor Wireless Networks*, vol. 21, no. 1-2, pp. 77–100, 2014.

[11] T. W. Rauber, E. M. do Nascimento, E. D. Wandekokem, and F. M. Varejao, "Pattern recognition based fault diagnosis in industrial processes: Review and application," in *Pattern Recognition Recent Advances*. IntechOpen, 2010.

[12] M. Madden, "Intel berkeley research lab," *http://db.csail.mit.edu/labdata/labdata.html*, 2004.

[13] Advanticsys, "Online data," *http://www.advanticsys.com/wiki/index.php?title=sg1000*.