IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 22, 2018, accepted September 17, 2018, date of publication October 1, 2018, date of current version October 25, 2018.

*Digital Object Identifier 10.1109/ACCESS.2018.2872687*

# A Real-Time Massive Data Processing Technique for Densely Distributed Sensor Networks

**HASSAN HARB[1], ABDALLAH MAKHOUL[2], AND CHADY ABOU JAOUDE[1]**
[1]TICKET Lab, Faculty of Engineering, Antonine University, Baabda, Lebanon
[2]FEMTO-ST Institute/CNRS, DISC Department, Université Bourgogne Franche-Comté, Belfort, France

Corresponding author: Abdallah Makhoul (abdallah.makhoul@univ-fcomte.fr)

This work was supported in part by the EIPHI Graduate School under Contract ANR-17-EURE-0002 and in part by the Hubert Curien CEDRE Program under Grant 40283YK.

**ABSTRACT** Today, we are awash in a flood of data coming from different data generating sources. Wireless sensor networks (WSNs) are one of the big data contributors, where data are being collected at unprecedented scale. Unfortunately, much of these data are of no interest, meaningless, and redundant. Hence, data reduction is becoming fundamental operation in order to decrease the communication costs and enhance data mining in WSNs. In this paper, we propose a two-level data reduction approach for sensor networks. The first level operated by the sensor nodes consists of compressing collected data while using the Pearson coefficient. The second level is executed at intermediate nodes (e.g., aggregators, cluster heads, and so on). The objective of the second level is to eliminate redundant data generated by neighboring nodes using two adapted clustering methods: EKmeans and TopK. Through both simulations and real experiments on real telosB sensors, we show the relevance of our approach in terms of minimizing the big data collected in WSNs and enhancing network lifetime, compared to other existing techniques.

**INDEX TERMS** Wireless sensor network (WSN), sensory data processing, clustering techniques, big-data sensing, data compression.

## I. INTRODUCTION

The rapid proliferation of connected devices and Wireless Sensor Networks (WSN) has given rise to various concepts that integrate the physical world with the virtual one. A vision in which billions of smart objects are linked together, thus enabling anytime, any place connectivity for anything and not only for anyone. With the growth of the number of participants in the future Internet of things, the data volumes collected and transmitted will increase significantly which makes the traditional process of collecting and processing the data insufficient. Consequently, the amount of data should be reduced in order to allow decision makers to mine and analyze this massive data.

Indeed, wireless sensor network is becoming one of the most contributors in big data in this era. Such networks contain hundreds or thousands of sensors that are deployed in a remote zones in order to send periodic information, about the monitored zone, to a sink node. WSNs enable various types of applications including environmental monitoring (climatic change, pollution, water quality) [1], [2], military surveillance (tracking the enemy movements, force protection) [3], [5], agriculture surveillance (precision, food production, plant growing) [6], [7], disaster monitoring (volcanic, seismic, tsunamic) and healthcare monitoring (vital signs, temperature) [8], [9]. These applications collect zettabytes of data everyday most of which are redundant, and useless. Therefore, reducing the amount of redundant data increases the energy consumed by the network and deliver cleaned data to data scientist. Especially, in networks like WSN which suffers from the fact that nodes have limited, and mostly non rechargeable, energy batteries which affects the network lifetime. Indeed, the energy consumption in the network is highly related to the amount of data transmitted [4], [10]. This requires that data needs to be fused along the path to the sink which can effectively reduce the total energy loss in the process of transmission.

In this paper, we study a two-level data reduction technique for minimizing big data collected in sensor network. We consider that our networks is composed of ordinary sensor nodes, intermediate nodes (e.g. aggregator), and the based station (sink). Each sensor node is assigned to an aggregator, and each aggregator receives the periodic collected data from several sensors which in his turn sends it to the sink after data processing (Fig. 1). The first level is done by the sensor

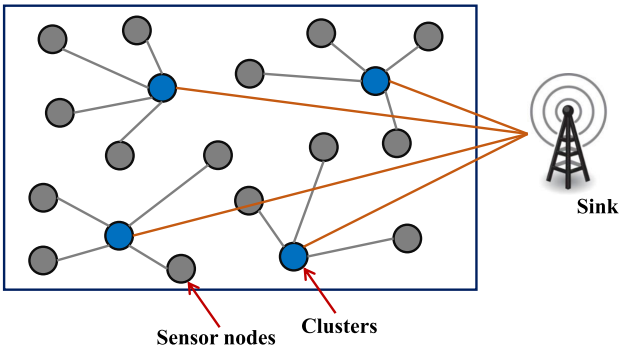


**FIGURE 1.** The proposed network scheme.

nodes themselves. It is an extension of our work [21] and allows each sensor node to periodically compress its data collected using the Pearson coefficient metric, before sending them to the aggregator. The second level is executed by the aggregators. It is an extension of our work [20] and consists on eliminating redundant data generated by neighboring nodes while using two clustering algorithms, Kmeans and TopK. The objective of this paper is to combine and adapt these two techniques together in order to propose a complete framework for data reduction in WSNs. Simulations and real experiments are presented to validate the performance and show the efficiency of the proposed approach.

The remainder of this paper is organized as follows. Section II gives an overview about existing data compression and clustering techniques in WSNs. Section III presents the data compression model based on the Pearson parameter. Section IV describes the two data clustering methods, Kmeans and TopK, to be executed by the aggregators. Simulation and experimentations on real sensors are described in Section V. The last section concludes the paper and give some directions for future work.

## II. RELATED WORK

In the literature, a huge number of researches have been proposed for data compression and data clustering in WSNs. The idea behind these approaches is to reduce the amount of data collected at the source nodes and fusing data of neighboring nodes along the path to the sink. Razzaque *et al.* [11] and Ambekari and Sirsikar [12] show an overview about various data compression and clustering techniques proposed recently by researches in sensor networks.

At the first side, compression methods have an objective to minimize data transmission by encoding data at nodes and decoding them at the sink [13]–[15]. Dedicated to underwater WSNs, a compressed data reduction technique is proposed in [16] consisting of two layers: compressed sampling and data reduction. After forming clusters, the first layer randomly selects a number of nodes for conducting sampling. Whilst, the second layer proposes a full sampling technique in order to minimize the entire energy consumed during data transmission. In [17], an efficient and robust compression method is proposed, Sequential Lossless Entropy Compression (S-LEC). S-LEC uses a differential predictor that arranges the alphabet of integer residues into a number of groups. Subsequently, S-LEC assigns two codes to each group: entropy code and binary code. The first code specifies the group where the second one represents the index inside the group. In [18], the proposed model uses spatial node clustering as well as the principal component analysis (PCA) in order to compress the collected data. In a first step, the authors group sensors with a strong correlation into clusters using novel similarity metrics like magnitude and trend. Then, the authors propose an adaptive strategy for the selection of cluster heads. Lastly, PCA is applied at the cluster heads with a predefined compression error in order to maintain the variance the collected data. Finally, the selected cluster heads apply principal component analysis with an error bound guarantee to compress the data and retain the definite variance at the same time. In [19] and [22], data compression and encryption are combined together in order to keep secure data after compressed and before sending them. First, Gaeta *et al.* [19], the authors propose a Fuzzy-transform (F-transform) compression method based on the discrete wavelet transform model. Then, in [22], an encryption layer called B-spline is added in order to encrypt data before sending to the sink.

At the other side, clustering is one of the effective methods which are applied in WSNs to preserve the battery power of sensor nodes [23]–[27], [42], [43]. In [28], a Distributed K-mean Clustering (DKC) method has been proposed for WSN. The idea behind DKC is to aggregate data based on the adaptive weighted allocation. DKC algorithm tries to eliminate data redundancy as much as closer to the sensor nodes in order to avoid the overloading of the network. In [29], the authors propose transmission-efficient technique dedicated to periodic clustering underwater WSNs. Each sensor node aggregates its similar data in order to clean them before sending to aggregator. Upon receiving the data, the aggregator uses K-means algorithm adopted to ANOVA model with statistical tests. The final goal of such technique is to eliminate redundancy within and between nodes. Bahi *et al.* [30] propose a two-level scheme called prefix frequency filtering (PFF) technique dedicated to periodic sensor applications. PFF divides the whole network into clusters where for each cluster a cluster-head (CH) is assigned. Then, PFF allows each CH to detect the similarities between data collected by neighboring nodes using Jaccard similarity function. Lastly, an associated sensor pattern tree (ASP-tree) is proposed in [28]. ASP-tree uses data mining algorithm and pattern growth-based approach in order to generate all associated patterns with only one scan over dataset.

Unfortunately, most of the proposed techniques present drawbacks. First, they are very complex and require huge processing. Second, they need additional communication when initializing the proposed methods and detecting node failures. In this work which is an extension of our previous work [20], [21], we introduce a new data reduction method

that it is less complex and suitable for sensor nodes with limited resources. Then, in order to show the relevance of our proposed approach, we conducted both and experiments on a real testbed networks based on telosB nodes.

## III. DATA COMPRESSION AT THE SENSOR LEVEL

As mentioned before, data transmission is highly cost operation in WSNs. Thus sending all collected data will quickly deplete the batteries. Hence, in this section we present a new data compression method to reduce the amount of data transmitted in the network. We consider that each sensor $S$ collects a vector of $\tau$ readings, e.g. $R = [r_1, r_2, \ldots, r_\tau]$, during each period then it sends it toward the sink at the end of the period (Fig. 2).

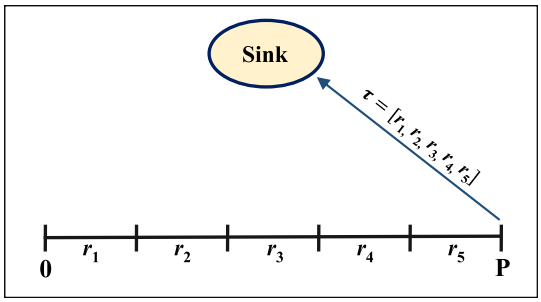


**FIGURE 2.** Periodic data transmission model.

In WSN applications, It is very likely that a sensor node collects redundant reading, especially when the monitored condition varies slowly. Therefore, our objective at this phase is to send selective readings instead of sending the whole vector $R$. Our model is based on the Pearson coefficient. This coefficient represents the degree of correlation between two data sets $R_i$ and $R_j$. Given the interval $[-1, 1]$, a positive correlation is indicated when the Pearson coefficient is equal to 1, a no correlation is indicated when it is equal to 0, and $-1$ indicates a negative Pearson correlation.

Indeed, the Pearson's coefficient between two sensor data sets is represented by the following equation:

$$\rho_{R_i,R_j} = \frac{n\sum r_i r_j - \sum r_i \sum r_j}{\sqrt{n\sum r_i^2 - (\sum r_i)^2}\sqrt{n\sum r_j^2 - (\sum r_j)^2}} \quad (1)$$

where $r_i \in R_i$, $r_j \in R_j$ and $n$ is the number of readings in each of $R_i$ or $R_j$.

Therefore, $R_i$ and $R_j$ are considered to be highly correlated (e.g. redundant) if and only if:

$$\rho_{R_i,R_j} < t_p \quad (2)$$

where $t_p$ is a threshold determined by the application itself.

### A. DATA COMPRESSION AND READINGS SELECTION

This section shows the algorithm used to compress the vector of readings collected by each sensor at each period. The main idea as presented in Algorithm 1, is to find, for each sensor, a subset of readings that represent the whole vector/set $R$ by applying recursively the coefficient of Pearson. It continues dividing $R$ into equal subvectors by applying Pearson's coefficient until finding highly correlated ones (function *DIVIDE*). Therefore, the process starts by considering that the readings in $R$ are not correlated (lines 4-7). Then, $R$ is divided into two subvectors, e.g. $R_{i_1}$ and $R_{i_2}$ (line 9), and the correlation between them is calculated (line 10). If the correlation is less than the threshold of Pearson's coefficient (line 10) then, the initial vector $R_i$ is a final vector of readings. Then, the mean of the readings in $R_i$ is computed and assigned to the vector $V_R$ with its weight (lines 11-13). The weight of the mean value is the number of readings in $R_i$ (line 12). This is repeated until the end of $R$ (line 16).

After applying Algorithm 1, each sensor will send a vector of representative readings $V_{R_i} = [\overline{r}_1, \overline{r}_2, \ldots, \overline{r}_k]$ to its proper aggregator, where $k \leq \tau$.

## IV. CLUSTERING DATA MODELS FOR AGGREGATOR LEVEL

Our data clustering technique based on Kmeans and TopK nearest algorithms, is proposed to eliminate redundant data sets. This is applied at the aggregator level on the received datasets from the members (sensor nodes). This allows similar data sets to be grouped at the same cluster thus, the aggregator will eliminate redundancy before sending them to the sink node.

**Algorithm 1** Data Compression Algorithm

**Require:** Reading vector: $R = [r_1, r_2, \ldots, r_\tau]$.
**Ensure:** Vector of representative readings of $R$: $V_R$.

1: $V_R \leftarrow \emptyset$
2: $V' \leftarrow \emptyset$ // a temporary set of reading vectors
3: $R_1 \leftarrow \emptyset$
4: **for** each reading $r_i \in R$ **do**
5: $\quad R_1 \leftarrow R_1 \cup \{r_i\}$
6: **end for**
7: $V' \leftarrow V' \cup \{R_1\}$
8: **repeat**
9: $\quad \{R_{i_1}, R_{i_2}\} \leftarrow DIVIDE(R_i)$
10: $\quad$ **if** $\rho_{R_{i_1},R_{i_2}} < t_p$ **then**
11: $\quad\quad$ find the mean value, $\overline{r}_i$, of readings in $R_i$
12: $\quad\quad wgt(\overline{r}_i) = R_i.length$
13: $\quad\quad V_R \leftarrow V_R \cup \{\overline{r}_i, wgt(\overline{r}_i)\}$
14: $\quad\quad$ remove $R_i$ from $V'$
15: $\quad$ **else**
16: $\quad\quad V' \leftarrow V' \cup \{R_{i_1}\} \cup \{R_{i_2}\}$
17: $\quad$ **end if**
18: **until** no reading vector $R_i \in V'$
19: return $V_R$

### A. EKmeans CLUSTERING

In this section we present Ekmeans Algorithm for data clustering. It is a combination between classik k-means and the Euclidean distance applied to sensory readings. Kmeans algorithm is based on the concept of classifying/grouping data sets

into $K$ clusters using the set means. As a result, the similarity between sets in the same cluster is high while the similarity between those in different clusters is low. It is known that the Kmeans algorithm is highly dependent on the randomly initial cluster centroids.

#### 1) EUCLIDEAN DISTANCE APPLIED TO SENSORY DATA

Assigning data sets to the nearest cluster centroid is a fundamental process when applying Kmeans algorithm. To do this, we propose to use distance functions (e.g. Hamming, Cosine, Euclidean, etc.) to calculate the similarity and distance between datasets/vectors. In our approach we will use the Euclidean distance function.

The Euclidean distance is the simple form of distance in mathematics representing the straight line between two points, sets or objects. Assume two sets of data, $R_i$ and $R_j$, then the Euclidean distance ($E_d$) between them is given in the following formula:

$$E_d(R_i, R_j) = \sqrt{\sum (r_i - r_j)^2}, \tag{3}$$

where $r_i \in R_i$ and $r_j \in R_j$.

However, the weights of the mean values used at the sensor level makes the computation of the Euclidean distance not easy. Therefore, we should transform each set of representative readings $V_{R_i}$ (respectively $V_{R_j}$) to a vector as follows:

$$v_{R_i} = \big[\underbrace{\overline{r}_1, \ldots, \overline{r}_1}_{wgt(\overline{r}_1)\text{ times}}, \underbrace{\overline{r}_2, \ldots, \overline{r}_2}_{wgt(\overline{r}_2)\text{ times}}, \ldots, \underbrace{\overline{r}_k, \ldots, \overline{r}_k}_{wgt(\overline{r}_k)\text{ times}}\big]. \tag{4}$$

Then, the Euclidean distance between $V_{R_i}$ and $V_{R_j}$ is calculated based on their readings vectors $v_{R_i}$ and $v_{R_j}$.

#### 2) EKmeans ALGORITHM

To classify the datasets received by the aggregator into clusters of similarity we propose the kmeans Algorithms. It is a combination of the classical Kmeans algorithm and the Euclidean distance as shown in Algorithm 2. The clustering process starts when the aggregator receives the sensor data sets at the end each period. First, $K$ sets of data are randomly selected to be the centers of clusters (lines 4-6). Then, for later iterations, the distances between every data set and all centers are calculated where the aggregator assigns the set to the cluster with nearest center (lines 8-10). After that, the new centroid for every cluster is calculated, and the algorithm restarts until no more changes in the cluster members (lines 11-13).

**Algorithm 2** EKmeans Algorithm

**Require:** Set of representative reading sets $V_R = \{V_{R_1}, V_{R_2}, \ldots, V_{R_n}\}, K$.
**Ensure:** Set of clusters $C = \{C_1, C_2, \ldots, C_K\}$.
1: **for** $j \leftarrow 1$ to $K$ **do**
2: $C_j \leftarrow \emptyset$
3: **end for**
4: **for** $j \leftarrow 1$ to $K$ **do**
5: randomly choose centroid $x_j$ among $V_R$ belongs to $C_j$
6: **end for**
7: **repeat**
8: **for** each set $V_{R_i} \in V_R$ **do**
9: Assign $V_{R_i}$ to the cluster $C_j$with nearest $x_i$ (i.e., $E_d(V_{R_i}, x_{j*}) \leq E_d(V_{R_i}, x_j); j \in \{1, \ldots K\}$)
10: **end for**
11: **for** each cluster $C_j$, *where* $j \in \{1, \ldots K\}$ **do**
12: Update the centroid $x_i$ to be the centroid of all sets currently in $C_j$, so that $x_j = \frac{1}{|C_j|} \sum_{i \in C_j} v_{R_i}$
13: **end for**
14: **until** no more changes in the centers of clusters
15: return $C$

### B. TOPK NEAREST NEIGHBORING ALGORITHM

TopK nearest neighbors [34] is one of the top 10 data mining algorithms used for classification and regression. It is considered as a non-parametric test that does not assume any hypothesis about the normality of the data. TopK algorithm has lots of applications ranging from business [35] and medical [36] to classification of web text [37]. The input of TopK algorithm consist of the whole training dataset. Subsequently, in order to search the similarities of a new data instance, TopK algorithm calculates the distance between the new data instance and all datasets in the training dataset. Then, it returns the $K$-most similar instances that having the minimum distance to the new instance. Usually, the TopK algorithm uses distance functions to search the $K$-nearest neighbors for a dataset. In our proposal we will use the Euclidean distance as presented before. On the other hand, the selection of the value of $K$ parameter is very crucial in the TopK algorithm, which is a user-defined constant. In general, the classification will be more accurate when the value of $K$ increases. Heuristic techniques are one of the approaches used to select the proper value of $K$ which is determined by the experts. Another way for the selection of $K$ is by experimenting different values of $K$ (e.g. values from 1 to 20) and see which works best for our problem, i.e. the most accurate results. Indeed, the optimal value of $K$ for many studied applications varied in the interval [3, 10].

Algorithm 3 describes the process of TopK algorithm to search the top $k$ similar datasets for a new dataset given as an input for the algorithm. The process starts by computing the Euclidean distance between the new dataset and every dataset in the training set $R$ (line 3). Thus, a dataset is added to the final list of top $K$ similar sets of the new set if the list is not yet full (line 4) or its distance to the new dataset is less than the maximum of an existing distance (line 7-10).

#### 1) SELECTING FINAL DATASETS

In this section, we show how to integrate the TopK nearest neighbor algorithm at the aggregator level in order to search, then eliminate, redundant datasets sent at the end of each period (Algorithm 4). First, the aggregator identifies the top $K$

**Algorithm 3** TopK Nearest Neighbors Algorithm

**Require:** List of datasets $R = \{R_1, R_2, \ldots, R_n\}$, new dataset $R_j$, $K$.
**Ensure:** List of top $K$ similar datasets to $R_j$: $TopK_{R_j}$.
1: $TopK_{R_j} \leftarrow \emptyset$
2: **for** each dataset $R_i \in R$ **do**
3: compute $distance = E_d(R_i, R_j)$
4: **if** $TopK_{R_j}.length < K$ **then**
5: $TopK_{R_j} \leftarrow TopK_{R_j} \cup \{(R_j, R_i, distance)\}$
6: **else**
7: find $R_l \in TopK_{R_i}$ corresponding to the maximum distance with $R_j$
8: **if** $E_d(R_l, R_j) > E_d(R_i, R_j)$ **then**
9: replace $R_l$ by $R_i$
10: **end if**
11: **end if**
12: **end for**
13: **return** $TopK_{R_j}$

similar sets for each dataset sent by a sensor (lines 3-5) using Algorithm 3. It aims to find the top $K$ sensors that generate similar data, in terms of temporal correlation, to every sensor in the network. Therefore, data transmission size sent to the sink node will be decreased. Lastly, the aggregator deletes pairs of similar datasets containing either $V_{R_i}$ or $V_{R_j}$ from the pair set (i.e. don't check again) (line 8).

**Algorithm 4** Removing Redundant Datasets Algorithm

**Require:** List of representative reading sets $V_R = \{V_{R_1}, V_{R_2}, \ldots, V_{R_n}\}$, $K$.
**Ensure:** List of sent reading sets: $V_L$.
1: $V_L \leftarrow \emptyset$
2: $topk \leftarrow \emptyset$
3: **for** each set $V_{R_i} \in V_R$ **do**
4: $topk \leftarrow topk \cup TopK(V_R - \{V_{R_i}\}, V_{R_i})$
5: **end for**
6: **for** each pair of sets$(V_{R_i}, V_{R_j}) \in topk$ **do**
7: $V_L \leftarrow V_L \cup \{V_{R_i}\}$ // or $V_L \leftarrow V_L \cup \{V_{R_j}\}$
8: Delete all pairs of sets that contain one of the two sets $V_{R_i}$ and $V_{R_j}$
9: **end for**
10: **return** $V_L$

## V. PERFORMANCE EVALUATION

We introduce, in this section, the setup used to validate the relevance and the efficiency of our proposal. We performed two types of evaluation, e.g. simulation and real experiment, in order to show the behavior of our technique in different environments. In the first environment, data of 54 sensors deployed in Berkeley lab [38] are used to simulate our technique. By varying their values, this evaluation allows decision makers to select the best parameter values for a given application. The second environment uses crossbow teloB nodes deployed in our laboratory. We aim to conducts real experiments in order to compare the behavior of our technique in real-world and simulation environment. The effectiveness of our technique at the sensor level is tested and compared to a data compression technique (S-LEC) proposed in [17] while, at the aggregator level, our results are compared to a data reduction technique (PFF) proposed recently in [30].

### A. SIMULATION RESULTS

This section shows the simulation we conducted on data collected in the Intel sensor network. In such network, 54 Mica2dot sensors are deployed in the lab for approximately two months collecting more than 2 millions of readings about weather conditions (temperature, humidity and light). Sensor sampling rate is fixed to 2 readings per minutes. The positions of sensors inside the lab are shown in Fig. 3 (yellow sign indicates the dysfunction of some sensors). For simplicity reason, we show in this section the results of temperature condition. Moreover, we simulate an aggregator node located at the middle of lab in order to collect data from all sensors. Table 1 shows the parameters used in the simulations.

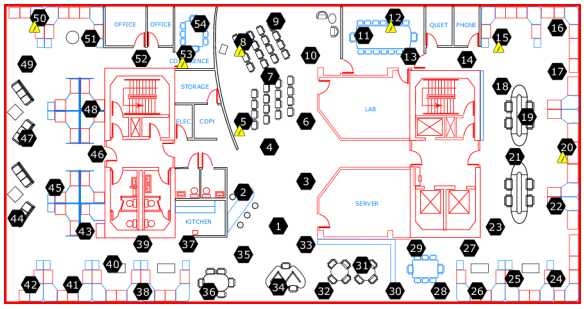


**FIGURE 3.** Sensors deployed in Intel lab network.

**TABLE 1.** Simulation environment.

| Parameter | Description | Value |
|---|---|---|
| $\tau$ | period size | 100, 200, 500, 1000 |
| $t_p$ | Pearson's threshold | 0.4, 0.5, 0.6, 0.7 |
| $K$ | number of clusters | 4, 6, 8, 10 |

#### 1) COMPRESSION RATIO AT EACH SENSOR

The proposed Pearson coefficient model allows sensors to periodically minimize its data transmission by compressing redundant data. Indeed, the compression ratio is highly dependent on the selection of Pearson threshold ($t_p$) and the period size ($\tau$). Fig. 4 shows the data compression ratio indicating the number of representative readings after applying Pearson coefficient at each sensor. The results are compared to the compression method S-LEC. We notice that our technique allows sensors to reduce their data transmission by at least 75% and up to 89% compared to S-LEC. In addition, we observe that data eliminated using our technique increases

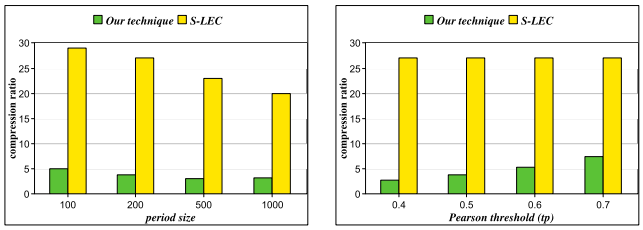


**FIGURE 4.** Compression ratio at each sensor. (a) $t_p = 0.5$. (b) $\tau = 200$.

when the period size increases. This is because, more data will be redundant when $\tau$ increases.

### 2) DATA TRANSMISSION VARIATION DURING PERIODS

The information integrity and the accuracy of the transmitted data is an important metric in WSNs. Hence, which readings are selected to be transmitted to the sink is a critical step in our method because the enduser decision could be affected. Subsequently, if data transmitted does not selected in an efficient manner, some important measures could be lost. Furthermore, more data are selected more the redundancy is existing (thus taking the right decision will be more complicated). By comparison to the naïve method, Fig. 5 shows that our technique allows each sensor to select a subset of useful/non-redundant readings, with their corresponding weights, to send to the sink. Moreover, we observe that number of representative reading varies depending on the changes on the monitored condition (Fig. 5(b)).

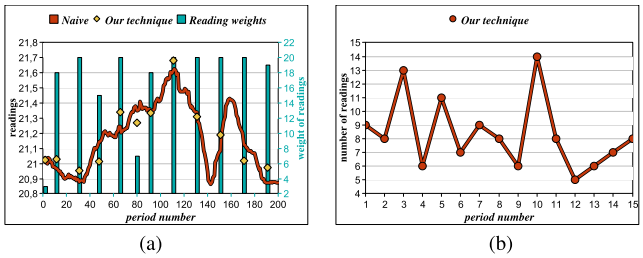


**FIGURE 5.** Variation of number of transmitted readings during periods, $\tau = 200$, $t_p = 0.5$. (a) within one period. (b) in all periods.

### 3) DATASETS TRANSMISSION FROM AGGREGATOR TO SINK

In our technique, the aggregator will periodically receive sets of representative readings coming from all sensor nodes. After searching redundancy between them using EKmeans or TopK algorithms, the aggregator selects some of them to be sent to the sink instead of the whole received sets. Fig. 6 shows the number of transmitted sets from the aggregator node to the sink at each period using the clustering algorithms, EKmeans and TopK, and the PFF technique. The obtained results show that EKmeans outperforms TopK and PFF in terms of eliminating redundancy and sending less number of sets to the sink. Subsequently, we observe that EKmeans can reduce from 78% to 91% of the whole received sets while TopK and PFF can reduce from 65% to 82% and from 23% to 45% respectively. These results confirm that the

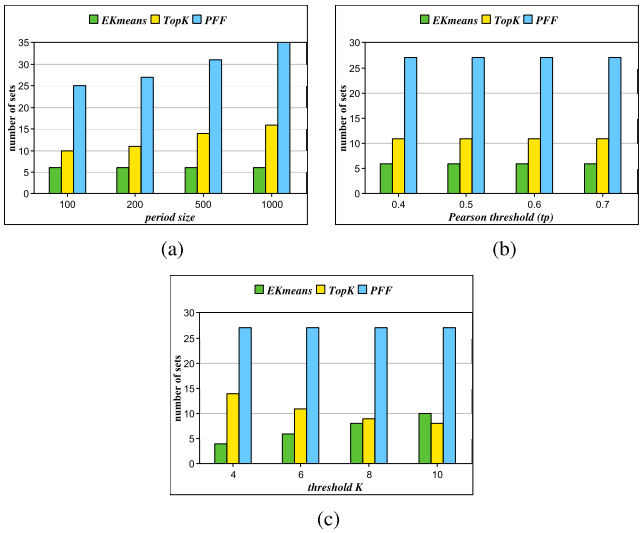


**FIGURE 6.** Number of sets sent periodically from aggregator to sink. (a) $t_p = 0.5$, $K = 6$. (b) $\tau = 200$, $K = 6$. (c) $\tau = 200$, $t_p = 0.5$.

clustering is a very efficient approach in terms of eliminating redundant data and providing useful information to the enduser, comparing to other existing approaches. On the other hand, the result confirms the behavior of our technique; the number of transmitted sets to the sink in EKmeans is equal to the number of selected clusters ($K$) (see Fig. 6(c)) while, in topK and PFF, it is dependent on the temporal correlation between the collected data which varies between periods. Finally, Fig. 6(b) shows that the variation of Pearson's threshold used at the sensor level does not affect the results at the aggregator level; thus, the effectiveness of clustering techniques, e.g. EKmeans and TopK, is independent on the selected Pearson threshold value.

Obviously, the energy consumption in sensor networks is highly related to the volume of transmission data; more data are sent more the sensor energy is wasted. In our simulation, we implemented the same energy model that used in [38] to calculate the energy consumption in the network. The proposed model computes the energy consumption in the aggregator when it receives data from sensors as well as sending them to the sink. Fig. 7 shows how the energy consumed in aggregator varies depending on the period size (Fig. 7(a)), Pearson threshold (Fig. 7(b)) and the number of clusters (Fig. 7(c)). The obtained results show that the energy consumption increases with the increasing of the period size or the Pearson threshold. This is because, the redundancy between datasets will decrease when $\tau$ or $t_p$ increases. Therefore, our proposed technique can be considered very efficiently in terms of reducing the network energy consumption, thus, increasing its lifetime.

### 4) PROCESSING TIME AT AGGREGATOR

Sometimes, delivering data as fast time as possible to the enduser is a crucial operation especially in e-health and military applications. Fig. 8 shows the processing time when

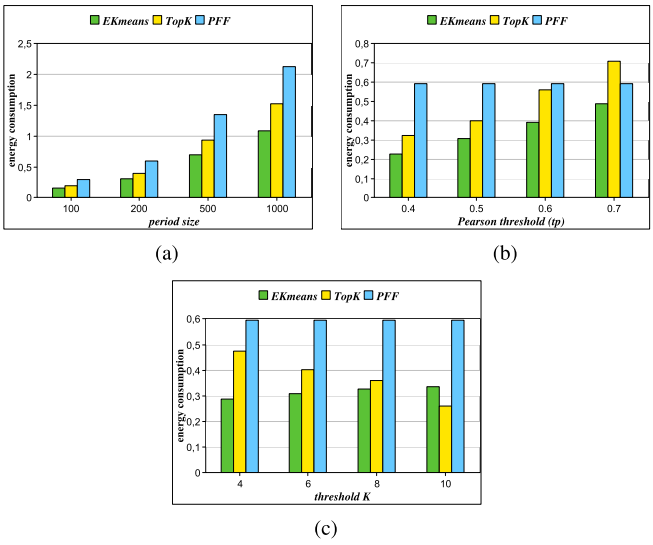


**FIGURE 7.** Energy consumption in CH. (a) $t_p = 0.5$, $K = 6$. (b) $\tau = 200$, $K = 6$. (c) $\tau = 200$, $t_p = 0.5$.

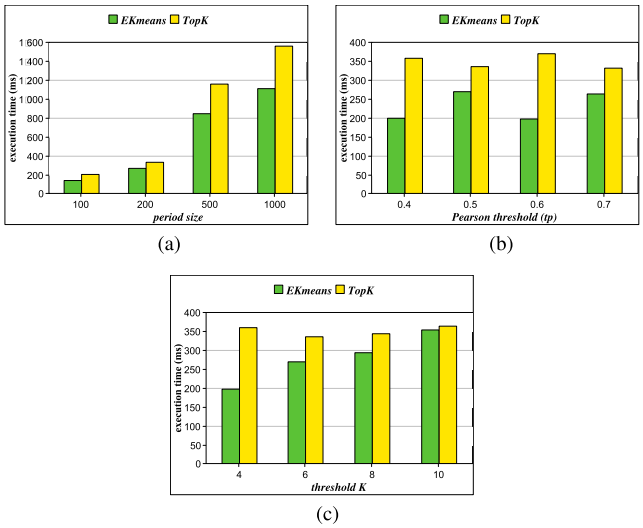


**FIGURE 8.** Processing time at the aggregator after applying EKmeans and TopK. (a) $t_p = 0.5$, $K = 6$. (b) $\tau = 200$, $K = 6$. (c) $\tau = 200$, $t_p = 0.5$.

applying data clustering algorithms, e.g. EKmeans and TopK, used in our technique. Obviously, the processing time of EKmeans will be highly affected by the random selection of the cluster centroids as well as the number of iteration loops to obtain the final clusters. Whilst, the processing time of TopK algorithm depends on the selection of final sets shown in Algorithm 4. From the obtained results, we observe that both techniques efficiently reduce data transmission while do not delay data delivery at the sink. Moreover, we observe that EKmeans outperforms, in all situations, TopK in terms of time processing. This is because, EKmeans searches groups of redundant sets which requires less processing time as searching by pairs that is used in TopK. Consequently, the processing time at the aggregator is twice accelerated when using EKmeans, compared to TopK algorithm.

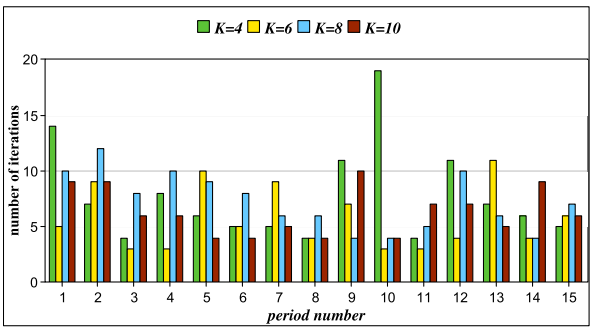


**FIGURE 9.** Number of iteration loops in EKmeans algorithm, $\tau = 200$, $t_p = 0.5$.

### 5) ITERATION LOOPS

One of the factor that can delay the delivery of message is the number of iterations used in the process of EKmeans. In Fig. 9, we show how many iterations are generated by EKmeans at each period. Again, iteration number will be highly affected by the random selection of the centroid clusters. Based on the results, we show that the loops number periodically generated by EKmeans varies between 3 (best case scenario) and 18 (worst case scenario). Thus, this number is considered as a small value independent on the used parameters. Therefore, EKmeans is considered as an efficient clustering method for the limited resources in the aggregator.

### 6) VARIATION OF SET NUMBER AMONG CLUSTERS

In this section, we show how sets are distributed in numbers between the clusters after using EKmeans method along with the period number (Fig. 10). The obtained results show that the sets are distributed in an unequal way into clusters. The behavior of EKmeans is confirmed by classifying data sets based on their dissimilarity and not on an equal distribution.

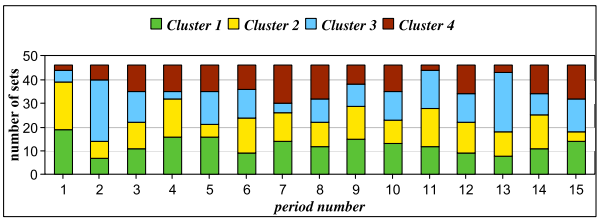


**FIGURE 10.** Number of sets in each cluster during periods, $\tau = 200$, $t_p = 0.5$, $K = 4$.

Therefore, we consider that EKmeans is an efficient data clustering in terms of data classification and data latency.

#### 7) SELECTION OF FINAL REPRESENTATIVE SETS USING TOPK ALGORITHM

This section shows an illustrative example for the selection of the final datasets determined by the aggregator after searching the top $K$ correlated/nearest sensors for each sensor node, during a taken period. The results of Fig. 11 sees that the sensor nodes generate highly correlated data sets. Thus, the aggregator selects a set of data (colored green) to send to the sink. We can also observe that each sensor has a strong correlation to its spatial neighbor nodes than those far in the network. However, the figure also shows that temporal correlation is also seen between distant nodes.

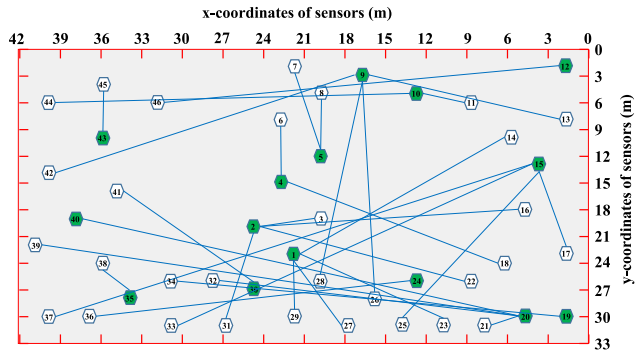


**FIGURE 11.** Example of TopK correlated sensors for each node during a period, $\tau = 200$, $t_p = 0.5$, $K = 4$.

### B. EXPERIMENT RESULTS

In this section, we show the relevance of our proposed technique after performing experiments on real sensor nodes deployed in our laboratory. We used Crossbow telosb motes in order to collect data about the zone. We have deployed twenty motes in our laboratory in order to monitor temperature data. The motes send their collected data to a sink of type SG1000 [36], which it is connected to a laptop machine in order to retrieve and make statistics over the collected data. 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. 12. We assign an ID for each mote starting from 1 to 20 as well

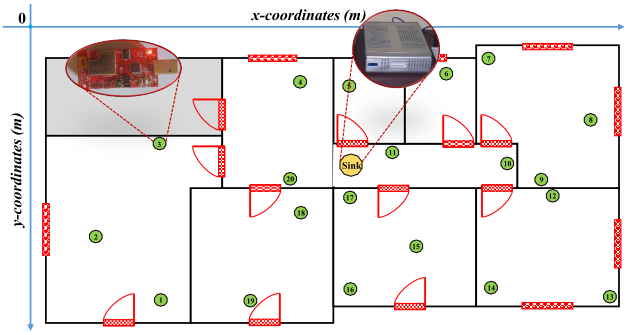


**FIGURE 12.** Distribution of motes in our lab.

as an ID $= 0$ is assigned to the SG1000. Table 2 shows the technique applied at each sensor:

**TABLE 2.** Techniques implemented on the motes.

| Technique | mote IDs |
|---|---|
| our technique | 1, 5, 6, 7, 9, 12, 13, 15, 19, 20 |
| S-LEC | 3, 8, 11, 14, 16, 18 |
| naïve | 2, 4, 10, 17 |

Finally, it must be noticed that all methods were implemented on the motes using the nesC language [40], i.e. the programming language used in tinyOS [41]. In addition, statistics over data received at the sink have been done thanks to Java code running on the laptop machine.

#### 1) COMPRESSION RATIO AT EACH MOTE

In Fig. 13, we show the average amount of transmitted temperature readings for each individual mote, comparison between our technique and S-LEC. We observe that our proposed data compression model makes motes sending less data compared to those operating with S-LEC technique. Subsequently, each mote can reduce up to 50% the temperature readings sent to SG1000. Therefore, these results and those shown in Fig. 4 lead to conclude that our technique is an efficient data reduction approach while its validity is tested on both simulation and real-world environment.

#### 2) REMAINING SETS AFTER APPLYING EKmeans AND TOPK ALGORITHMS

In Fig. 14, we show the average number of remaining sets after applying EKmeans and TopK algorithms at the sink node, when varying the cluster number. The obtained results show a difference to those obtained in the simulation because of the small value of taken $K$ and the highly temporal correlation between the motes in the lab. Subsequently, we observe that, for a small number of clusters (i.e. $\leq 4$), the number of remaining sets after applying EKmeans is less than those obtained after applying TopK. Otherwise, e.g. when the number of clusters increases ($> 4$), TopK algorithm eliminates more redundant sets compared to EKmeans.

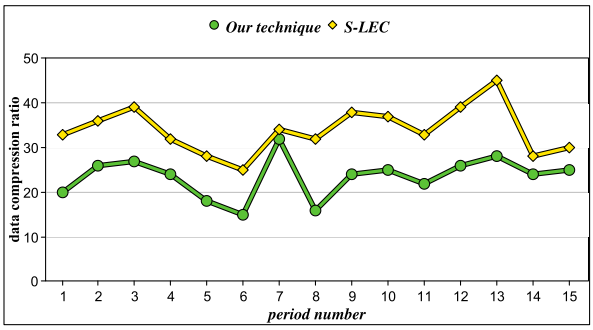


**FIGURE 13.** Compression ratio at each mote during periods, $\tau = 200$, $t_p = 0.5$.

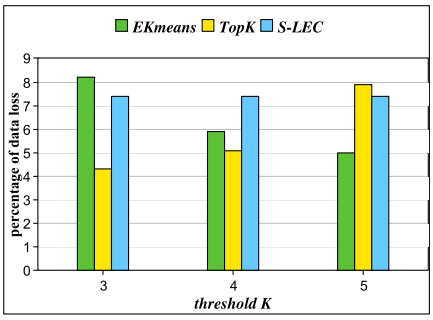


**FIGURE 15.** Percentage of data loss, $\tau = 200$, $t_p = 0.5$.

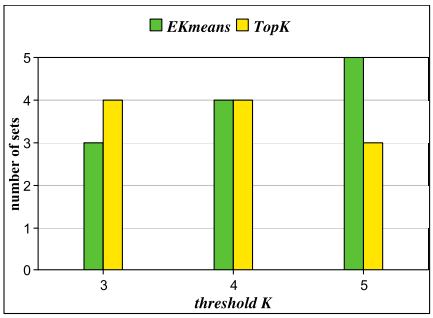


**FIGURE 14.** Sets transmitted to sink in EKmeans and TopK, $\tau = 200$, $t_p = 0.5$.

#### 3) QUALITY OF INFORMATION

Preserving the quality of information is very essential when removing redundant data in WSNs. In our experiments, we calculated the accuracy of the information by dividing the number of loss readings after applying EKmeans and TopK algorithms over the whole readings sensed by the naïve motes. Fig. 15 presents the data accuracy results for both EKmeans and TopK compared to S-LEC technique, when varying the cluster number $K$. It is obvious that the results are linked to the number of remaining datasets after applying EKmeans and TopK (see results of Fig. 14); more the number of remaining sets less the readings are lost. Indeed, we observe that all techniques give important results regarding the accuracy of the collected data where the integrity of the information is highly conserved for the end user. Subsequently, we notice that TopK algorithm gives the best results, in terms of conserving the quality of information, when the number of cluster is small, e.g. $\leq 4$, whilst the information is more conserved using EKmeans when $K$ increases, e.g. $> 4$.

### C. MORE DISCUSSION

In this section, we aim to generalize our comparison between both algorithms EKmeans and TopK. We make the decision makers able to select the best algorithm depending on the monitored zones and the requirements of application.

At the sensor node level, both algorithms EKmeans and TopK can largely extend the sensor node lifetime. However, if the collected data are highly temporal correlated and the number of clusters is large ($\geq$ 10) then TopK gives better results, else, EKmeans gives better results. Therefore, in applications where the decision makers want to ensure a long monitoring of the zone, they have to choose the more suitable algorithm depending on the number of clusters.

Talking about the quality of information, similar conclusions to the energy consumption can be observed. Consequently, when the number of clusters increases EKmeans can save the integrity of the information more than TopK. This is because EKmeans keeps a certain amount of redundancy in the final sent data that leads to increases the quality of information received at the sink. Therefore, in applications where the decision makers want to ensure a high quality of information, EKmeans is more recommended.

Talking about the processing time, EKmeans allows the aggregator to process and send more quickly the information to the sink, compared to TopK algorithm. Subsequently, EKmeans can accelerate the processing time at the aggregator up to twice compared to that required using TopK. This is because, EKmeans searches groups of redundant sets which requires less processing time as searching by pairs that is used in TopK. Consequently, in application where data should be delivered to the sink as much time as possible, EKmeans will be more recommended.

## VI. CONCLUSION AND FUTURE WORK

The exponential growth of the objects connected is producing a large amount of sensory collected data. Thus, the management of the massive data is essential in order to provide real-time and trusted applications. However, due to the network constraints and especially the energy consumption, the processing of huge data remains a big challenge. In this paper, we have proposed a complete framework for data reduction in sensor networks. It is composed of two levels. At the first level the sensor nodes use the Pearson coefficient in order to compress the collected data. Whilst, at the aggregator level, we used two data clustering methods, Kmeans and TopK, in order to eliminate data redundancy among neighboring nodes. Our proposed technique is evaluated through both simulation and experimentations on real telosB sensors. Compared to other existing techniques, the obtained results show the effectiveness of our technique in reducing the big data collected in WSNs and enhancing network lifetime. In a

future work and in order to save more energy and bandwidth in the network, we aim to propose a scheduling strategy allowing sensors generating redundant data to go into sleep mode.

## REFERENCES

[1] T. Alhmiedat, "A survey on environmental monitoring systems using wireless sensor networks," *J. Netw.*, vol. 10, no. 11, pp. 606–615, 2015.
[2] K. S. Adu-Manu, C. Tapparello, W. Heinzelman, F. A. Katsriku, and J.-D. Abdulai, "Water quality monitoring using wireless sensor networks: Current trends and future research directions," *ACM Trans. Sensor Netw.*, vol. 13, no. 1, 2017, Art. no. 4.
[3] T. Azzabi, H. Farhat, and N. Sahli, "A survey on wireless sensor networks security issues and military specificities," in *Proc. IEEE Int. Conf. Adv. Syst. Electr. Technol. (ICASET)*, Hammamet, Tunisia, Jan. 2017, pp. 1–6.
[4] G. B. Tayeh, A. Makhoul, D. Laiymani, and J. Demerjian, "A distributed real-time data prediction and adaptive sensing approach for wireless sensor networks," *Pervasive Mobile Comput.*, vol. 49, pp. 62–75, Sep. 2018.
[5] I. Ahmad, K. Shah, and S. Ullah, "Military applications using wireless sensor networks: A survey," *Int. J. Eng. Sci. Comput.*, vol. 6, no. 6, pp. 7039–7043, 2016.
[6] J. P P and K. S S, "Wireless sensor network and monitoring of crop field," *IOSR J. Electron. Commun. Eng.*, vol. 12, pp. 23–28, 2017, doi: 10.9790/2834-1201022328.
[7] G. Deepika and P. Rajapirian, "Wireless sensor network in precision agriculture: A survey," in *Proc. Int. Conf. Emerg. Trends Eng., Technol. Sci. (ICETETS)*, Pudukkottai, India, Feb. 2016, pp. 1–6.
[8] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2688–2710, Oct. 2010.
[9] P. Prasad, "Recent trend in wireless sensor network and its applications: A survey," *Sensor Rev.*, vol. 35, no. 2, pp. 229–236, 2015.
[10] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Netw.*, vol. 7, no. 3, pp. 537–568, May 2009.
[11] M. A. Razzaque, C. Bleakley, and S. Dobson, "Compression in wireless sensor networks: A survey and comparative evaluation," *ACM Trans. Sensor Netw.*, vol. 10, no. 1, 2013, Art. no. 5.
[12] J. S. Ambekari and S. Sirsikar, "Comparative study of optimal clustering techniques in wireless sensor network: A survey," in *Proc. ACM Symp. Women Res. (WIR)*, Indore, India, 2016, pp. 38–44.
[13] E. Zimos, D. Toumpakaris, A. Munteanu, and N. Deligiannis, "Multiterminal source coding with copula regression for wireless sensor networks gathering diverse data," *IEEE Sensors J.*, vol. 17, no. 1, pp. 139–150, Jan. 2017.
[14] J. He, G. Sun, Z. Li, and Y. Zhang, "Compressive data gathering with low-rank constraints for wireless sensor networks," *Signal Process.*, vol. 131, pp. 73–76, Feb. 2017.
[15] S. Kim, C. Cho, K.-J. Park, and H. Lim, "Increasing network lifetime using data compression in wireless sensor networks with energy harvesting," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 1, pp. 1–10, 2017.
[16] H. Lin *et al.*, "Energy-efficient compressed data aggregation in underwater acoustic sensor networks," *Wireless Netw.*, vol. 22, no. 6, pp. 1985–1997, 2016.
[17] Y. Liang and Y. Li, "An efficient and robust data compression algorithm in wireless sensor networks," *IEEE Commun. Lett.*, vol. 18, no. 3, pp. 439–442, Mar. 2014.
[18] Y. Yin, F. Liu, X. Zhou, and Q. Li, "An efficient data compression model based on spatial clustering and principal component analysis in wireless sensor networks," *Sensors*, vol. 15, no. 8, pp. 19443–19465, Aug. 2015.
[19] M. Gaeta, V. Loia, and S. Tomasiello, "Multisignal 1-d compression by F-transform for wireless sensor networks applications," *Appl. Soft Comput.*, vol. 30, pp. 329–340, May 2015.
[20] H. Harb, A. Makhoul, and C. A. Jaoude, "En-route data filtering technique for maximizing wireless sensor network lifetime," in *Proc. 14th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2018, pp. 298–303.
[21] H. Harb and C. A. Jaoude, "Combining compression and clustering techniques to handle big data collected in sensor networks," in *Proc. IEEE MENACOMM*, Apr. 2018, pp. 1–6.
[22] M. Gaeta, V. Loia, and S. Tomasiello, "Cubic B–spline fuzzy transforms for an efficient and secure compression in wireless sensor networks," *J. Inf. Sci.*, vol. 339, pp. 19–30, Apr. 2016.
[23] H.-S. Kim, J.-S. Han, and Y.-H. Lee, "Scalable network joining mechanism in wireless sensor networks," *Proc. IEEE Top. Conf. Wireless Sensors Sensor Netw. (WiSNet)*, Santa Clara, CA, USA, Jun. 2012, pp. 45–48.
[24] A. Gachhadar and O. N. Acharya, "K-means based energy aware clustering algorithm in wireless sensor network," *Int. J. Sci. Eng. Res.*, vol. 5, no. 5, pp. 156–161, 2014.
[25] M. Bidaki, R. Ghaemi, and S. R. K. Tabbakh, "Towards energy efficient K-means based clustering scheme for wireless sensor networks," *Int. J. Grid Distrib. Comput.*, vol. 9, no. 7, pp. 265–276, 2016.
[26] B. Malhotra, M. A. Nascimento, and I. Nikolaidis, "Exact top-K queries in wireless sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 10, pp. 1513–1525, Oct. 2011.
[27] W.-H. Liao and C.-H. Huang, "An efficient data storage scheme for top-K query in wireless sensor networks," in *Proc. IEEE Netw. Oper. Manage. Symp.*, Maui, HI, USA, Apr. 2012, pp. 554–557.
[28] P. Zou and Y. Liu, "A data-aggregation scheme for WSN based on optimal weight allocation," *J. Netw.*, vol. 9, no. 1, pp. 100–107, 2014.
[29] H. Harb, A. Makhoul, and R. Couturier, "An enhanced K-means and ANOVA-based clustering approach for similarity aggregation in underwater wireless sensor networks," *IEEE Sensors J.*, vol. 15, no. 10, pp. 5483–5493, Oct. 2015.
[30] J. M. Bahi, A. Makhoul, and M. Medlej, "A two tiers data aggregation scheme for periodic sensor networks," *Ad Hoc Sensor Wireless Netw.*, vol. 21, nos. 1–2, pp. 77–100, 2014.
[31] M. M. Rashid, I. Gondal, and J. Kamruzzaman, "Mining associated patterns from wireless sensor networks," *IEEE Trans. Comput.*, vol. 64, no. 7, pp. 1998–2011, Jul. 2015.
[32] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques* (Data Management Systems), 3rd ed. San Mateo, CA, USA: Morgan Kaufmann, 2012.
[33] M. M. Deza and E. Deza, "Encyclopedia of distances," in *Encyclopedia of Distances*. Berlin, Germany: Springer, 2009, pp. 1–583.
[34] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *Amer. Statist.*, vol. 46, no. 3, pp. 175–185, 1992.
[35] S. B. Imandoust and M. Bolandraftar, "Application of K-nearest neighbor (KNN) approach for predicting economic events: Theoretical background," *Int. J. Eng. Res. Appl.*, vol. 3, no. 5, pp. 605–610, 2013.
[36] H. S. Khamis, K. W. Cheruiyot, and S. Kimani, "Application of K-nearest neighbour classification in medical data mining," *Int. J. Inf. Commun. Technol. Res.*, vol. 4, no. 4, pp. 121–128, 2014.
[37] J. Fuli and C. Chu, "Application of KNN improved algorithm in automatic classification of network public proposal cases," in *Proc. IEEE 2nd Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Chengdu, China, Apr. 2017, pp. 82–86.
[38] S. Madden. *Intel Berkeley Research Lab*. Accessed: Oct. 2, 2018. [Online]. Available: http://db.csail.mit.edu/labdata/labdata.html
[39] Advanticsys. *Advanticsys*. Accessed: Oct. 2, 2018. [Online]. Available: http://www.advanticsys.com/wiki/index.php?title=sg1000
[40] D. Gay, P. Levis, D. Culler, and E. Brewer. *nesC 1.1 Language Reference Manual*. Accessed: Oct. 2, 2018. [Online]. Available: https://github.com/tinyos/nesc/blob/master/doc/ref.pdf?raw=true
[41] P. A. Levis and D. Gay. *TinyOS Programming*. Accessed: Oct. 2, 2018. [Online]. Available: http://csl.stanford.edu/pal/pubs/tos-programming-web.pdf
[42] H. Harb, A. Makhoul, S. Tawbi, and R. Couturier, "Comparison of different data aggregation techniques in distributed sensor networks," *IEEE Access*, vol. 5, pp. 4250–4263, 2017.
[43] H. Harb, A. Makhoul, D. Laiymani, and A. Jaber, "A distance-based data aggregation technique for periodic sensor networks," *ACM Trans. Sensor Netw.*, vol. 13, no. 4, pp. 32:1–32:40, 2017.

**HASSAN HARB** received the master's degree in computer science and risks management from Lebanese University in 2013 and the the Ph.D. degree in computer science from Lebanese University and the University of Franche-Comté, France, in 2016. He is currently an Instructor with Antonine University and the American University of Culture and Education, Lebanon. His research interests are in wireless sensor networks emphasizing both practical and theoretical issues, and data mining and analyzing.

**ABDALLAH MAKHOUL** received the M.S. degree in computer science from INSA Lyon, Lyon, France, in 2005, and the Ph.D. degree in the problems of localization, coverage, and data fusion in wireless sensor networks from the University of Franche-Comté, Belfort, France, in 2008.

Since 2009, he has been an Associate Professor with the University of Franche-Comté. His research interests include internet of things, structural health monitoring, and real-time issues in wireless sensor networks.

Dr. Makhoul has been the TPC chair and a member of several networking conferences and workshops and a reviewer for several international journals.

**CHADY ABOU JAOUDE** received the M.E. degree (Hons.) in computer and telecommunications from Antonine University in 2002, and the Ph.D. degree (Hons.) in information and communication sciences from the University of Valenciennes and Hainaut-Cambresis, France, in 2013. He is currently an Associate Professor and the Dean of the Faculty of Engineering, Antonine University, Lebanon. His research interests include decision aiding processes, multimedia, wireless sensor networks, Internet of Things, and engineering education.