IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 18, 2019, accepted April 6, 2019, date of publication April 12, 2019, date of current version April 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2910886

A Spatial-Temporal Correlation Approach for Data Reduction in Cluster-Based Sensor Networks

GABY BOU TAYEH[1], ABDALLAH MAKHOUL[1], CHARITH PERERA[2], AND JACQUES DEMERJIAN[3]

[1]Femto-St Institute, UMR 6174 CNRS, Université de Bourgogne Franche-Comté, 25000 Besançon, France
[2]Cardiff University, Cardiff CF10 3AT, U.K.
[3]LARIFA-EDST, Faculty of Sciences, Lebanese University, Fanar 90656, Lebanon

Corresponding author: Charith Perera (pererac@cardiff.ac.uk)

This work was supported in part by the EIPHI Graduate School under contract ANR-17-EURE-0002, in part by the France-Suisse Interreg RESponSE Project, in part by the Lebanese University Research Program under Number: 4/6132, and in part by the EPSRC PETRAS 2 (EP/S035362/1).

ABSTRACT In a resource-constrained Wireless Sensor Networks (WSNs), the optimization of the sampling and the transmission rates of each individual node is a crucial issue. A high volume of redundant data transmitted through the network will result in collisions, data loss, and energy dissipation. This paper proposes a novel data reduction scheme that exploits the spatial-temporal correlation among sensor data in order to determine the optimal sampling strategy for the deployed sensor nodes. This strategy reduces the overall sampling/transmission rates while preserving the quality of the data. Moreover, a back-end reconstruction algorithm is deployed on the workstation (Sink). This algorithm can reproduce the data that have not been sampled by finding the spatial and temporal correlation among the reported data set, and filling the ''non-sampled'' parts with predictions. We have used real sensor data of a network that was deployed at the Grand-St-Bernard pass located between Switzerland and Italy. We tested our approach using the previously mentioned data-set and compared it to a recent adaptive sampling based data reduction approach. The obtained results show that our proposed method consumes up to 60% less energy and can handle non-stationary data more effectively.

INDEX TERMS Wireless sensor networks, data reconstruction, spatial-temporal correlation, data reduction.

I. INTRODUCTION

The momentum and growth of large-scale sensor networks have been increasing over the recent years. The rising popularity of such networks is due to the fact that they can be used in numerous and diverse event monitoring applications including traffic, air and water quality, e-health, environmental monitoring (wildlife, forest fires, storms, etc.), and many other applications. Such networks are expected to operate autonomously and for a long period of time. However, in a large scale sensor networks, the high volume of redundant data being communicated through the network increases collision, causes data loss, and most importantly it costs sensor nodes a large amount of scarce energy resources. Therefore, due to severe energy, computational and bandwidth constraints, a sound body of literature has centered on optimizing the efficiency of both the sensing and transmitting activities in order to maximize the lifetime of the network.

One of the most commonly used approaches to tackle this problem is the sampling rate adaptation [1]–[4]. A sampling rate is a rate at which a new sample is taken from a continuous signal provided by the sensor board. This rate can be adapted according to the input acquired from the monitoring area. If no significant change is noticed for a certain period of time, the sampling rate could be reduced for the upcoming period, and in contrast, if an event is detected, the sampling rate is increased. This sampling rate adaptation is based on event detection [1], [5]. Another sampling rate adaption technique takes into consideration the temporal and spatial correlation among the reported data [2], and limits the sampling rate of

The associate editor coordinating the review of this manuscript and approving it for publication was Mahmoud Barhamgi.

the sensors that show high correlation with other neighboring ones, and maximizes the sampling rate of those showing a little or no correlation at all. Both approaches aim to reduce the amount of redundant data being transferred through the network.

Other data reduction approaches focus solely on reducing the number of transmissions while maintaining a fixed sampling rate [6]–[10]. The most popular of them all is the dual prediction scheme. A prediction model capable of forecasting future values is trained and shared between the source and the destination, thus enabling the source sensor node to transmit only the samples that do not match the predicted value. Some approaches also combine both adaptive sampling and transmission reduction into a single mechanism [11] aiming to minimize further energy consumption.

In this paper, we propose a spatial-temporal Correlation based Approach for Sampling and Transmission rate Adaptation (STCSTA) in cluster-based sensor networks. The sensor nodes do not need to run any algorithm. The cluster head is responsible for collecting data from its member sensor nodes, computing a correlation function in order to measure the correlation degree among these nodes. Finally, the sensors that show high correlation will be asked to reduce their sampling rate and the ones showing low correlation will be asked to increase it. Moreover, in order to ensure the integrity of the data, a reconstruction algorithm deployed on the Sink station. The latter is used to reconstruct the ''non-sampled'' measurements by exploiting the temporal and spatial correlation among the reported data. We compare our approach to a Data Prediction with Cubic Adaptive Sampling (DPCAS) and to an exponential Double Smoothing-based Adaptive Sampling (EDSAS) using real sensor data. The latter and the former combines both adaptive sampling and transmission reduction into a single mechanism, allowing us to compare the efficiency of our proposal with two very effective approaches in terms of reducing radio communication.

The rest of the paper is organized as follows: In section II, the work related to energy efficient data reduction in a wireless sensor network is presented. In section III the system model is briefly explained and the energy model to calculate the energy consumption is illustrated. A detailed explanation of the proposed approach is provided in section IV, while experimental results are discussed in Section V. This paper ends with a conclusion section, in which the contribution is summarized and intended future work is outlined.

II. RELATED WORK

Resource management in sensor networks is a widely discussed topic among researchers. Subsequently, there have been numerous studies regarding this topic. In this section, we present and discuss the different approaches used to tackle this issue.

Compression [12]–[15] and aggregation [16]–[18] are two techniques aiming to reduce the amount of data routed through the network [19]. The former focus on compressing the data before transmission to the upper node in the network hierarchy and the latter filters and clean the data by removing redundant information before routing these data to the Sink station. Several data compression and aggregation techniques have been proposed in the literature. The authors in [12] proposed a compression technique for sensor networks organized in a cluster topology. The approach called Cluster-Based Compressive Sensing Data Collection (CCS) compresses data on the cluster head level by generating Compressive Sensing (CS) measurements based on block diagonal matrices created from the raw data received from neighboring sensors. Moreover, the compressed CS measurements are finally reconstructed at the base station (Sink). In [13] the authors proposed a compression scheme called Compressive Data Collection (CDC) for Wireless Sensor Networks, it exploits the spatial-temporal correlation among sensory data to perform compression. The scheme consists of two layers, the opportunistic routing with compression and the nonuniform random projection based estimation for reconstruction. The authors in [20] proposed a data aggregation technique called the Prefix-Frequency Filtering (PFF). This approach mainly consists of two aggregation layers, the first one is on the sensor level, and the second one is on the cluster head or the aggregator. On both layers, redundant measurements are filtered using the Jaccard similarity that measures the correlation among collected measurements. In [16] a Dynamical Message List Based Data Aggregation (DMLDA) technique is presented, it is based on a special data structure called dynamical list. The latter stores the history of received measurements, that are then used to filter any duplicates.

One of the most energy consuming activities in WSN beside transmission and processing is sampling, therefore several studies have been conducted on how to reduce the amount of sampled data through a technique known as ''adaptive sampling'', where a sensor can adapt its sampling rate according to the change in the input environment. The authors in [1] proposed and event-sensitive adaptive sampling and low-cost monitoring (e-Sampling) scheme, where each sensor has short and recurrent bursts of high sampling rate in addition to a low sampling rate. Depending on the analysis of the frequency content of the signal, each sensor can autonomously switch between the two sampling speed. The authors in [2] presents a decentralized temporal correlation based adaptive sampling approach, enabling each sensor to decide its own sampling rate while controlling the size of the sampling interval by limiting the interval size to an automatically calculated ''MaximumSkipSamplesLimit (MSSL)'' value.

The overwhelming majority of studies agree on the fact that radio transmission is the most consuming activity in WSN [21]–[23]. Accordingly, numerous studies focused on developing techniques to limit the number of radio transmissions. Most of these techniques are based on the concept of data prediction. The idea is to build on the Sink a prediction model using previously collected readings, that is capable of forecasting future measurements. Enabling the sensor node to transmit a reading only when the prediction does not

respect the error tolerance predefined the user. The authors in [6] proposed a Hierarchical Least Mean Squares (HLMS) adaptive filter as a prediction model, which is one of the many adaptive filter based approaches [7], [8], [24]. This filter consists of multiple layers of regular Least Mean Square (LMS) filters, each layer takes feedback from the previous layer in the hierarchy aiming to minimize the prediction error of the model. Another technique called Derivative Based Prediction (DBP) was introduced in [25], it is less complex than the adaptive-filter based methods. The prediction model is simply a straight line that interpolates a fixed window of data of size m using the first and last l values in the window. In [11] the authors proposed an approach that combines an adaptive sampling method that is based on the TCP CUBIC congestion protocol, with a transmission reduction method that is based on an exponential predictive mode. The complete data set including the ''non-sampled'' and ''non transmitted'' measurements are then reproduced on the sink by interpolating the received measurements. This approach was inspired by both [26], and [27]. The latter uses an exponential Double Smoothing-based Adaptive Sampling (EDSAS) technique, that adapts the sampling rate of a sensor based on the accuracy of a prediction model. As long as this model is producing good predictions the sampling rate is kept low. It is increased, however, when the predictions surpass a predefined error threshold. The former operates in similar fashion, more specifically it adopted the TCP congestion control to adapt the sampling rate of the sensor node. Thus the approach is called Adaptive sampling TCP (ASTCP).

Both compression and aggregation are effective in term of reducing the data load in the network, however, their performance is limited and cannot reach the efficiency of techniques such as adaptive sampling and transmission reduction. Therefore, compression and aggregation are considered to be as a complementary layer that can be added to adaptive sampling and transmission reduction to further increase their efficiency. Despite being very effective in reducing the amount of sampled and transmitted data, adaptive sampling and transmission reduction techniques can still consume a substantial amount of energy. This is proportionally related to the complexity of the algorithms that are required to be implemented on the sensor level. The CPU running complex algorithms can consume more energy than the sampling activity [21], which renders the adaptive sampling technique obsolete in case the implemented algorithm requires a large number of CPU cycles.

In order to schedule the sampling intervals of sensor nodes and reduce energy transmission, some approaches rely on the spatial-temporal correlation between sensor nodes deployed in the monitoring area [28]–[32]. The Authors in [28] proposed an Efficient Data Collection Aware of spatial-temporal Correlation (EAST). In the latter, the sink subdivides the event area into spatially correlated cells of the same size, then, in each cell, the node having the highest residual energy is elected as a representative node. Only the latter transmits data to the sink while also applying a temporal correlation suppression method on its collected data. Finally, at each time instance, the representative node is re-elected according to the same previous rule. The main drawback of this approach is the size of the cell representing an area of spatially correlated nodes is static, and it is not calculated according to the real level of correlation. Moreover, the representative node is chosen according to residual energy rather than its correlation with other nodes in the cell. Therefore, the term ''representative'' is not necessarily true.

In [29] the authors proposed a sleeping schedule algorithm that aims to minimize the total spatial-temporal coverage redundancy among neighboring nodes while maximizing coverage. Each sensor node compares itself with neighboring ones using a weight criteria and it locally optimizes its scheduling according to its coverage redundancy. This method requires constant message exchange between sensor nodes in order to keep track of the changing weight of each one of them, which can produce overhead.

The authors in [30] proposed a spatial-temporal correlation model that aims to extend the network lifetime by scheduling a sleeping period for sensors showing high similarities with other ones belonging to the same cluster. The similarity is measured by computing the Euclidean Distance, Cosine Similarity and Pearson Product-Moment Coefficient (PPMC). If the result of one of the three indicates a high similarity, the sensor node is set to sleep for half of the period time (1 period = N samples). The first problem with such an approach is if a sensor X shows a similarity with a sensor Y, the opposite is also true (sensor Y will show similarity with sensor X), therefore, according to this approach, both sensors will be set to sleep. By doing so correlated sensors will miss simultaneously instead of compensating for one another by keeping one of them awake. The second problem is that the sleeping duration is static instead of being computed in a dynamic way according to the level of correlation.

Motivate by the problems related to the aforementioned approaches, we present in this paper a spatial-temporal Correlation approach for Sampling and Transmission rate Adaptation (STCSTA) in cluster-based sensor networks. Our approach does not require any algorithm to be implemented on the sensor level, the only task performed by sensors are uniquely sampling and transmission. All the work is done on the Cluster-Head (CH) level, where at the end of each round (duration predefined by the user), the CH runs an algorithm that finds the spatial correlation among the data reported by the sensors belonging to the same cluster. Then, it transmits to each one of them its new sampling rate for the next round according to its level of correlation with other neighboring sensors in the cluster. The sampling rate scheduling respects a strict protocol that keeps the sampling rate of the sensors showing high correlation with a large number of nodes at an optimal maximum level. Moreover, the protocol prevents highly correlated sensors from missing simultaneously, allowing one to compensate for another. in addition to sampling rate scheduling, and in order to ensure the integrity of the data, a reconstruction algorithm is deployed on the Sink.

This algorithm can identify the time stamps where data has not been received due to a reduction in the sampling rate of a specific sensor, and reconstruct them using the spatial-temporal relations among the collection of data reported by the sensors.

III. SYSTEM AND ENERGY MODEL

A. SYSTEM MODEL

We consider a set S of N sensor nodes and C cluster heads deployed over a specific monitoring area at locations LS = $\{ls_1, ls_2, \ldots, ls_N\}$ and LC = $\{lc_1, lc_2, \ldots, lc_C\}$ respectively, where a sensor S_i is located at the location ls_i and a cluster-head C_j is located at the location lc_j, and the Sink S is placed in a distant location at a position l_0. Sensor nodes are grouped into clusters, where each one of them belongs to one cluster only. The cluster heads are considered to be more powerful than sensor nodes in term of processing capabilities and they have been allocated larger energy resources. Figure 1 illustrates an example of the described network architecture for one cluster.

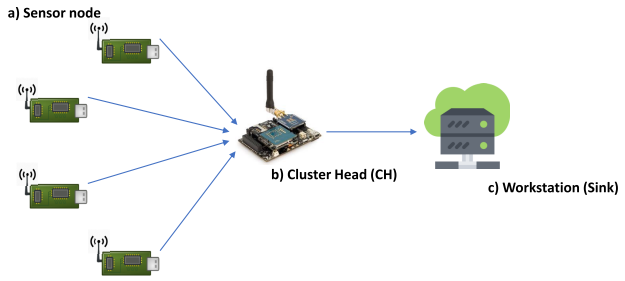

FIGURE 1. Illustrative example of the network architecture.

The network is periodic and operates in rounds, where each round R is exactly P seconds, and it is subdivided into m time slots, where at each time slot a sensor samples one measurement. Therefore, the maximum sampling rate (SR_{max}) is considered to be P/m samples per round. During the very first round, each sensor node collects data using the maximum sampling rate SR_{max} and transmits the readings to the CH after each acquisition. On the CH level, when the latter receives a measurement from any sensor S_i it stores the values in its memory and routes it directly to the Sink. At the end of the first round, the CH would have stored in his memory the following matrix M. where n is equal to the current sampling rate (SR_{max}) in this case, and N is the number of sensors in the cluster.

$$M = \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \ldots & x_1^n \\ x_2^1 & x_2^2 & x_2^3 & \ldots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_N^1 & x_N^2 & x_N^3 & \ldots & x_N^n \end{bmatrix}$$

The CH than proceeds to computing the correlation between each pair of sensors (The number of possible pairs is $\frac{N(N-1)}{2}$). Using the obtained correlation results the CH calculates than transmit to each sensor node its new SR. A detailed explanation of how the correlation is calculated and how the new SR is determined is provided in section IV. For the next round, each sensor samples data according to its new sampling rate provided by the CH. For Instance, if the latter demands a specific sensor to reduce its sampling rate by 40%, and supposing that SR_{max} is equal to 50 measures/round, the sensor is supposed to sample 30 measurements instead. If each period is 10 minutes long (600s), instead of sampling a measurement every 12 seconds (600/50), the sensor would sample a measurement every 20 sec (600/30). Moreover, Knowing the duration of each period, the maximum sampling rate and the time stamp when each measurement was received, both the Sink and the CH are capable of identifying the non-sampled data, which will be replaced by "Nan" (see matrix M′) in order to reconstruct them later at the Sink station and in order to make the computation of the correlation among sensor nodes easier for the CH as explained in section IV-A. Therefore, the stored matrix that is used to compute the correlation will actually be as shown below, where n is equal to the maximum number of samples per round (SR_{max}):

$$M' = \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \ldots & x_1^{50} & Nan & x_1^n \\ x_2^1 & x_2^2 & x_2^3 & \ldots & Nan & Nan & x_2^n \\ \vdots & \vdots & \vdots & Nan & \vdots & \vdots & \vdots \\ x_N^1 & x_N^2 & x_N^3 & \ldots & x_1^{50} & Nan & x_N^n \end{bmatrix}$$

B. ENERGY MODEL

In order to compute the energy consumption of a sensor node [33], [34], it is necessary to take into consideration the energy consumed by every single operation performed by the node. Generally, the consumed energy relates to four main tasks, namely, sampling, logging, processing, and radio transmission. Therefore, the energy consumption model can be defined as:

$$E_{node} = E_{sampling} + E_{logging} + E_{processing} + E_{radio} \quad (1)$$

where $E_{sampling}$ is the energy required for sampling one value, $E_{logging}$ is the required energy to log data in the memory, $E_{proccessing}$ is the required energy to run and algorithm consenting of N CPU cycles, and E_{radio} is the energy required to transmit a b bits packet for a distance d. In this article we use the energy model discussed in [21] to calculate the overall energy consumption of each sensor node.

IV. THE PROPOSED APPROACH (STCSTA)

In this section, we will explain in detail, how the correlation between sensor nodes and the new sampling rates of each sensor are calculated.

A. COMPUTING CORRELATION AND SAMPLING RATE ALLOCATION

1) ALGORITHM 1 - LINE(2–14)

After a round is completed, each sensor node would have transmitted to the cluster head a different number of measurements since the sampling rate of each one of them can be

different. Nevertheless, as mentioned earlier the CH identifies the non sampled data and fill their corresponding place in the vector by a Nan value, therefore all the vectors will have the same size n. However, the correlation between two vectors containing Nan values cannot be computed. Therefore, each and every Nan value is replaced by the value of the first ''non-Nan'' value that comes before it in the same vector. For instance, in the ''M′'' matrix, x_1^{51} is Nan it will be set equal to the same value as x_1^{50}, and x_2^{50} and x_2^{51} are set equal to the same value as x_2^{49}, and so on.

2) ALGORITHM 1 - LINE(17–22)

Afterward, the linear dependency of each pair of vectors $(v_i, v_j) \in \mathrm{M}'$ is calculated using the Pearson correlation coefficient. The latter is known as the best method of measuring the association between variables of interest because it is based on the method of covariance. It gives information about the magnitude of the association, or correlation, as well as the direction of the relationship. The Pearson correlation coefficient is described in the equation 2 below, where μ and σ are the mean and standard deviations.

$$\rho(v_i, v_j) = \frac{1}{n-1} \times \sum_{k=1}^{n} \left(\frac{\overline{v_{ik} - \mu_{v_i}}}{\sigma_{v_i}}\right)\left(\frac{\overline{v_{ik} - \mu_{v_j}}}{\sigma_{v_j}}\right) \qquad (2)$$

The justification behind using the Pearson correlation can is illustrated in Figure 2. We have used a data set of 92 sensors to generate 4 graphs that show the number of sensors that are moderately & highly correlated with 4 randomly chosen sensors during each period and for the first 100 periods. For instance, in Figure 2(a) we notice that this randomly chosen ambient temperature sensor correlates with a large number of sensors during each period. On average it correlates with 27 sensors as the mean values shows. Same for Figure 2(b) and (c) on average these sensors correlate with approximately 30 other sensors that are in the same cluster. However, The mean value in Figure 2(d) is significantly lower (mean=19), in section V-D we will see how this will reflect on the results.

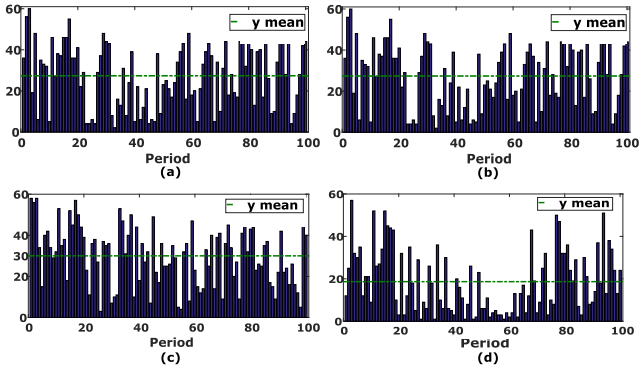

FIGURE 2. Figure showing the number of moderately & highly correlated sensors (Pearson correlation coefficient ≥ 0.5) during each one of the first 100 Periods. (a) Ambient temperature sensor. (b) Surface temperature sensor. (c) Relative humidity sensor. (d) Wind speed sensor.

Heterogeneous environmental data beside other types of data such as medical data (vital signs), movement tracking data (speed, acceleration, location) and etc, are usually highly and/or moderately correlated. This correlation thus can be used in order to reduce the number of transmitted measurements by deriving values from other observed ones. This is indeed the motivation behind using correlation to adapt the sampling rate of the sensors.

3) ALGORITHM 1 - LINE(23–28)

After computing the correlation value of each sensor i with all the other sensors belonging to the same cluster, the CH looks for the sensor j that it correlates the most with as shown in table 1.

TABLE 1. The correlation table.

Sensor i	Sensor j (has the max correlation with Sensor i)	Correlation degree
1	54	0.91
2	7	0.87
3	5	0.70
4	2	0.96
5	6	0.75
...	...	..
n	32	0.88

4) ALGORITHM 1 - LINE(29–38)

Afterward, the CH counts the number of occurrences of each sensor j in the second column of the table and stores them in a list according to their ascending order.

5) ALGORITHM 1 - LINE(39–49)

Starting from the first sensor j in the ordered list, the CH looks in table 1 for the sensor j in the first column and extract the value of its max correlation from the third column. Then the CH notifies j that its sampling rate must be reduced proportionally to the correlation value. For instance, if sensor 5 was first in the ordered list, the CH would notify it that its sampling rate for the next round must be reduced by 75%, since its level of correlation with sensor 6 is 0.75. Then the sensor j (in this case 5) is flagged as already notified. Thus, for the next sensor j in the ordered list, if its matching sensor i is already flagged. Instead of reducing its sampling rate proportionally to the level of correlation, it is reduced by (100 - i's reduction %). For instance, if the next sensor j in the list is 3, it matches with sensor 5 in table 1, therefore, it's sampling rate will be reduced by $100 - 75 = 25\%$. And so on, until the last element in the ordered array.

6) ALGORITHM 1 - LINE(50–56)

However, some sensors may not appear in the second column of the table 1, since they have not been matched with other sensors. Therefore, the CH looks for these sensor in the 1'st column of table 1, and for each sensor i, it find their matching sensor j in the second column, looks at how much the sampling rate was reduced for sensor j and notifies sensor i that its sampling rate must be reduced by (100 - sensor j's reduction %).

The same explained operation is repeated at the end of each round. Therefore, enabling each sensor node to adjust its sampling rate according to its level of correlation with other sensors in the network. The algorithm 1 illustrates the proposed method that is implemented on the CH.

Algorithm 1 STCSTA.

Input: *SRmax* (1 sample/ X seconds)
while *Energy* $\neq 0$ **do**
  $k \leftarrow 1$
  for each sensor j in the cluster **do**
    receive the first value v_j^0 at the beginning of the round
    $data[j][0] \leftarrow v_j^0$
    $lastReceived[j] \leftarrow v_j^0$
  end for
  while ! end of round **do**
    if nothing is received from sensor j after X seconds **then**
      $data[j][k] \leftarrow lastReceived[j]$
    else if v_j^n is received during the X seconds count **then**
      $data[j][k] \leftarrow v_j^n$
      $lastReceived[j] \leftarrow v_j^n$
    end if
    $k \leftarrow k + 1$
  end while
  if end of round **then**
    for i=1 to N **do**
      for j=i+1 to N **do**
        $corrArray[i][j] \leftarrow PearsonCorr(data[i][:], data[j][:])$
      end for
    end for
    for i=1 to N **do**
      $maxCorr[i][0] \leftarrow i$
      $[index, value] \leftarrow max(corrArray[i][:])$
      $maxCorr[i][1] \leftarrow index$
      $maxCorr[i][2] \leftarrow value$;
    end for
    $k \leftarrow 1$
    for each element i $\in$ the second column of maxCorr **do**
      if i $\notin$ first column of countOcc **then**
        $count \leftarrow$ count how many times i occures in the second column of maxCorr
        $countOcc[k][0] \leftarrow i$
        $countOcc[k][1] \leftarrow count$
        $k \leftarrow k + 1$
      end if
    end for
    order countOcc in ascending order according to the second column
    $k \leftarrow 1$

Algorithm 1 *(Continued.)* STCSTA.

    for each element j $\in$ the first column of countOcc **do**
      $match \leftarrow maxCorr[j][1]$
      if reduce[match-1] is empty **then**
        Notify sensor j that its sampling rate must be reduced by (maxcorr[j][2]*100)%
        $reduce[j-1] \leftarrow (maxcorr[j][2] * 100)$
      else
        Notify sensor j that its sampling rate must be reduced by (100 - reduce[match-1])%
        reduce[j-1]=100 - reduce[match-1]%
      end if
    end for
    for j=1 to N **do**
      if reduce[j-1] is empty **then**
        $match \leftarrow maxCorr[j][1]$
        Notify sensor j that its sampling rate must be reduced by (100 - *reduce*[match-1])%
      end if
    end for
  end if
end while

B. ANALYSIS STUDY

The objective of this algorithm is to create and manage a sampling rate balancing system based on the correlation degree between the nodes belonging to the same cluster. The idea is to match each sensor node with the one that correlates the most with, in such a way that, if one node of the paired couple reduces heavily its sampling rate, the other one keeps it high and vice versa, allowing them to compensate one another. This compensation mechanism is crucial for the success of the reconstruction algorithm in term of minimizing the estimation error and increasing the quality of the replicated data. The latter relies on the correlation among sensor nodes in order to reconstruct the non-sampled measurements. Therefore, if highly correlated sensors are missing data simultaneously this would negatively affect the accuracy of the reconstructed measurement. When the balancing of non-sampled data is kept in check on the CH level, The reconstruction algorithm on the Sink will theoretically produce better estimations.

In this section, we will illustrate an example that explains our algorithm step by step. The latter provides a better analysis of what happens at the end of each round on the cluster head to better understand why and how this compensation system works. Let us start by assuming that at the end of a given period, the CH has already computed the correlation between each pair of sensors belonging to the same cluster. In addition, we assume that the CH already matched each sensor with the one that correlates the most with and stored the results in a table similar to Table 2. The next step is to count for the sensors appearing in the second row of the

TABLE 2. Table showing for each sensor its best match (maximum correlation) and the degree of correlation with this match.

Sensor i	1	2	3	4	5	6	7	8	9	10
Sensor j (has max correlation with i)	8	1	7	3	9	1	10	7	7	7
Correlation degree $*10^{-2}$	78	69	54	92	85	72	79	83	89	90

table how many times it has been matched. For instance, sensor 7 has been matched 4 times, sensor 1 has been matched 2 times, and sensor 10, 9, 3, and 8 have been matched only one time. The matched sensors are then ordered in ascending order according to how many times they have been matched. the order will then be: {sensor 8, sensor 3, sensor 9, sensor 10, sensor 1, sensor 7}.

Starting from the first sensor in the list (sensor 8) the CH looks for the sensor that it matches with. Looking at table 2 we see that sensor 8 matches with sensor 7. The CH then checks whether the sampling rate of sensor 7 for the next round has been decided yet. If it is not the case the CH notes that the sensor 8 must reduce its sampling rate for the next round by 83%, since the correlation degree for sensor 8 with its match is 0.83. The CH then follows the same procedure for the next sensor in the ordered list. sensor 3, 9, and 10 they all match with sensor 7 too, and since the sampling rate of sensor 7 has not been decided yet, their sampling rate will be reduced by 54%, 89%, and 90% respectively for the next round. Now the CH searches for the sensor that matches with the next sensor in the ordered list (sensor 1). Looking at table 2 we see that it is sensor 8. However, the sampling rate of sensor 8 has been already decided to be reduced by 83%, therefore instead of reducing the sampling rate of sensor 1 by 78% it will be reduced by 100-83%, therefore 17% only. Same for sensor sensors 7, it matches with sensor 10, therefore its sampling rate must be reduced by 100-90% (10% only).

The next step is to adapt the sampling rate of the sensors that do not appear in the second row of the table, or in other words they have not been matched with other sensors in the cluster. In this example, the non-matched sensors are sensor 2, 4, 5 and 6. Starting by sensor 2, its match is 1, therefore the sampling rate of sensor 2 for the next round must be reduced by 100-17% (83%), same for sensor 4,5, and 6 their sampling rate will be reduced respectively by 46%, 11%, and 83%.

Before computing the percentage of the reduction in sampling rate, the matched sensors are first ordered in ascending order according to how many times they have been matched. The reason behind this crucial step can be explained as follows: Let us suppose the list has not been ordered, and the CH started by sensor 7, which has been matched 4 times with 4 different sensors. The sampling rate of sensor 7 will be reduced by 79%. Therefore, eventually, the sampling rates of sensors 3, 8, 9, and 10 will be reduced by 21% only compared to 54%, 83%, 89%, and 90% respectively if the list was ordered. In consequence of not ordering the list first, the overall reduction in the sampling rate of the sensors would be reduced, which would lead to an increase in data transmission and energy consumption. Since sensor 7 can compensate for 4 other sensors, it is wise to leave it until the end, allowing the sensors that it matches with to reduce more their sampling rate.

TABLE 3. Table showing the % of SR reduction for each sensor compared with its match.

Sensor i	1	2	3	4	5	6	7	8	9	10
SR reduction (%)	17	83	54	46	11	83	10	83	89	90
Sensor j (has max correlation with i)	8	1	7	3	9	1	10	7	7	7
SR reduction (%)	83	17	10	54	89	17	90	10	10	10

A summary of the results is illustrated in table 3. We notice that if a sampling rate of a particular sensor is highly reduced, the one of the sensor that it correlates the most with will be proportionally and slightly reduced (e.g. sensors 2 and 1). This balanced reduction is meant to compensate for the matched sensor since the non-sampled values will eventually be derived mostly from its best match. Similarly, if the sampling rate of a sensor is slightly reduced, this will give more freedom to its match thus allowing it to highly reduce its sampling rate (e.g. sensors 5 and 9).

C. RECONSTRUCTION OF THE NON SAMPLED DATA

In this section, the algorithm used to reconstruct non-sampled data is explained. As mentioned earlier, the Sink detects and replaces non sampled data with a "Nan" value. After a certain period of time, let's say M rounds, defined by the user, the sink runs a reconstruction algorithm that can replace all the "Nan" values with estimations calculated using the spatial and temporal correlation among the data reported by the sensor nodes in the network. This algorithm it is deployed on the Sink instead of the CH due to its complexity. If deployed on CH it will consume a great amount of energy. The reconstruction algorithm proposed in [35] essentially used to estimate missing data in co-evolving time series was adopted and adapted to suit our case. Assuming after M rounds, the Sink would have stored in his Sink the following data-set:

$$SinkDataSet = \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \dots & x_1^{50} & Nan & x_1^{n*M} \\ x_2^1 & x_2^2 & x_2^3 & \dots & Nan & Nan & x_2^{n*M} \\ \vdots & \vdots & \vdots & Nan & \vdots & \vdots & \vdots \\ x_N^1 & x_N^2 & x_N^3 & \dots & x_1^{50} & Nan & x_N^{n*M} \end{bmatrix}$$

A probabilistic model (Figure 3) is built to estimate the expectation of missing values conditioned by the observed part. The model is built by initializing a latent variable Z_1, a linear mapping matrix F and a projection matrix G, for the readers interested in how these values are initialized please refer to [35]. Afterward, using the linear mapping F the algorithm can proceed to calculate the other Z_n (n $\in$ [1,n*M]) by simply multiplying $Z_{n-1} * F$. Once all the values of Z_n are calculated, the algorithm then estimates the observed and

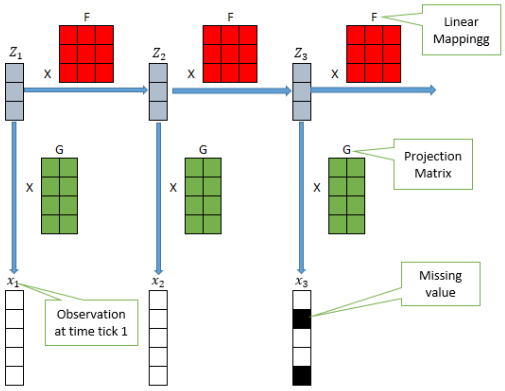

FIGURE 3. The probabilistic model.

non-observed (Nan) values. This is achieved by multiplying each Z_n by the projection Matrix G, which gives the predictions ($[x_1^n, \ldots, x_N^n]$) of the values at the sampling time n. Using the estimations, and the observed part, the algorithm then tries to maximize the log-likelihood of the observed sequences using an EM iterative algorithm [36] in order to update F and G and produce more accurate predictions. The same operation is repeated with the newly computed F and G until the number of iteration reaches a maximum value predefined by the user, or until the log-likelihood is no longer increasing.

V. EXPERIMENTAL RESULTS

We implemented our algorithm in addition to DPCAS [11] in a custom WSN simulator built in Matlab, and we conducted multiple experiments in order to evaluate and compare their performances. In the simulation, we used real sensor readings collected from a sensor network that was deployed at the Grand-St-Bernard pass between Switzerland and Italy [37]. The Network consisted of 23 sensors, each one of them collects 9 different environmental features with a fixed sampling rate of 1 sample every 2 minutes. We have chosen 4 out of these 9 features (ambient temperature [$C°$], Surface temperature [$C°$], relative humidity [%], and wind speed [m/s]), since the others are not complete. Environmental features are usually stationary, therefore, in addition to taking a sample every 2 minutes, and for a rigorous comparison, we set up two other scenarios, the first one, a sample is taken every 10 minutes instead, and the second one, a sample is taken every 20 minutes. In this way, the data will become ''non-stationary'' which makes it more realistic and harder for both algorithms to adapt to high variation in collected measurements. The raw data set (sample every 2mins) consists of 10000 readings for each sensor, for the 1st scenario we will end up with 2000 readings instead, and 1000 readings for the second one.

In DPCAS the parameter ϵ defines the error tolerance of the application, the greater is ϵ, the less is the amount of data that will be sampled and transmitted. However, the error of the estimated data will increase. Therefore, the value of ϵ is the level of trade-off between the quality of the replicated data and the amount of sampled and transmitted measurements. In our experimentation, we set up five different values for ϵ ranging between 0.1 and 0.5 and we compare our approach to DPCAS for each value of ϵ.

A. SAMPLING AND TRANSMISSION REDUCTION

In this section, we will explore and compare the effectiveness of each algorithm in reducing the number of both sampled and transmitted data in three different scenarios. As mentioned earlier, each sensor node collects 4 different environmental features (ambient temperature, surface temperature, relative humidity, and wind speed). For simplicity and better visualization of the results, all the figures will be illustrating the percentage of the aggregated sum of the data sampled and transmitted by the 23 nodes combined and for all features.

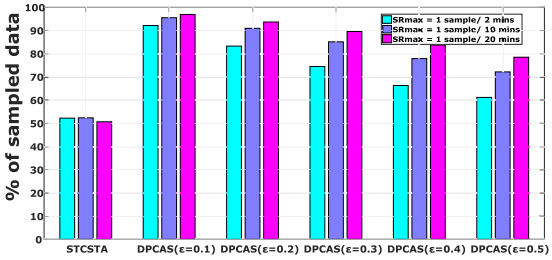

FIGURE 4. Average percentage of data sampled by each sensor node.

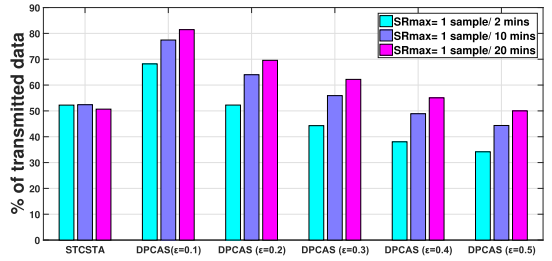

FIGURE 5. Average percentage of data transmitted by each sensor node.

Figure 4 and 5 shows that on one hand, the bigger is the sampling interval between two consecutive measurements (higher variations in data), the greater is the average percentage of both sampled and transmitted data will be when DPCAS is deployed. On the other hand, when our approach (STCSTA) is deployed, the average percentage remains stable despite the level of variations in collected measurements, which makes it more robust, dynamic and tolerable to high variations. This is not the case for DPCAS however, its effectiveness can be significantly affected (a double-digit increase in sampled and transmitted data) depending on the type of data being collected. Moreover, Figure 4 and 5 shows that STCSTA has the upper hand when it comes to reducing the number of both sampled and transmitted data. For sampled data, Figure 4 shows that STCSTA outperforms DPCAS in all scenarios and for all the values of ϵ. Figure 5 shows the average percentage of data transmitted by each one of the

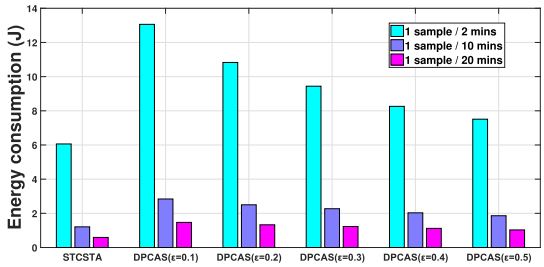

FIGURE 6. Average energy consumption of each sensor node.

23 nodes for both algorithms in 3 different scenarios and using different ϵ for DPCAS. The obtained results show the following: STCSTA outperforms DPCAS when $\epsilon \leq 0.2$ in all scenarios. However, for $\epsilon = 0.3$ DPCAS transmits less data in the first scenario ($SR_{max} = 1$ sample/ 2mins), but more data in the other two scenarios ($SR_{max} = 1$ sample/ 10 mins and 1 sample/ 20 mins). Finally, For $\epsilon = 0.4$ and 0.5, DPCAS is slightly better in the first two scenarios. To sum it all up, the results in Figure 5 show that STCSTA outperformed DPCAS 9 times, the latter outperformed STCSTA 5 times, and finally, we have 1 tie.

To conclude on this, when it comes to reducing the sampling and transmission rate, thus the energy consumed by the sampling activity $E_{sampling}$ and the transmission activity E_{radio} STCSTA is more effective than DPCAS.

B. ENERGY CONSUMPTION

In this section, we present a comparison between the average energy consumed by the 23 sensor nodes when DPCAS and STCSTA are deployed.

Previously, in section V-A, the obtained results clearly show that the E_{radio} and $E_{sampling}$ are less when STCSTA is deployed since the amount of sampled and transmitted data is directly related to the energy consumed by the sampling and transmitting activities. However, according to Equation 1, we still need to calculate $E_{logging}$ and $E_{Processing}$. This is where our approach shows a clear advantage. Knowing that in DPCAS an algorithm must be deployed on the node that handles 4 different sensors at a time. The node needs to perform reading and writing in the memory, and it needs to compute mathematical operation using the CPU. Therefore, the node will be consuming additional energy ($E_{logging}$ and $E_{Processing}$). However, for STCSTA, the node does not have to run an algorithm, nor to perform read and write in the memory, it simply collects a measurement using the integrated sensors, and directly transmits it to the CH. Therefore, no additional energy consumption is required. Figure 6 shows the average energy in Joule consumed by each one of the 23 deployed nodes. It is clear that our approach consumes approximately from 20% up to 60% less energy than DPCAS depending on the scenario and the value of ϵ.

C. COMPARISON WITH A BASELINE METHOD

The previously described results demonstrated that our approach STCSTA outperforms DPCAS in terms of energy preservation. The DPCAS algorithm in [11] was compared to two other approaches that use a similar technique, namely EDSAS [27] and ASTCP [26]. As mentioned in section II, the ASTCP algorithm was inspired by the EDSAS. Moreover, the DPCAS algorithm was inspired by both ASTCP and EDSAS. In this section, we will use the EDSAS as a baseline for comparison since it was the root algorithm that inspired both ASTCP and DPCAS. Table 4 below shows the average energy consumed by each node in all scenarios and for the same value of $\epsilon = 0.1$ used in [11]. The obtained results are fairly similar to the ones obtained in [11] and our approach remains better.

TABLE 4. Table comparing STCSTA and DPCAS to the baseline EDSAS.

Algorithm	STCSTA			DPCAS			EDSAS		
Sampling Rate 1 sample/ x min	x=2	x=10	x=20	x=2	x=10	x=20	x=2	x=10	x=20
Energy (J)	6.06	1.21	0.59	13.06	2.84	1.47	13.38	2.92	1.52

TABLE 5. Quality of the reconstructed data.

		Ambient Temp	Surface Temp	Relative Humidity	Wind Direction
1 sample/2 mins	RMSE	1.12	1.33	2.7	16.5
	MAE	0.71	0.91	1.89	8.78
1 sample/10 mins	RMSE	1.26	1.56	3.68	18.26
	MAE	0.74	1.09	2.55	9.13
1 sample/20 mins	RMSE	1.43	1.95	4.78	23.53
	MAE	0.87	1.43	3.21	11.93

D. THE QUALITY OF THE REPLICATED DATA

In order to measure the quality of the final set of data, we use the accuracy of the estimations as the validation criteria. Specifically, we use the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) as an accuracy metric. Table 5 shows the RMSE and MAE of the estimated data for the three scenarios. For ambient temperature, surface temperature and relative humidity the errors are low. This is due to the fact that the spatial-temporal correlation of these features is strong, so the estimation algorithm can obtain an accurate and solid relationship based on mining correlation rules. Table 5 also shows that the error increases when the sampling interval widens. The bigger is the sampling interval, the weaker is the temporal correlation, therefore the harder is for the estimation algorithm to accurately estimate values. For Wind direction, the errors increase significantly but they are still proportionally low compared with the range of value for the wind speed (between 0 and 350 m/s). Wind speed has no spatial correlated with any other feature. Moreover, the wind speed value varies significantly between one sample and the other as shown in Figure 10, therefore the temporal correlation is weak as well, that is why it has the highest error among other features.

Figures figs. 7 to 10 shows a reconstructed signals for ambient temperature, surface temperature, relative humidity, and wind speed respectively. As shown in the figures, the data estimation (reconstruction) algorithm has been able to capture both the dynamics of the signal as well as the correlation across given inputs, therefore achieving a very satisfying

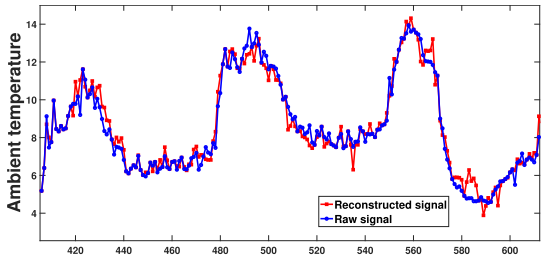

FIGURE 7. Reconstructed ambient temperature signal.

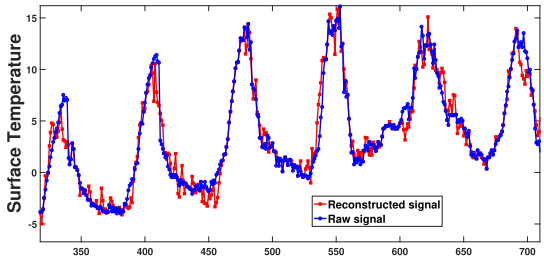

FIGURE 8. Reconstructed surface temperature signal.

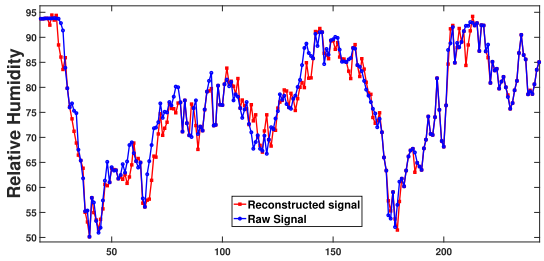

FIGURE 9. Reconstructed relative humidity signal.

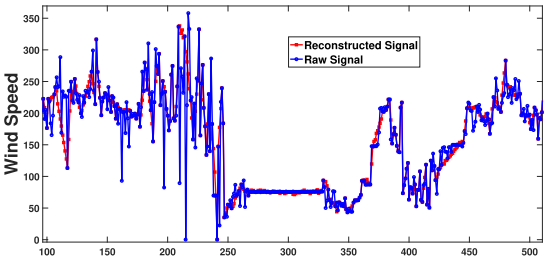

FIGURE 10. Reconstructed wind speed signal.

reconstruction of the signals. To conclude on the quality of the replicated data, simulation results presented in this section, demonstrated that the Sink is capable of reproducing the ''non-sampled'' data with a tolerable error margin. Thus, using our approach a sensor node can significantly reduce its sampling rate without affecting the integrity of the data.

E. THE EFFECT OF THE SAMPLING STRATEGY ON ERROR MINIMIZATION

The previous results have evaluated the efficiency of our proposed approach (STCSTA) in terms of reducing data transmission and energy consumption as well as the quality of the data replicated on the Sink. However, as previously explained in section IV-B, the objective of our algorithm is to guarantee that the highly correlated sensors are not skipping data sampling simultaneously in order to reduce the reconstruction error. That was in theory, Therefore, in this section, we put the theory into practice in order to justify this claim.

Instead of building a list of matching sensors, ordering the list, and reducing the sampling rate of each sensor proportionally to its match. We eliminated the steps from line 30 and upward in Algorithm 1, only to allow a sensor to reduce its sampling rate according to its highest degree of correlation. For instance, let's assume that the sensor 1 has the highest correlation degree with sensor 5 (0.8). Without checking whether sensor 5 has already reduced its sampling rate or not, it will automatically reduce it by 80%. There is a chance that sensor 5 has already reduced its sampling rate lets say by 70%. Thus, both sensor 5 and 1 will skip sampling simultaneously which would, in theory, affect negatively the reconstruction algorithm, which will lead to an increase in the reconstruction error. We will be calling this method ''The exaggerated sampling reduction'' method. Table 6 shows the % of increase in the reconstruction error when this method is applied. We notice that the Reconstruction error increases significantly in all scenarios and for all environmental features, which justifies our controlled sampling strategy.

TABLE 6. Percentage of increase in reconstruction error (the exaggerated sampling reduction method).

		Ambient Temp	Surface Temp	Relative Humidity	Wind Direction
1 sample/2 mins	RMSE	16.9 %	34.5 %	20 %	45.3 %
	MAE	12.6 %	41.7 %	16.93 %	23.4 %
1 sample/10 mins	RMSE	26.9 %	44.8 %	35.8 %	52.7 %
	MAE	39.1 %	52.3 %	29.4 %	75.2 %
1 sample/20 mins	RMSE	25.8 %	48.2 %	50.2 %	36.0 %
	MAE	25.2 %	44.0 %	45.7 %	59.2 %

F. SCALABILITY AND LIMITATIONS

Obviously, the scalability of such a network depends on the computational power of the CH and its memory capacity. A more powerful CPU and big memory size mean that the CH could handle a large number of sensors simultaneously. The weaker is the CPU and the smaller is the memory size, the fewer nodes a CH can handle. A great number of devices that can be used as a CH are currently available in the market, they all have different features and characteristics. One can find cheap less powerful CH device for personal use or an expensive and powerful device for commercial use. Therefore, the choice of the CH depends on the size of the network a user wants to deploy. A network consisting of thousands of nodes will certainly need a powerful CH. However, a network consisting of a few hundred or tens of nodes could work just fine with a less powerful CH.

Our proposed algorithm is not very complex though, it has a complexity that is linear in time (O(n)). This linear complexity allows the CH to handle a large number of nodes with minimal computational power. Regarding the memory size required by the STCSTA, assuming that the number of nodes in the cluster is N, and each value is encoded into 8bytes.

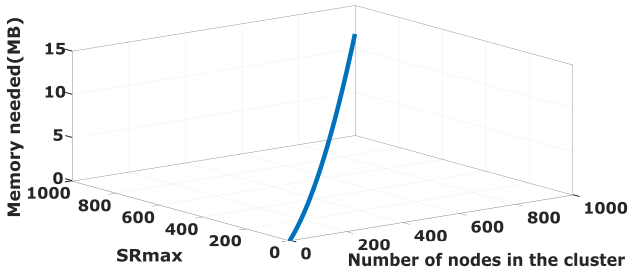

FIGURE 11. Memory size needed for the first part of the Algorithm (line 1-28).

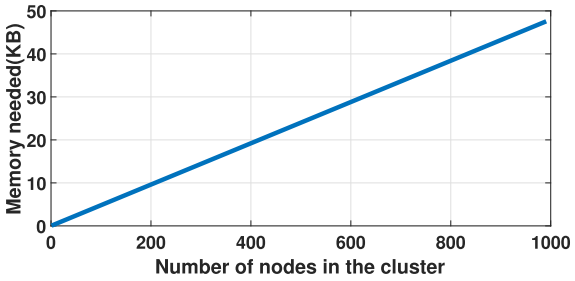

FIGURE 12. Memory size needed for the second part of the Algorithm (line 23-57).

- $8 \times (N(SR_{max} + \frac{1}{2}N + 4) + 1)$ bytes is the memory size required by the Algorithm1 from line 1-28. Figure 11 shows the memory size needed by the CH in function of SR_{max} and the number of nodes belonging to the cluster.
- $8(\times 6N + 1)$ bytes is the memory size required by the Algorithm1 from line 23-57 if we assume that the matching sensors are at maximum equal to the number of sensors in the cluster. Figure 12 shows the maximum memory size needed by the CH in function of the number of nodes belonging to the cluster.

The maximum memory size required by the CH is $8 \times Max(N(SR_{max} + \frac{1}{2}N + 4) + 1, 6N + 1)$ bytes, since the values stored in the first part of the Algorithm (1-17), could be cleared once the sensors have been matched (Algorithm 1, line 17-28).

Nevertheless, the greater is the number of nodes belonging to the same cluster, the better is the correlation among these nodes, the fewer data a sensor will sample and transmit which eventually leads to less energy consumption. Therefore, the number of sensors belonging to the same cluster should be maximized in function of its computational and memory resources.

As for the limitation of our proposed algorithm, it is evident when there is no or little correlation among the collected measurements, the sampling rate of the sensors will be always kept high. Since the role of this algorithm is to minimize the sampling rate of the sensor node, it will not be as efficient as it should be.

VI. CONCLUSION

We proposed in this paper a sampling and transmission rate adaptation algorithm for cluster-based sensor networks. This algorithm is deployed on the Cluster-Head (CH) and it operates in rounds. The latter controls the sampling rate of each individual sensor node by increasing it or decreasing it according to its spatial correlation with other sensors in the network. Moreover, we adopted and adapted a data reconstruction algorithm that is implemented on the Sink station. The latter can identify the "non-sampled" data that are not collected due to a decrease in the sampling rate of a specific sensor and it estimates them using an EM iterative approach that is capable of capturing the temporal and spatial correlation among the reported measurements. We presented experimentation that we have conducted on real sensor data of a network that was deployed at the Grand-St-Bernard pass located between Switzerland and Italy. We have compared our approach with a recent data reduction technique that combines both adaptive sampling and transmission reduction. The obtained results demonstrate that our proposal is better at reducing the energy consumption of the sensor node, thus extending the operational lifetime of the network while preserving the integrity and the quality of the data.

For future work, we aim to tune better the Algorithm deployed on the CH by incorporating other attributes to determine the optimal sampling rate of each individual sensor. Moreover, we will explore the possibility of adding a compression phase between the CH and the workstation in order to reduce more the amount of transmitted data.

REFERENCES

[1] M. Z. A. Bhuiyan, J. Wu, G. Wang, T. Wang, and M. M. Hassan, "e-sampling: Event-sensitive autonomous adaptive sensing and low-cost monitoring in networked sensing systems," *ACM Trans. Auton. Adapt. Syst.*, vol. 12, no. 1, May 2017, Art. no. 1.

[2] A. Masoum, N. Meratnia, and P. J. M. Havinga, "A decentralized quality aware adaptive sampling strategy in wireless sensor networks," in *Proc. 9th Int. Conf. Ubiquitous Intell. Comput. 9th Int. Conf. Autonomic Trusted Comput.*, Sep. 2012, pp. 298–305.

[3] H. Harb and A. Makhoul, "Energy-efficient sensor data collection approach for industrial process monitoring," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 661–672, Feb. 2018.

[4] A. Makhoul, H. Harb, and D. Laiymani, "Residual energy-based adaptive data collection approach for periodic sensor networks," *Ad Hoc Netw.*, vol. 35, pp. 149–160, Dec. 2015.

[5] 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.

[6] L. Tan and M. Wu, "Data reduction in wireless sensor networks: A hierarchical LMS prediction approach," *IEEE Sensors J.*, vol. 16, no. 6, pp. 1708–1715, Mar. 2016.

[7] S. Santini and K. Römer, "An adaptive strategy for quality-based data reduction in wireless sensor networks," in *Proc. 3rd Int. Conf. Netw. Sens. Syst.*, 2006, pp. 29–36.

[8] M. Wu, L. Tan, and N. Xiong, "Data prediction, compression, and recovery in clustered wireless sensor networks for environmental monitoring applications," *Inf. Sci.*, vol. 329, pp. 800–818, Feb. 2016.

[9] G. B. Tayeh, A. Makhoul, J. Demerjian, and D. Laiymani, "A new autonomous data transmission reduction method for wireless sensors networks," in *Proc. IEEE Middle East North Afr. Commun. Conf. (MENACOMM)*, Apr. 2018, pp. 1–6.

[10] B. Alturki, S. Reiff-Marganiec, and C. Perera, "A hybrid approach for data analytics for Internet of Things," in *Proc. 7th Int. Conf. Internet Things*, New York, NY, USA, 2017, Art. no. 1. doi: 10.1145/3131542.3131558.

[11] L. C. Monteiro, F. C. Delicato, L. Pirmez, P. F. Pires, and C. Miceli, "DPCAS: Data prediction with cubic adaptive sampling for wireless sensor networks," in *Proc. Int. Conf. Green, Pervas., Cloud Comput.* Cham, Switzerland: Springer, 2017, pp. 353–368.

[12] M. T. Nguyen, K. A. Teague, and N. Rahnavard, "CCS: Energy-efficient data collection in clustered wireless sensor networks utilizing block-wise compressive sensing," *Comput. Netw.*, vol. 106, pp. 171–185, Sep. 2016.

[13] X. Liu *et al.*, "CDC: Compressive data collection for wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 8, pp. 2188–2197, Aug. 2015.
[14] J. Azar, R. Darazi, C. Habib, A. Makhoul, and J. Demerjian, "Using DWT lifting scheme for lossless data compression in wireless body sensor networks," in *Proc. 14th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2018, pp. 1465–1470.
[15] J. Azar, A. Makhoul, R. Darazi, J. Demerjian, and R. Couturier, "On the performance of resource-aware compression techniques for vital signs data in wireless body sensor networks," in *Proc. IEEE Middle East North Afr. Commun. Conf. (MENACOMM)*, Apr. 2018, pp. 1–6.
[16] J. Yang, S. Tilak, and T. S. Rosing, "An interactive context-aware power management technique for optimizing sensor network lifetime," in *Proc. SENSORNETS*, 2016, pp. 69–76.
[17] H. Harb, A. Makhoul, R. Couturier, and S. Tawbi, "Comparison of different data aggregation techniques in distributed sensor networks," *IEEE Access*, vol. 5, pp. 4250–4263, Mar. 2017.
[18] 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, Dec. 2017, Art. no. 32.
[19] D. Kaur, G. S. Aujla, N. Kumar, A. Y. Zomaya, C. Perera, and R. Ranjan, "Tensor-based big data management scheme for dimensionality reduction problem in smart grid systems: SDN perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 10, pp. 1985–1998, Feb. 2018.
[20] A. Makhoul and H. Harb, "Data reduction in sensor networks: Performance evaluation in a real environment," *IEEE Embedded Syst. Lett.*, vol. 9, no. 4, pp. 101–104, Dec. 2017.
[21] M. N. Halgamuge, M. Zukerman, K. Ramamohanarao, and H. L. Vu, "An estimation of sensor energy consumption," *Prog. Electromagn. Res. B*, vol. 12, pp. 259–295, Sep. 2009.
[22] H.-Y. Zhou, D.-Y. Luo, Y. Gao, and D.-C. Zuo, "Modeling of node energy consumption for wireless sensor networks," *Wireless Sensor Netw.*, vol. 3, no. 1, p. 18, 2011.
[23] W. Du, F. Mieyeville, and D. Navarro, "Modeling energy consumption of wireless sensor networks by systemC," in *Proc. 5th Int. Conf. Syst. Netw. Commun. (ICSNC)*, Aug. 2010, pp. 94–98.
[24] Q. A. Bakhtiar, K. Makki, and N. Pissinou, "Data reduction in low powered wireless sensor networks," in *Wireless Sensor Networks-Technology and Applications*. Rijeka, Croatia: InTech, 2012.
[25] U. Raza, A. Camerra, A. L. Murphy, T. Palpanas, and G. P. Picco, "Practical data prediction for real-world wireless sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 8, pp. 2231–2244, Aug. 2015.
[26] N. Al-Hoqani and S.-H. Yang, "Adaptive sampling for wireless household water consumption monitoring," *Procedia Eng.*, vol. 119, pp. 1356–1365, 2015.
[27] M. Gupta, L. V. Shum, E. Bodanese, and S. Hailes, "Design and evaluation of an adaptive sampling strategy for a wireless air pollution sensor network," in *Proc. IEEE 36th Conf. Local Comput. Netw. (LCN)*, Oct. 2011, pp. 1003–1010.
[28] L. A. Villas, A. Boukerche, D. L. Guidoni, H. A. B. F. de Oliveira, R. B. de Araujo, and A. A. F. Loureiro, "An energy-aware spatio-temporal correlation mechanism to perform efficient data collection in wireless sensor networks," *Comput. Commun.*, vol. 36, no. 9, pp. 1054–1066, 2013.
[29] K. Karuppasamy and V. Gunaraj, "Optimizing sensing quality with coverage and lifetime in wireless sensor networks," *Int. J. Eng. Res. Technol.*, vol. 2, no. 2, pp. 1–7, 2013.
[30] S. Dhimal and K. Sharma, "Energy conservation in wireless sensor networks by exploiting inter-node data similarity metrics," *Int. J. Energy, Inf. Commun.*, vol. 6, no. 2, pp. 23–32, 2015.
[31] H. Harb and A. Makhoul, "Energy-efficient scheduling strategies for minimizing big data collection in cluster-based sensor networks," *Peer-to-Peer Netw. Appl.*, vol. 12, no. 3, pp. 620–634, 2019.
[32] H. Harb, A. Makhoul, A. Jaber, and S. Tawbi, "Energy efficient data collection in periodic sensor networks using spatio-temporal node correlation," *Int. J. Sensor Netw.*, vol. 29, no. 1, pp. 1–5, 2019.
[33] C. Perera, D. S. Talagala, C. H. Liu, and J. C. Estrella, "Energy-efficient location and activity-aware on-demand mobile distributed sensing platform for sensing as a service in IoT clouds," *IEEE Trans. Comput. Social Syst.*, vol. 2, no. 4, pp. 171–181, Dec. 2015.
[34] P. P. Jayaraman, C. Perera, D. Georgakopoulos, and A. Zaslavsky, "Efficient opportunistic sensing using mobile collaborative platform MOSDEN," in *Proc. 9th IEEE Int. Conf. Collaborative Comput., Netw., Appl. Worksharing*, Oct. 2013, pp. 77–86.
[35] L. Li, J. McCann, N. S. Pollard, and C. Faloutsos, "DynaMMo: Mining and summarization of coevolving sequences with missing values," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 507–516.
[36] Z. Ghahramani and M. I. Jordan, "Supervised learning from incomplete data via an EM approach," in *Proc. Adv. Neural Inf. Process. Syst.*, 1994, pp. 120–127.
[37] *Sensorscope*. Accessed: Sep. 13, 2018. [Online]. Available: https://lcav.epfl.ch/page-145180-en.html

GABY BOU TAYEH received the M.S. degree in mobile and distributed computing from the University of Franche-Comté (UFC), Belfort, France, in 2017, where he is currently pursuing the Ph.D. degree. His research interests include wireless sensor networks, LoRaWan, data science, and big data.

ABDALLAH MAKHOUL received the M.S. degree in computer science from INSA Lyon, Lyon, France, in 2005, and the Ph.D. degree from the University of Franche-Comté, Belfort, France, in 2008, where he has been an Associate Professor, since 2009. His research interests include the Internet of Things, structural health monitoring, and real-time issues in wireless sensor networks. He has been the TPC Chair and a member of several networking conferences and workshops and a Reviewer for several international journals.

CHARITH PERERA received the B.Sc. degree (Hons.) in computer science from Staffordshire University, U.K., the MBA degree in business administration from the University of Wales, Cardiff, U.K., and the Ph.D. degree in computer science from The Australian National University, Canberra, Australia. He was with the Information Engineering Laboratory, ICT Centre, CSIRO. He is currently a Lecturer (Assistant Professor) with Cardiff University, U.K. His research interests include the Internet of Things, sensing as a service, privacy, middleware platforms, and sensing infrastructure. He is a member of the ACM.

JACQUES DEMERJIAN received the Ph.D. degree in network and computer science from TELECOM ParisTech (ENST-Paris), in 2004. He is currently an Associate Professor with the Faculty of Sciences, Lebanese University, Lebanon. His main research interests include human–computer interaction, streaming data quality and summarization, mobile cloud computing, VANET, body sensor networks, data mining, and wired and wireless network security.