Abstract—The amount of sensory data manifests an explosive growth due to the increasing popularity of Wireless Sensor Networks. The scale of the sensory data in many applications has already exceeded several petabytes annually, which is beyond the computation and transmission capabilities of the conventional WSNs. On the other hand, the information carried by big sensory data has high redundancy because of strong correlation among sensory data. In this paper, we define the concept of $\epsilon$-dominant dataset, which is only a small data set and can represent the vast information carried by big sensory data with the information loss rate being less than $\epsilon$, where $\epsilon$ can be arbitrarily small. We prove that drawing the minimum $\epsilon$-dominant dataset is polynomial time solvable and provide a centralized algorithm with $O(n^3)$ time complexity. Furthermore, a distributed algorithm with constant complexity ($O(1)$) is also designed. It is shown that the result returned by the distributed algorithm can satisfy the $\epsilon$ requirement with a near optimal size. Finally, the extensive real experiment results and simulation results are carried out. The results indicate that all the proposed algorithms have high performance in terms of accuracy and energy efficiency.

I. INTRODUCTION

With the increasing popularity of Wireless Sensor Networks (WSNs), the amount of sensory data manifests an explosive growth. In many applications, the scale of the sensory data has already exceeded several petabytes (PB) annually. For instance, the volume of the climate data is larger than 2.5PB in 2010 and is expected to reach 100PB in 2020 [1]. The Large Hadron Collider experiments in Europe has 150 million sensors that deliver data 40 million times per second [2]. There are about 67000 taxis and 400 thousand electronic eyes in Beijing, resulting in more than 48PB GPS data and 1440PB other monitoring data each year. Compared with the data storage capability of the world whose growth rate is only 40% per year, the growth rate of sensory data is more than 58%. Therefore, the world already produced over twice much data as can be stored in 2011 [3]. All these facts indicates the era of Big Sensory Data (BSD) which brings us many new challenges as well as opportunities. The existing data acquisition, [4], data collection [5][6], query processing [7][8][9], and information mining algorithms [10] cannot be adopted for BSD management due to a series of special features of BSD, such as large scale, low quality, strong correlation and so on. Therefore, a group of new data acquisition, collection and computation algorithms are expected for BSD management.

Since the volume of BSD is beyond the computation and transmission capabilities of the conventional WSNs, one feasible solution is to dramatically reduce the amount of sensory data involved in the computation, which is known as “Do More with Less”. Based on such a motivation, several sampling based algorithms [11][12][13] were proposed. These algorithms sample a small portion of sensory data to answer queries, where the size of the involved data set is determined by the user-specified precision requirement. Although the sampling based algorithms are efficient and effective for processing queries in WSNs for BSD, the characteristics and correlations of sensory data are missing during sampling and collection periods. Thus, they are only suitable for some simple queries such as aggregation, and unable to recover the original information with high precision. Another option to reduce the amount of sensory data is compression. Many sensory data compression techniques were proposed up to now, including the sketch based compression [14], linear regression based compression [15], source coding based compression [16], information entropy based compression [17] et al. However, for most existing compression methods, the computation task on the compressed data cannot be carried out without a decompression process, which results in additional energy and time consumption. Other compression techniques, such as the Sketch based ones, can only deal with very trivial queries.

The above facts motivates us to investigate a new data reduction algorithm which can support both recovery and efficient computation of BSD. Many research works reveal that sensory data are strong correlated since the monitored physical world always varies continuously in space and time [18][19]. Such strong correlation will incur high redundancy in BSD, i.e. the vast majority of information carried by BSD can be represented by a small data set referred to as a dominant dataset of BSD. BSD management on a dominant dataset instead of original BSD can significantly reduce the costs of storing, transmitting and processing. Meanwhile, it can easily support data recovery and computation without decompression. In this paper, we investigate how to draw a dominant dataset from BSD in order to support data computation and management. The main contributions of this paper are as follows.

1. The definitions of the information loss rate, $\epsilon$-dominant dataset and minimum $\epsilon$-dominant dataset are firstly introduced.

2. We prove that drawing the minimum $\epsilon$-dominant dataset can be solved in polynomial time, and propose a centralized

Drawing Dominant Dataset From Big Sensory Data in Wireless Sensor Networks

Siyao Cheng
School of Computer Science and Tech
Harbin Institute of Technology
Email: csy@hit.edu.cn

Zhipeng Cai
Department of Computer Science
Georgia State University
Email: zcai@cs.gsu.edu

Jianzhong Li
Xiaolin Fang
School of Computer Science and Tech
Harbin Institute of Technology
Email: lijzh@hit.edu.cn, xlfou@hit.edu.cn

978-1-4799-8381-0/15/$31.00 ©2015 IEEE
algorithm to solve it in $O(n^3)$ time. The correctness of the algorithm is proven, and the computation and communication complexities of the algorithm are analyzed.

(3). To reduce the complexity, a distributed algorithm with constant complexity ($O(1)$) is designed. We prove that the result returned by the distributed algorithm can satisfy the $\epsilon$ requirement specified by a user with a near optimal size. The detailed analysis on the computation and communication complexities of the distributed algorithm is also provided.

(4). A distributed algorithm of how to maintain the correlations between different sensor nodes are also developed. Besides, the extensive experiments were carried out, and the results show that the proposed algorithms have high performance in terms of accuracy and energy efficiency.

The organization of the paper is as follows. Section II presents the problem definition. Section III introduces the mathematic foundations, which helps with proving that drawing the minimum $\epsilon$-dominant dataset is polynomial time solvable. Section IV provides the correlation maintenance algorithms, which is necessary for drawing an $\epsilon$-dominant dataset. Section V illustrates the centralized and distributed algorithms for retrieving an $\epsilon$-dominant dataset. Section VI presents the experimental results. Section VII is devoted to the related works and Section VIII concludes the paper.

### II. PROBLEM DEFINITION

Assume that there are $n$ sensor nodes in a WSN, where $V = \{1, 2, ..., n\}$ denotes the set of the sensor nodes. For each sensor node $i (1 \leq i \leq n)$, it collects data continuously from the physical world with the sampling frequency $f$. Since both of $n$ and $f$ are large, the amount of the sensory data generated by a given WSN is quite huge during a long period.

We further assume the clocks of all the sensor nodes are synchronized, which can be achieved through some well-established techniques, such as [20] and [21]. Let $T_{w}^{(s)}$ and $T_{w}^{(f)}$ denote the start and end time of the $w$-th time window, where $w \geq 1$. Let $m = (T_{w}^{(f)} - T_{w}^{(s)}) \times f$ denote the size of each time window, and $t_{w1}, t_{w2}, ..., t_{wn}$ be the sampling time slots, i.e. $t_{w(j+1)} - t_{wj} = 1/f$ for any $1 \leq j < m - 1$, $t_{w1} = T_{w}^{(s)}$, and $t_{wn} = T_{w}^{(f)}$.

Let $s_{it}$ be the sensed value of sensor node $i (1 \leq i \leq n)$ at time $t$. Thus, the sensed values collected by sensor node $i$ in the time window $[T_{w}^{(s)}, T_{w}^{(f)}]$ can be denoted by a vector $S_{i}^{(w)}$, where $S_{i}^{(w)} = [s_{itw1}, s_{itw2}, ..., s_{itwn}]^T$. It is reasonable to assume that $m$ is large enough so that there at least exist two distinct values in $S_{i}^{(w)}$ for any $1 \leq i \leq n$. All the sensed values collected by the WSN in $[T_{w}^{(s)}, T_{w}^{(f)}]$ can be denoted by $S^{(w)} = [S_{1}^{(w)}, S_{2}^{(w)}, ..., S_{n}^{(w)}]$, where $S^{(w)}$ is an $m \times n$ matrix. Since the size of a WSN ($n$) is usually quite large, the transmission and storage of $S^{(w)}$ will cost a huge amount of energy and may be impossible in some cases. Therefore, we expect to find a subspace, denoted by $[U_{1}, U_{2}, ..., U_{p}]$, so that $S^{(w)}$ can project in this subspace and get a smaller data matrix, where $p \ll n$. Meanwhile, it is desired that the information lost by such a projection is minimized.

In this paper, the information loss rate of a projection is measured by the proportion of the data characteristic it drops. Thus, the data characteristic must be defined firstly. Considering the metrics of different types of sensory data are not the same, e.g. the metric of light data is different from that of the temperature data. It is difficult to evaluate the data characteristic of sensory data from different sensor nodes. Therefore, normalization of sensory data is required. Based on the normalization method [22], the definition of data characteristic vector and matrix are given as follows.

**Definition 1 (Data Characteristic Vector and Matrix).** For any $i (1 \leq i \leq n)$, $Z_{i}^{(w)}$ is called the data characteristic vector of $S_{i}^{(w)}$, if $Z_{i}^{(w)}$ is the normalization of $S_{i}^{(w)}$, that is, $Z_{i}^{(w)}$ satisfies $Z_{i}^{(w)} = \frac{1}{\sigma_{i}} \{S_{i}^{(w)} - [S_{1}^{(w)}, S_{2}^{(w)}, ..., S_{n}^{(w)}]^T\}$, where $S_{i}^{(w)} = [s_{itw1}, s_{itw2}, ..., s_{itwn}]^T$, $\sigma_{i} = 1/m \sum_{k=1}^{m} s_{itw_k}$ and $\sigma_{i} = \sqrt{\sum_{k=1}^{m} (s_{itw_k} - \overline{S_{i}^{(w)}})^2}$. The data characteristic matrix $Z^{(w)}$ satisfies that $Z^{(w)} = [Z_{1}^{(w)}, Z_{2}^{(w)}, ..., Z_{n}^{(w)}]^T$.

According to Definition 1, we also have $Z^{(w)} = (S^{(w)} - \text{Avg}(S^{(w)})) \text{Diag}(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, ..., \frac{1}{\sigma_n})$, where

$$\text{Avg}(S^{(w)}) = \begin{bmatrix}
\bar{S}_1^{(w)} & \bar{S}_2^{(w)} & \cdots & \bar{S}_n^{(w)} \\
\bar{S}_1^{(w)} & \bar{S}_2^{(w)} & \cdots & \bar{S}_n^{(w)} \\
\vdots & \vdots & \ddots & \vdots \\
\bar{S}_1^{(w)} & \bar{S}_2^{(w)} & \cdots & \bar{S}_n^{(w)}
\end{bmatrix}_{m \times n}$$

$$\text{Diag}(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, ..., \frac{1}{\sigma_n}) = \begin{bmatrix}
\frac{1}{\sigma_1} & 0 & \cdots & 0 \\
0 & \frac{1}{\sigma_2} & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \frac{1}{\sigma_n}
\end{bmatrix}_{n \times n}$$

In the rest of the paper, we use data characteristic matrix $Z^{(w)}$ during the projection since it has no metric influence.

Let $Z_{(i)}^{(w)}$ be the $i$-th row of $Z^{(w)}$, i.e. $Z_{(i)}^{(w)} = [Z_{i1}^{(w)}, Z_{i2}^{(w)}, ..., Z_{in}^{(w)}]^T$, where $Z_{i}^{(w)}$ is an $n$-dimensional vector and stands for the normalization sensed values from all the sensors at $t_{wi}$. Since the sensory data are strong correlated in space, there exists many redundant information in the sensed values sampled at each time slot. Thus, we hope to find a low-dimensional space to map $\{Z_{(i)}^{(w)} | 1 \leq i \leq m\}$ in such space.

**Fig. 1:** Projection of $Z_{(i)}^{(w)}$ on $\text{span}(U_{r})$

Given a unit vector $U_{r}$, $\text{span}(U_{r})$ is the space defined by it, which is a line as shown in Fig.1. The projection of $Z_{(i)}^{(w)}$ on line $\text{span}(U_{r})$ is equal to $(Z_{(i)}^{(w)})^T U_{r}$. For any $i (1 \leq i \leq m)$, where $(Z_{(i)}^{(w)})^T$ is the transpose of $Z_{(i)}^{(w)}$. The data characteristic dropped by such projection is measured by the square of the Euclidean distance between $Z_{(i)}^{(w)}$ and
The definition of the information loss rate is as follows.

Definition 2. (Information Loss Rate) Let $U_r$ be a unit vector and $Z^{(w)} = \sum_{i=1}^{m}Z^{(i)}$, $Z^{(w)} = \sum_{i=1}^{m}Z^{(i)}$ be the data characteristic matrix of $S^{(w)}$, the information loss rate of projecting $Z^{(w)}$ on $span(U_r)$, denoted by $R(U_r)$, satisfies that

$$R(U_r) = \frac{\sum_{i=1}^{m}||Z^{(i)}_w||^2}{\sum_{i=1}^{m}||Z^{(i)}||^2}. \quad \text{Let} \ U_1, U_2, ..., U_p \text{ be } p \text{ unit orthogonal vectors, the information loss rate of projecting } Z^{(w)} \text{ on } span(U_1, U_2, ..., U_p), \text{ denoted by } R(U_1, U_2, ..., U_p), \text{ satisfies that } R(U_1, U_2, ..., U_p) = 1 - \sum_{r=1}^{n}(1 - R(U_r)), \text{ where } span(U_r) \text{ and } span(U_1, U_2, ..., U_p) \text{ are the spaces defined by } U_r \text{ and } \{U_1, U_2, ..., U_p\}, \text{ respectively.}$$

In Definition 2, the intuition of requiring $U_1, U_2, ..., U_p$ to be orthogonal with each other is that the orthogonal vectors can reduce the information reduction as much as possible during projection. Based on Definition 2, the definition of the $\epsilon$-subspace and $\epsilon$-dominant dataset are given in Definition 3 and Definition 4, respectively.

Definition 3. ($\epsilon$-subspace) Let $span(U_1, U_2, ..., U_p)$ denote a space whose bases are $U_1, U_2, ..., U_p$. $span(U_1, U_2, ..., U_p)$ is called $\epsilon$-subspace iff the information loss rate of projecting $Z^{(w)}$ on $span(U_1, U_2, ..., U_p)$ is less than $\epsilon$, i.e.,

$$R(U_1, U_2, ..., U_p) \leq \epsilon.$$ 

Definition 4. ($\epsilon$-dominant dataset) If $span(U_1, U_2, ..., U_p)$ is an $\epsilon$-subspace, then $D^p_{\epsilon,w}(U_1, ..., U_p) = Z^{(w)} \times [U_1, U_2, ..., U_p]$ is called an $\epsilon$-dominant dataset of $Z^{(w)}$ and

$$D^p_{\epsilon,w}(U_1, ..., U_p) = [Z^{(w)} \times \text{Diag}(\sigma_1, \sigma_2, ..., \sigma_n) + \text{Avg}(S^{(w)})] \times [U_1, U_2, ..., U_p]$$

is called an $\epsilon$-dominant dataset of $S^{(w)}$.

The intuition of the above is as follows. We do pan- ning and zooming during the normalization, therefore, the opposite operations should be done to obtain the $\epsilon$-dominant dataset of $S^{(w)}$. Meanwhile, since $Z^{(w)} = (S^{(w)} - \text{Avg}(S^{(w)})) \text{Diag}(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, ..., \frac{1}{\sigma_n})$, $D^p_{\epsilon,w}(U_1, ..., U_p) = S^{(w)}[U_1, U_2, ..., U_p]$.

Based on Definition 4, the problem of drawing an $\epsilon$-dominant dataset in the current time window is defined as follows. Given $\epsilon$, $m$, $t_e$, $S^{(w)}$, $f$ and $n$, the problem seeks an $\epsilon$-dominant dataset of $S^{(w)}$, which satisfies Definition 4, in $[t_e - m/f, t_e]$, where $t_e$ is the current time and $[t_e - m/f, t_e]$ is the current time window with size $m$.

However, the numbers of $\epsilon$-subspaces and $\epsilon$-dominant datasets are infinite. Among thousands of these results, we want to draw the one with the smallest size to minimize the cost of transmitting and storing an $\epsilon$-dominant dataset. Therefore, we call an $\epsilon$-subspace and an $\epsilon$-dominant dataset are minimum if they have the smallest size. The formal definition is given as follows.

Definition 5. (Minimum $\epsilon$-subspace) The subspace, $span(V_1, V_2, ..., V_q)$, is called the minimum $\epsilon$-subspace if and only if 1) $span(V_1, V_2, ..., V_q)$ is an $\epsilon$-subspace; 2) there is $q \leq p$ for any other $\epsilon$-subspace $span(U_1, U_2, ..., U_p)$.}

Thus, $D^p_{\epsilon,w}(V_1, ..., V_q) = S^{(w)}[V_1, ..., V_q]$ is called the minimum $\epsilon$-dominant dataset if $span(V_1, V_2, ..., V_q)$ is the minimum $\epsilon$-subspace. Similarly, the problem of drawing the Minimum $\epsilon$-dominant dataset is as follows. Given $\epsilon$, $m$, $t_e$, $S^{(w)}$, $f$ and $n$, we want to find the Minimum $\epsilon$-dominant dataset of $S^{(w)}$ in $[t_e - m/f, t_e]$.

To solve this problem, we need to find the minimum $\epsilon$-subspace firstly, which will be discussed in Section III.

### III. Mathematic Foundation

Before discussing how to obtain the minimum $\epsilon$-subspace, we firstly give the definition of correlation coefficient matrix.

Definition 6. (Correlation Coefficient Matrix) The correlation between sensor node $i$ and sensor node $u$ in the time window $[T^{u,w}_{i}, T^{u,w}_{f}]$, denoted by $r^{(u)}_{i,u}$, is

$$r^{(u)}_{i,u} = \frac{\sum_{k=1}^{m}(s_{i,u,k} - \bar{S}_{i,u})(s_{i,u,k} - \bar{S}_{i,u})}{\sqrt{\sum_{k=1}^{m}(s_{i,u,k} - \bar{S}_{i,u})^2} \sqrt{\sum_{k=1}^{m}(s_{i,u,k} - \bar{S}_{i,u})^2}},$$

where $m$ is the number of the sampling time slots in $[T^{u,w}_{i}, T^{u,w}_{f}]$, $\bar{S}_{i,u} = \frac{1}{m} \sum_{k=1}^{m} s_{i,u,k}$, and $\bar{S}_{i,u} = \frac{1}{m} \sum_{k=1}^{m} s_{i,u,k}$. Furthermore, the $n \times n$ matrix $C^{w} = [r^{(u)}_{i,u}]_{1\leq i,u \leq n}$ is called the Correlation Coefficient Matrix of $S^{(w)}$ in $[T^{u,w}_{i}, T^{u,w}_{f}]$.

Theorem 1. For any unit vector $U_r$, the information loss rate of projecting $Z^{(w)}$ on $span(U_r)$, denoted by $R(U_r)$, satisfies that

$$R(U_r) = 1 - \frac{U_r^T C^{w} U_r}{n}.$$ 

The proof of theorem 1 is given in our technique report [23] due to the space limitation. As $r^{(u)}_{i,u} = r^{(u)}_{i,u}$ for $\forall 1 \leq i \leq u \leq n$, $C^{w}$ is a real symmetric matrix, and $C^{w}$ has $n$ real eigenvalues and their associated eigenvectors are linear independent [24]. Let $\lambda_1 \geq ... \geq \lambda_n$ denote the eigenvalues of $C^{w}$, and $I_1, I_2, ..., I_n$ be their associated eigenvectors where the orthonormalization process has been performed, i.e. $I_1$ and $I_2$ are orthogonal with each other for any $1 \leq i \neq j \leq n$ and $||I_i|| = 1$. We have the following lemma and theorem.

**Lemma 1.** The eigenvalues of $C^{w}$ satisfies that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq 0$.

**Theorem 2.** For any $1 \leq p \leq n$ and any other subspace $span(V_1, V_2, ..., V_p)$, we have $R(V_1, V_2, ..., V_p) \geq R(I_1, I_2, ..., I_p) = 1 - \frac{\sum_{i=1}^{p} \lambda_i}{n}$, where $R(I_1, I_2, ..., I_p)$ and $R(V_1, V_2, ..., V_p)$ are information loss rates of projecting $Z^{(w)}$ on $span(I_1, I_2, ..., I_p)$ and $span(V_1, V_2, ..., V_p)$. 

The proof of Lemma 1 and Theorem 2 is also included in our technique report [23]. Based on such theorem, the following Corollary 1 is easy to be proved.

**Corollary 1.** Let $q = \min\{\lambda_i - \frac{\sum_{i=1}^{p} \lambda_i}{n} \leq \epsilon\}$, then $span(I_1, I_2, ..., I_q)$ is the minimum $\epsilon$-subspace.

Based on Corollary 1, the minimum $\epsilon$-subspace can be determined by the eigenvalues and the eigenvectors of $C^{w}$. Since these values can be obtained in polynomial time [25], the problem of drawing the minimum $\epsilon$-dominant dataset from $S^{(w)}$ is also polynomial time solvable.
IV. THE CORRELATION COEFFICIENT MATRIX
MAINTENANCE ALGORITHM

According to the analysis in section 3, the correlation coefficient matrix \( C^{(w)} \) is required to be obtained before determining the minimum \( \epsilon \)-dominant dataset in a given time window. Moreover, maintaining \( C^{(w)} \) during the network lifetime of a WSN is also necessary as it gives a measure for evaluating the correlations among sensors. Therefore, this section provides an accurate distributed algorithm for maintaining \( C^{(w)} \).

A. Algorithm Description

Since \( C^{(w)} = \{ r_{ij}^{(w)} \}_{1 \leq i, j \leq n} \), each sensor node \( i (1 \leq i \leq n) \) only needs to maintain \( \{ r_{ij}^{(w)} \}_{1 \leq j \leq n} \), where \( r_{ij}^{(w)} \) is the correlation between sensor nodes \( i \) and \( j \). As sensory data is spatial-correlated, the correlation between \( i \) and \( j \) is very weak and can be ignored if their distance is large.

Based on such an observation, we set \( r_{ij}^{(w)} = 0 \) if the distance between \( i \) and \( j \) is larger than a given threshold \( d \), where \( d \) is determined by the sensing radius of a sensor node.

The definition of neighbor is given as follows.

Definition 7. (Neighbor). Sensor node \( j \) is a neighbor of sensor node \( i \) if \( \text{dis}(i, j) \leq d \), and \( N_i = \{ j | \text{dis}(i, j) \leq d \} \) is called the neighbor set of sensor node \( i \), where \( \text{dis}(i, j) \) denotes the distance between \( i \) and \( j \).

Clearly, each sensor node \( i (1 \leq i \leq n) \) only needs to maintain \( \{ r_{ij}^{(w)} \}_{j \in N_i} \).

The algorithm for maintaining the correlation coefficient matrix \( C^{(w)} \) in-network can be divided into two phases, initial phase and maintenance phase. The initial phase happens in the first time window \([T_1^{(s)}, T_1^{(f)}]\). Afterward, each sensor node only needs to keep the maintenance phase while the time window is sliding. For each sensor node \( i (1 \leq i \leq n) \), the initial phase has the following four steps.

1. **Step 1.** Sensor node \( i \) samples the sensed values, \( s_{t_{11}}, s_{t_{12}}, ..., s_{t_{1m}} \), in the first time window \([T_1^{(s)}, T_1^{(f)}] \) with frequency \( f \), where \( t_{1l} - t_{1(l-1)} = 1/f \) for any \( 2 \leq l \leq m \), \( t_{11} = T_1^{(s)} \) and \( t_{1m} = T_1^{(f)} \).

2. **Step 2.** Sensor node \( i \) stores \( s_{t_{11}}, s_{t_{12}}, ..., s_{t_{1m}} \), computes \( |S_i^{(1)}|^2 = \sum_{l=1}^{m} s_{t_{1l}}^2 \) and \( \text{Sum}_i^{(1)} = \sum_{l=1}^{m} s_{t_{1l}} \), and broadcasts \( s_{t_{11}}, s_{t_{12}}, ..., s_{t_{1m}} \) to its neighbors in \( N_i \).

3. **Step 3.** Sensor node \( i \) also receives the data messages from its neighbors. For any \( j \in N_i \), it computes and stores \( IP_{ij}^{(1)} = \sum_{l=1}^{m} s_{t_{1l}} s_{t_{jl}} \), \( |S_j^{(1)}|^2 = \sum_{l=1}^{m} s_{t_{jl}}^2 \) and \( \text{Sum}_j^{(1)} = \sum_{l=1}^{m} s_{t_{jl}} \).

4. **Step 4.** Finally, sensor node \( i \) can calculate the correlation between sensor \( j (j \in N_i) \) and itself as follows:

\[
\rho_{ij}^{(1)} = \frac{IP_{ij}^{(1)} - \frac{\text{Sum}_i^{(1)} \text{Sum}_j^{(1)}}{|S_i^{(1)}|^2 - \frac{1}{m} (\text{Sum}_i^{(1)})^2}}{|S_j^{(1)}|^2 - \frac{1}{m} (\text{Sum}_j^{(1)})^2}
\]

The maintenance phase keeps the correlations stored in-network to be valid all the time. Let \([T_w^{(s)}, T_w^{(f)}]\) be the current time window, where \( w > 1 \). Since the time window is sliding, \( T_w^{(s)} = T_w^{(s)} + 1/f \) and \( T_w^{(f)} = T_w^{(f)} + 1/f \), where \( f \) is the sampling frequency of each sensor node. For each sensor node \( i (1 \leq i \leq n) \), a new sensed value, denoted by \( s_{t_{w1}}^{(1)} \), will be sampled and an old sensed value, denoted by \( s_{t_{w-1}^{(1)}} \), becomes out of date while the time window is sliding. Sensor node \( i \) will carry out the following operations in the maintenance phase to update the coefficients between it and its neighbors.

1. **Step 1.** Sensor node \( i \) samples \( s_{t_{w1}}^{(1)} \) from the physical world, and updates \( \text{Sum}_i^{(w)} \) and \( |S_i^{(w)}|^2 \) by \( \text{Sum}_i^{(w)} = \text{Sum}_i^{(w-1)} - s_{t_{w-1}^{(1)}} + s_{t_{w1}}^{(1)} \) and \( |S_i^{(w)}|^2 = |S_i^{(w-1)}|^2 - s_{t_{w-1}^{(1)}}^2 + s_{t_{w1}}^{(1)}^2 \).

Then, it broadcasts \( s_{t_{w1}}^{(1)} \) and \( s_{t_{w-1}^{(1)}} \) to its neighbors.

2. **Step 2.** Sensor node \( i \) also receives the sensory data information from its neighbors. For any \( j \in N_i \), sensor node \( i \) updates \( IP_{ij}^{(w)} \), \( \text{Sum}_j^{(w)} \), and \( |S_j^{(w)}|^2 \) by \( IP_{ij}^{(w)} = IP_{ij}^{(w-1)} - s_{t_{w-1}^{(1)}} s_{t_{w-1}^{(1)}} + s_{t_{w1}}^{(1)} s_{t_{w1}^{(1)}} \), \( \text{Sum}_j^{(w)} = \text{Sum}_j^{(w-1)} - s_{t_{w-1}^{(1)}} + s_{t_{w1}^{(1)}} \), and \( |S_j^{(w)}|^2 = |S_j^{(w-1)}|^2 - s_{t_{w-1}^{(1)}}^2 + s_{t_{w1}^{(1)}}^2 \).

3. **Step 3.** Finally, the correlation \( r_{ij}^{(w)} \) can be updated by

\[
r_{ij}^{(w)} = \frac{IP_{ij}^{(w)} - \frac{\text{Sum}_i^{(w)} \text{Sum}_j^{(w)}}{|S_i^{(w)}|^2 - \frac{1}{m} (\text{Sum}_i^{(w)})^2}}{|S_j^{(w)}|^2 - \frac{1}{m} (\text{Sum}_j^{(w)})^2}
\]

for any \( j \in N_i \).

The correctness of maintenance phase is guaranteed by the following theorem, its proof is given in [23].

**Theorem 3.** The correlations \( \{ r_{ij}^{(w)} \}_{1 \leq i, j \leq n} \) returned by maintenance phase are coincident with Definition 6. □

B. Complexity Analysis

Let \( e_1 \) and \( e_2 \) be the energy costs of a sensor node for sending and receiving one byte, and \( d_{\text{max}} = \max_{1 \leq i \leq n} |N_i| \).

The maximum computation and communication complexities of a sensor node in the initial phase are \( O(md_{\text{max}}) \) and \( O(m(e_1 + e_2 d_{\text{max}})) \), respectively. Since the transmission radius is much larger than the sensing radius for most sensor products [26], the broadcast message can reach the neighbor nodes by one or two hops of transmission. Therefore, the energy consumed by each sensor node to broadcast its sensed values in the initial phase is low.

The maximum computation and communication complexities of a sensor node are \( O(d_{\text{max}}) \) and \( O(e_1 + e_2 d_{\text{max}}) \) in the maintenance phase, respectively. The cost of the maintenance phase is extremely low since the stored history information is used sufficiently. Meanwhile, since some routing data should be exchanged between neighbor nodes periodically to keep the connection of the whole network, \( s_{t_{w1}}^{(1)} \) and \( s_{t_{w-1}^{(1)}} \) can be added to the routing data. Therefore, the additional cost generated by the maintenance phase can be ignored.

V. DOMINANT DATASET DRAWING ALGORITHMS

A. Centralized Algorithm

The centralized algorithm to draw the \( \epsilon \)-dominant dataset in a given time window \([T_w^{(s)}, T_w^{(f)}]\) contains five steps.

First, the sensors in the network are organized as a spanning tree rooted at the sink. The sink broadcasts a command along the spanning tree when it wants the \( \epsilon \)-dominant dataset.

Second, let \( T_w^{(f)} = t_c \) and \( T_w^{(s)} = T_w^{(f)} - m/f \) be the end and start time of the current time window. Each sensor node
methods is based on the Lanczos algorithm [27]. Being similar
to the centralized algorithm, the first step of drawing the \( \epsilon \)-dominant dataset is to determine a \( \epsilon \)-subspace.

Let \( V_1, V_2, \ldots, V_p \) satisfy that

\[
V_{j+1} = \frac{W_j}{\beta_{j+1}}
\]

for any \( j > 1 \), where

\[
\beta_{j+1} = ||W_j||, \quad W_j = C^{(w)} V_j - \alpha_j V_j - \beta_j V_{j-1}, \quad \alpha_j = V_j^T C^{(w)} V_j,
\]

\( \beta_1 = 0 \) and \( ||V_1|| = 1 \) is a starting vector.

The following two theorems show that \( V_1, V_2, \ldots, V_p \) form
the standard orthogonal bases of a subspace with size \( p \), and the information loss rate of projecting \( Z^{(w)} \) to subspace \( \text{span}(V_1, \ldots, V_p) \) satisfies that \( R(V_1, \ldots, V_p) = 1 - \frac{\sum_{j=1}^{n} \alpha_j}{n} \).

**Theorem 4.** \( V_1, V_2, \ldots, V_p \) are the standard orthogonal bases of a subspace with size \( p \), where \( p \leq n \).

**Theorem 5.** \( R(V_1, V_2, \ldots, V_p) = 1 - \frac{\sum_{j=1}^{n} \alpha_j}{n} \), where \( R(V_1, V_2, \ldots, V_p) \) denotes the information loss rate of projecting \( Z^{(w)} \) to subspace \( \text{span}(V_1, V_2, \ldots, V_p) \), where \( p \leq n \).

The proof of Theorems 4 and 5 are given in [23]. According to them, we could use the subspace \( \text{span}(V_1, V_2, \ldots, V_p) \), to define the \( \epsilon \)-subspace, and the information loss rate of projecting \( Z^{(w)} \) to subspace \( \text{span}(V_1, V_2, \ldots, V_p) \) can be measured by

\[
1 - \frac{\sum_{j=1}^{n} \alpha_j}{n}.
\]

According to Formulas (5), (6), (7) and (8), \( V_{j+1} \) and \( \alpha_{j+1} \) only depend on \( V_j \) and \( V_{j-1} \). Thus, the bases of such \( \epsilon \)-subspace and the information loss rate of projection can be determined iteratively, which makes it easier to obtain the \( \epsilon \)-subspace and \( \epsilon \)-dominant dataset.

2) Description of Algorithm: Specifically, the detailed description of the distributed algorithm is as follows.

**Step 1.** Similarly, all the sensors in the network are organized as a spanning tree rooted at the sink. The sink broadcasts a command and \( n \) when it is the time to draw the \( \epsilon \)-dominant dataset, where \( n \) is the size of the network.

**Step 2.** Let \( T_{w}^{(s)} \) and \( T_{f}^{(f)} \) be the current time window, where \( T_{w}^{(s)} = t_c - m/f \) and \( T_{f}^{(f)} = t_c \). The starting vector \( V_1 \) is set to be \( \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}} \). Therefore, each sensor node \( i \) only needs to set \( V_{i1} = \frac{1}{\sqrt{n}} \) when it receives \( n \) transmitted from the sink node. Finally, let \( p = 1, \beta_1 = 0 \) and \( \Sigma_{m_{n_1}} = 0 \).

**Step 3.** Each sensor node \( i \) broadcasts \( V_{ip} \) to its neighbors and receives \( V_{jp} \) \( j \in N_i \) from its neighbors. Since each sensor node \( i \) is only correlated with its neighbors. Therefore, \( \alpha_{p}^{(i)} \) can be calculated by \( \alpha_{p}^{(i)} = V_{ip} \times \sum_{j \in N_i} r_{ij}^{(w)} V_{jp} \). \( \alpha_{p}^{(i)} \) is transmitted to the sink node along the spanning tree. All \( \alpha_{p}^{(i)} \) \( 0 \leq i \leq n \) are added together during transmission. Finally, the sink will obtain \( \alpha_{p} = \sum_{i=1}^{n} \alpha_{p}^{(i)} \).

**Step 4.** The sink updates \( \Sigma_{m_{n}} \) by \( \Sigma_{m_{n}} = \Sigma_{m_{n}} + \alpha_{p} \). The sink stores \( \Sigma_{m_{n}} \). If \( 1 - \Sigma_{m_{n}} > \epsilon \), the sink broadcasts \( \alpha_{p} \) along the spanning tree. Otherwise, the sink broadcasts a command to collect \( \epsilon \)-dominant dataset.

**Step 5.** If the sensor nodes receive \( \alpha_{p} \) transmitted from the sink node, the following operations should be carried out.
1) Each sensor node \( i \) computes \( W_{ip} \) by \( W_{ip} = \sum_{j \in N_i} r_{ij} V_{jp} - \alpha_p V_{ip} - \beta_p V_{(i-1)p} \). Sensor node \( i \) stores \( V_{ip} \) and \( W_{ip} \), and calculates \( W_{ip}^2 = W_{ip} W_{ip} \).

2) \( \{W_{ip}^2\}_{1 \leq i \leq n} \) are transmitted and aggregated. Finally, the sink obtains \( \sum_{i=1}^n W_{ip}^2 \). Let \( \beta_{p+1} = \sqrt{\sum_{i=1}^n W_{ip}^2} \) and broadcast \( \beta_{p+1} \) to the network.

3) Each sensor \( i \) calculates \( V_{i(p+1)} = \frac{W_{ip}}{\beta_{p+1}} \). It stores \( V_{i(p+1)} \). Let \( p = p + 1 \) and go to Step 3.

**Step 6.** If the sensor nodes receive the command to collect \( \epsilon \)-dominant dataset transmitted from the sink node, then,

1) Each sensor node \( i \) computes \( D_i \) by \( D_i = [V_{ij}S_{ij}^{(w)}, V_{i2}S_{i2}^{(w)}, ..., V_{ip}S_{ip}^{(w)}]_{m \times p} \). where \( S_{ij}^{(w)} = [s_{it_1}, s_{it_2}, ..., s_{it_m}]^T \) denotes the sensed values of sensor node \( i \) in time window \( [T_w(s), T_w(f)] \).

2) \( \{D_i\}_{1 \leq i \leq n} \) from different sensor nodes are transmitted and aggregated along the spanning tree towards the sink. Finally, the sink obtains \( D_p^{(w)} = \sum_{i=1}^n D_i \), which is the \( \epsilon \)-dominant dataset in \( [T_w(s), T_w(f)] \).

Theorem 5 guarantees the correctness of the above algorithm. Next the complexities of the algorithm is analyzed.

3) **Complexities of the Algorithm:** Let \( e_1 \) and \( e_2 \) be the energy cost of a sensor for sending and receiving one byte, and \( d_{max} = max_{1 \leq i \leq n} |N_i| \).

Obviously, in the first two steps, the computation complexity of each node is \( O(1) \) and the max communication complexity of each node is \( O(e_1 + e_2) \) since only \( T_w^{(s)}, T_w^{(f)} \) and \( V_{i1} \) needs to be computed and \( n \) and the command needs to be broadcasted. In the third step, the maximum computation and communication costs are \( O(d_{max}) \) and \( O(e_1 + e_2) \) respectively for each node at each round since \( \{V_{jp}\}_{j \in N_i} \) requires to be collected from the neighbors, and \( \alpha_p \) needs to be transmitted and aggregated. If there exist \( p \) rounds, the total computation and communication complexities of each node are at most \( O(d_{max} p) \) and \( O((e_1 + e_2) d_{max}) \) in Step 3.

Similarly, the total computation and communication complexities of each node are at most \( O(d_{max} p) \) and \( O(e_1 + e_2 p) \) in the fourth and fifth steps if there exists \( p \) round since \( W_{ip}^2 \) needs to be calculated and transmitted by each sensor node at each round. In the sixth step, \( \{D_i\}_{1 \leq i \leq n} \) needs to be computed, transmitted and aggregated, the total computation and communication complexities of each node are \( O(mp) \) and \( O(mp(e_1 + e_2)) \) respectively.

In summary, the maximum computation and communication complexities of each node are \( O(mp + d_{max} p) \) and \( O(p(me_1 + (d_{max} + m)e_2)) \). Since \( m, d_{max} \) and \( p \) are much smaller than \( n \), they can be regarded as constants. Thus, the complexity and energy cost of the distributed algorithm is quite small comparing with the centralized algorithm.

4) **Near Optimal Property:** However, \( \text{span}(V_1, V_2, ..., V_p) \) is not the minimum \( \epsilon \)-subspace, the size of the \( \epsilon \)-dominant dataset determined by it is also not minimized. Then, we wonder how close the \( \epsilon \)-dominant dataset returned by the distributed algorithm is to the minimum one.

According to Corollary 1, the minimum \( \epsilon \)-dominant dataset is a data matrix with size \( m \times q \), where \( q = \min\{l|1 - \sum_{j=1}^n \lambda_j / n \leq \epsilon\} \). Meanwhile, the \( \epsilon \)-dominant dataset returned by the distributed algorithm is an \( m \times p \) matrix, where \( p = \min\{l|1 - \sum_{j=1}^p \alpha_j \leq \epsilon\} \). Therefore, we hope that the difference between \( \sum_{j=1}^p \alpha_j \) and \( \sum_{j=1}^n \lambda_j \) is small enough.

To verify this fact, we introduce a new variance, denoted by \( T \), and \( T \) is a \( p \times p \) matrix that satisfies the following formula,

\[
T = \begin{pmatrix}
\alpha_1 & \beta_2 & \beta_3 & \ldots & \beta_{p-1} & \alpha_{p-1} & \beta_p & \alpha_p \\
\beta_2 & \alpha_2 & \beta_3 & \ldots & \beta_{p-1} & \alpha_{p-1} & \beta_p & \alpha_p \\
\beta_3 & \beta_3 & \alpha_3 & \ldots & \beta_{p-1} & \alpha_{p-1} & \beta_p & \alpha_p \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\beta_{p-1} & \beta_{p-1} & \beta_{p-1} & \ldots & \alpha_{p-1} & \beta_p & \alpha_p & \alpha_p \\
\alpha_{p-1} & \alpha_{p-1} & \alpha_{p-1} & \ldots & \beta_p & \alpha_p & \alpha_p & \alpha_p \\
\beta_p & \beta_p & \beta_p & \ldots & \beta_p & \alpha_p & \alpha_p & \alpha_p \\
\alpha_p & \alpha_p & \alpha_p & \ldots & \alpha_p & \alpha_p & \alpha_p & \alpha_p
\end{pmatrix},
\]

where \( \{(\alpha_i, \beta_i)|1 \leq i \leq p\} \) is determined by Formulas (6), (7) and (8). Let \( \eta_1, \eta_2, ..., \eta_p \) denote the eigenvalues of \( T \), then we have the following two theorems.

**Theorem 6.** \( \sum_{i=1}^p \eta_i = \sum_{i=1}^n \lambda_i \). □

**Theorem 7.** The eigenvalues of \( T \) are the same with that of \( C^{(w)} \) when \( p = n \). □

The proofs of the above two theorem are given in [23]. When \( p = n \), we have \( \sum_{i=1}^n \eta_i = \sum_{i=1}^n \lambda_i \) as \( T \) and \( C^{(w)} \) have the same eigenvalues when \( p = n \). Furthermore, as Lanczos first noted, \( T \)'s eigenvalues are also excellent approximations of the eigenvalues of \( C^{(w)} \) even when \( p < n \) [27]. i.e. \( \sum_{i=1}^p \lambda_i \) is close to \( \sum_{i=1}^p \eta_i \). Based on Theorem 8, \( \sum_{i=1}^p \eta_i \) is close to \( \sum_{i=1}^p \alpha_i \) especially when \( \epsilon \) is small. Based on the analysis at the beginning of this section, the size of the \( \epsilon \)-kernel component returned by the distributed algorithm is close to the minimum one, which is also verified by the experiment results in Section VI.

VI. EXPERIMENT RESULTS

To evaluate the performance of the Correlation Coefficient Matrix Maintenance Algorithm, we firstly use TelosB motes to continuously sample indoor temperature, humidity and light intensity. The transmission radius of each sensor is set to be 20m. The data sampling system is built on TinyOS 2.1.0, and the light intensity datasets are used for evaluation. Second, we use Tossim to simulate a network with 1024 sensor nodes. The network is deployed in a 160m x 160m rectangular region, and the transmission radius of each sensor is also set to be 20m. The sensory data is generated according to the model given by [28], whose accuracy has been verified by the real dataset from the Intel Berkeley Research Lab.

To evaluate the performance of the centralized and distributed Dominant Dataset Drawing Algorithms in large scale networks, we use two simulators, Tossim and NS2, to construct simulated networks with different sizes. For a network whose size is smaller than 1024, Tossim simulator is used. Otherwise, NS2 simulator is adopted. The transmission radius of each sensor node in all simulated networks is 20m, and the sensory data is also generated by the model given by [28].

According to [29], the energy cost of a sensor to send and receive 1 byte message is set to be 0.0144mj and 0.0057mj. For convenience, we use CCM to denote the Correlation Coefficient Matrix Maintenance Algorithm, and use Centralized-DD and Distributed-DD to represent the Centralized and Distributed Dominant Dataset Drawing Algorithms.
A. The performance of the CCM Algorithm

According to the analysis in Section IV, the correlation between two different sensors will decline sharply with the growth of their distance due to the spatial correlation. To verify such a fact, the following experiments are carried out.

The first group of experiments is based on the real sensor networks. In the experiments, the absolute value of the correlation of two sensors were calculated while their distance, \(d\), increased from 0m to 8m, and the time window size \(m\) was set to be 30, 50 and 100 respectively, where the correlation is defined in Section III. The results in Fig.2(a) shows the following two facts. First, the absolute value of the correlation between two sensors declines sharply with the increment of \(d\). Considering that the correlation between two sensors is nearly 0 when \(d\) is increased to the transmission radius, the sensors correlated with sensor \(i\) (\(1 \leq i \leq n\)) can be reached by one-hop transmission in most cases. Second, the correlation calculated when \(m = 50\) is almost the same as that in the cases when \(m = 100\), therefore, the correlation will become stable when the time window size reaches a certain value.

The second group of experiments is to investigate the impact of \(d\) on the correlation of two sensors in the simulated networks. In the experiments, the absolute value of the correlation of two sensors was calculated while their distance \(d\) increased from 0m to 14m, and the time window size \(m\) was set to be 10, 20, and 50 respectively. The results are given in Fig.2(b). The figure presents similar patterns as Fig.2(a).

The third group of experiments is to investigate the relationship between the energy cost of CCM and \(d\). Since CCM is divided into two phases, the energy costs of these two phases is investigated separately. In the experiments, the energy cost by the two phases were calculated while \(d\) was increased from 0m to 20m, and \(m\) was set to be 20 and 50. Fig.3.(a) indicates that the energy consumed by the two phases increase with the growth of \(d\) since the sensory data of each sensor needs to be broadcasted in range of \(d\). It also shows that the energy cost by the maintenance phase is extremely small comparing with the initial phase. Since the initial phase only happens once, and thus the total energy cost of CCM is very small.

The fourth group of experiments is to investigate the impact of \(m\) on the energy consumed by CCM. In the experiments, the energy costs of two phases were calculated while \(m\) was increased from 10 to 50, and \(d\) was set to be 10m and 20m. The results in Fig.3.(b) show that the energy cost of the initial phase increases with the growth of \(m\) as more sensory data needs to be broadcasted. However, the energy cost of the maintenance phase is stable even when \(m\) becomes large, which is because the history information is fully used while the time window is sliding, so that the energy cost is independent with \(m\). Moreover, the total energy cost of CCM is very small since the maintenance phase happens many times and the energy cost of CCM mainly depends on it.

B. The performance of Centralized-DD and Distributed-DD

As none of the published algorithms can deal with such a problem in WSNs, we only evaluate the performance of Centralized-DD and Distributed-DD.

1) The Comparison on the Ratio of the dominant dataset: The ratio of the dominant dataset equals to the size of it divided by the size of the whole data generated by a WSN in a time window. It is an important parameter to evaluate the compression ability of a dominant data drawing algorithm.

The first group of experiments is to investigate the relationship between \(\epsilon\) and the ratios of the dominant dataset drawn by Centralized-DD and Distributed-DD, respectively. In the experiments, the ratios of such dominant datasets were calculated while \(\epsilon\) increased from 0.05 to 0.5, and \(n \in \{400, 1024, 2025\}\). The results in Fig.4 show that the ratio of the dominant dataset is very small even when \(\epsilon\) is small for both Centralized-DD and Distributed-DD. For example, the ratios of the dominant datasets returned by both of two algorithms are less than 6.55% when \(\epsilon = 0.05\) and \(n = 1024\), i.e. the dominant dataset only use 6.55% data values to guarantee that the 95% information from the raw sensory data is preserved. It also
indicate that the ratio of the dominant dataset decreases when $n$ is large. Since the density of the networks becomes larger and more redundant data are generated with growth of $n$, thus, more useless data are filtered out by the two algorithms.

The second group of experiments is to compare the ratios of the dominant datasets drawn by Centralized-DD and Distributed-DD. In the experiments, we firstly calculated the ratios of the dominant datasets returned by Centralized-DD and Distributed-DD while $\epsilon$ increased from 0.1 to 0.4 and $n = 1024$. Then, these ratios were computed while $n$ increased from 400 to 2025 and $\epsilon = 0.2$. Fig.5(a) shows that the ratios of the dominant data drawn by two algorithms are almost the same, which verifies the near optimal property of Distributed-DD since the size of dominant data drawn by Centralized-DD is minimized. Fig.5(b) shows that the ratio of the dominant data drawn by Distributed-DD is very close to the minimized one when $n$ is large. Thus, Distributed-DD is more suitable to draw the dominant data when the size of a network is large and the information preservation requirement is strict.

2) The Comparison on Energy Cost and Complexities:
The first group of experiments is to investigate the energy costs of Centralized-DD and Distributed-DD with varying $\epsilon$. In the experiments, the network size was set to be 1024, and the energy costs of Centralized-DD and Distributed-DD were calculated while $\epsilon$ was increased from 0.1 to 0.4. The results in Fig.6(a) indicate that the energy cost of Distributed-DD is extremely small. For example, the energy cost of Centralized-DD is $2.78\ J$ when $\epsilon = 0.1$, however, the energy consumed by Distributed-DD is about $0.13\ J$ to satisfy the same $\epsilon$. Meanwhile, Centralized-DD still consumes lots of energy even when $\epsilon$ is large, which is because Centralized-DD requires all the sensors to transmit the correlations to the sink. Thus, the energy cost of Centralized-DD is still quite large even when the information preservation requirement is relaxed.

The second group of experiments is to investigate the energy costs of Centralized-DD and Distributed-DD with the size of a network increasing. In the experiments, $\epsilon$ was set to be 0.2, and the energy costs of the two algorithms were calculated while $n$ grew from 400 to 2025. The results in Fig.6(b) show that the energy cost of Centralized-DD is more than $10\ J$ when $n = 2025$, while the energy consumed by Distributed-DD is smaller than $0.2\ J$ in the same network. Thus, the energy cost of Distributed-DD is 50 times smaller than that of Centralized-DD, which indicates Distributed-DD is very efficient for large-scale networks. Moreover, the energy cost of Centralized-DD increases sharply with the growth of network size as more energy is consumed for transmitting correlations to the sink.

The third group of experiments is to investigate the computation complexities of the two algorithms. In the experiments, we firstly computed the computation complexities of the two algorithms while $n = 1024$ and $\epsilon$ increased from 0.1 to 0.4. Then, the computation complexities of the two algorithms were calculated while $\epsilon = 0.2$ and $n$ grew from 400 to 2025. Fig.7(a) and Fig.7(b) show that the computation complexity of Distributed-DD is extremely small compared with that of Centralized-DD. For example, when $n = 1024$ and $\epsilon = 0.2$, the complexity of Centralized-DD is $O(10^9)$ and the complexity of Distributed-DD is $O(1)$. The reasons are as follows. First, Centralized-DD needs to calculate the eigenvalues and eigenvectors of the correlation coefficient matrix, so that the computation complexity of Centralized-DD is at least $O(n^3)$. However, Distributed-DD does not need to carry out such operations, and it has a constant complexity. Furthermore, Fig.7(b) shows that the computation complexity of Centralized-DD increases sharply with the growth of $n$, and the reason is the same as the aforementioned one.

VII. Related Works

Since we are first one to consider the problem of drawing the dominant dataset from WSNs, the published literatures introducing sensory data reduction techniques are only one that related to our work. Current sensory data reduction techniques can be divided into two categories.

One of such techniques are based on sampling. The work in [30] proposed a sampling based approach to process the top-$k$ queries. Using a group of linear programming functions according to history sensory data, the authors try to assigned larger sampling probability to the sensor used to have the large sensed values in history. The method is not suitable to detect abnormal events since the method depends on history information. The works in [11] and [12] provided several sampling-based aggregation algorithms to deal with aggregation queries in static and dynamic networks. The author proved that these methods can satisfy arbitrary precision requirement. However, the queries processed by the above algorithms are simple, and they cannot deal with complicated queries. The works in [31] and [32] proposed the sampling-based quantile computation algorithms in sensor networks. All above sampling based techniques have a similarly problem. They did not preserve the core information of the original sensory data and did not support to recover data or other complexity queries.

The other ones are compression techniques. The sketch based compression methods [14] were proposed firstly. It provided an $\log(n)$ bits sketch to represent $n$ bits original sensory data. However, it also only can deal with a few simple queries. [33] and [16] proposed a Huffman and source coding based compression algorithm, respectively. It required the sink to generate a group of coding and broadcasting to the network, and each sensor compressed their sensory data according to the received coding. These algorithm required the sink to know the distribution of the sensory data in details, which is almost impossible when the scale of a WSN is large. The wavelet based compression techniques were proposed by [34]. The methods can efficiently compress sensory data on the time dimension, however, it leads to high communication cost when using these methods to compress sensory data [15] and [17] proposed two compression methods based on linear regression and information entropy, respectively. All of these compression methods cannot directly support computation and other analysis operations, and a decompression process is required before dealing with the computation tasks, which will lead to additional energy and time consumptions.
VIII. CONCLUSION

This paper studies the problem of drawing the minimum $\epsilon$-dominant dataset from a WSN. We prove that it is P-problem, and provide an accurate centralized algorithm with $O(n^3)$ complexity to solve it. A distributed algorithm with constant complexity is also proposed to save the energy and computation resource. We prove that the result returned by the distributed algorithm has a near optimal size. A correlation coefficient matrix maintenance algorithm is also given. The experiment results verify that all proposed algorithms have high performance in terms of accuracy and energy efficiency.

ACKNOWLEDGMENT

This work was supported in part by the National Basic Research Program of China (973 Program) under Grant No. 2012CB316200, the National Natural Science Foundation of China (NSFC) under Grant No. 61190115, 61370217, the Fundamental Research Funds for the Central Universities under grant No. HIT.KISTP201415, the National Science Foundation (NSF) under Grants No. CNS-1152001, CNS-1252292, the Research Fund for the Doctoral Program of Higher Education of China under grant No.20132302120045, and the Natural Science Research Innovation Foundation in Harbin Institute of Technology under grant No.HIT.NSRIF:2014070.

REFERENCES


Fig. 6: The Comparison on Energy Cost

(a) Affected by $\epsilon$
(b) Affected by $n$

Fig. 7: The Comparison on Computation Complexity

(a) Affected by $\epsilon$
(b) Affected by $n$