Research Article

A Three-Phase Top-|k| Query Based Distributed Data Collection Scheme in Wireless Sensor Networks

Guorui Li, Jingsha He, and Cong Wang

1School of Computer and Communication Engineering, Northeastern University at Qinhuangdao, Qinhuangdao, Hebei 066004, China
2School of Software Engineering, Beijing University of Technology, Beijing 100124, China

Correspondence should be addressed to Guorui Li; lgr@mail.neuq.edu.cn

Received 8 October 2014; Revised 23 November 2014; Accepted 12 December 2014

Academic Editor: Qilian Liang

Copyright © 2015 Guorui Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We propose a three-phase top-|k| query based distributed data collection scheme which is designed for clustered or multisink wireless sensor networks. The proposed scheme consists of a distributed iterative hard thresholding algorithm and a three-phase top-|k| query algorithm. In the distributed iterative hard thresholding algorithm, the cluster heads or sink nodes reconstruct the compressed data in a distributed and cooperative manner. Meanwhile, the top-|k| query operation in the above algorithm is realized by pruning unnecessary elements among cluster heads or sink nodes in the three-phase top-|k| query algorithm. Simulation results show that there is no obvious difference in the performance of data reconstruction between our proposed scheme and existing compressive sensing theory based data collection schemes. However, both the number of interactions and the amount of transmitted data among cluster heads or sink nodes can be effectively reduced in the proposed scheme. The performance of the proposed scheme is analyzed in detail in this paper to support the claims.

1. Introduction

A wireless sensor network (WSN) is a cooperative network consisting of small, battery operated, and energy constrained wireless sensor nodes. Main applications of WSNs include the monitoring of the areas covered by sensor nodes through the collection of local surveillance data and the transmission of the collected data to one or more sink nodes in multihop communication for processing and analysis [1]. WSNs can be deployed to support a series of new and exciting applications such as wildlife monitoring, disaster response, military surveillance, smart building, and industrial quality control [2].

However, limited resources in wireless sensor nodes in terms of computation, storage, communication bandwidth, and energy make data collection a very challenging issue. Two representative types of data collection schemes that have been proposed for WSNs are spatial-temporal correlation based data prediction schemes [3–5] and distributed source coding schemes [6, 7]. In the first types of schemes, the sensed data are derived using a data prediction model based on preconstructed spatial and/or temporal correlation in the server without being transmitted to the sink node through one or more hops. In the second types of schemes, a coding algorithm based on distributed Slepian-Wolf theory can be designed to reduce the number of transmitted messages among sensor nodes. However, both types of schemes still suffer from the shortcomings of high computational cost in the sensor nodes as well as high communication cost. For example, revised model parameters and coded sensed data need to be transmitted within the network from time to time.

Recently, some new ideas based on the compressive sampling theory have been introduced into solving the data collection problem in WSNs. In the classical Shannon sampling theory, a limited bandwidth of the signal is utilized, resulting in a high degree of redundancy in the sampled data. The compressive sampling theory aims to lift the limitation of the Shannon sampling theory by suggesting that a few random linear projections of a sparse or compressible signal would contain enough information to reconstruct the original signal [8]. The characteristics of the compressive sampling theory, such as compressibility, asymmetry, versatility, and robustness, make it feasible to design sampling algorithms for WSNs [9]. The compressive sampling theory has started to
be applied to deal with the problems of data collection [10], localization [11], channel estimation [12], video streaming [13], and coding [14] in WSNs only recently.

In the area of research in data collection based on the compressive sampling theory, Luo et al. proposed the compressive data gathering (CDG) scheme [15] in which through adopting a chain-type topology and a distributed random coefficient projection method the sensed data are compressed among wireless sensor nodes and reconstructed in the sink node. Wang et al. proposed a data collection scheme based on the adaptive compressive sampling theory [16] by taking advantage of characteristics of the spatial-temporal inconsistency of the sparsity in the sensed data and by introducing an autoregressive model into the data reconstruction process. Xu et al. proposed a group of compressive sparse functions by utilizing the symmetric and orthogonal properties of the discrete cosine transform (DCT) functions so that sensed data can be reconstructed after the coefficients of the compressive sparse functions are recovered from the partial sensed data [17]. Motivated by the chaos technology and compressive sampling theory, Lu et al. proposed a distributed secure and efficient data collection scheme by designing a chaotic sequence based sensing matrix generation algorithm and active node matrix algorithm [18]. Wu et al. proposed a sparsest random scheduling scheme for compressive data gathering in wireless sensor networks based on a specially designed sparsest measurement matrix [19]. Although the above data collection schemes can perform energy efficient data collection in WSNs, the compressed projections in each sensor node must be transmitted to the sink node in a multihop fashion and the sensed data can be reconstructed only in the server in a centralized manner. In a clustered or multisink WSN, the cluster heads or sink nodes are not able to obtain the sensed data unless the server retransmits the reconstructed results to them. However, such a compression-reconstruction-retransmission process is time consuming and less energy efficient.

In this paper, we propose a three-phase top-$|k|$ query based distributed data collection scheme for clustered or multisink WSNs. A key characteristic of our scheme is that, by collecting the local set of sensed data and exchanging the compressed projections of these data, the cluster heads or sink nodes are able to obtain the sensed data of the whole network without the need of retransmission from the server. Our proposed data collection scheme consists of a distributed iterative hard thresholding algorithm and a three-phase top-$|k|$ query algorithm. With the distributed iterative hard thresholding algorithm, the cluster heads or sink nodes can reconstruct the compressed data in a distributed and cooperative manner. With the three-phase top-$|k|$ query algorithm, the top-$|k|$ query operation in the above algorithm is realized in a distributed manner among cluster heads or sink nodes through pruning unnecessary elements in the data set. We show through experiment results that the proposed data collection scheme requires a few numbers of interactions and a smaller amount of data transmission among cluster heads or sink nodes than existing compressive sensing theory based data collection schemes while achieving the same performance for data reconstruction.

The main contributions of this paper are summarized as follows.

1. We propose a distributed compressive sensing theory based data collection scheme in WSNs by utilizing the spatial sparsity of the sensed data. By decomposing the tasks of the iterative hard thresholding algorithm into several correlated parts, the cluster heads or sink nodes can reconstruct the original sensed data in a distributed, energy efficient manner.

2. We propose a three-phase top-$|k|$ query algorithm to implement the hard thresholding operation $H_k()$ in the iterative hard thresholding algorithm in a distributed manner. Differing from the traditional top-$k$ query algorithm which is only suitable to monotonic aggregation functions, such as the sum of nonnegative numbers, the proposed three-phase top-$|k|$ query algorithm can be applied to nonmonotonic aggregation functions. In particular, we use the sum of all numbers as the nonmonotonic aggregation function in this paper.

3. We compare the performance of the proposed data collection scheme with some current compressive sampling theory based data collection schemes. We also perform a thorough analysis on the overall performance of the proposed scheme, including data reconstruction performance, the amount of transmitted data, and that of computed data.

The remainder of this paper is organized as follows. In Section 2, we briefly introduce the compressive sampling theory and the iterative hard thresholding algorithm. In Section 3, we introduce the system model. In Section 4, we describe the three-phase top-$|k|$ query based distributed data collection scheme. In Section 5, we evaluate the performance of the proposed scheme. Finally, in Section 6, we conclude this paper in which we also discuss our future research directions.

2. Compressive Sampling

2.1. The Basics. In the compressive sampling theory, the signal measurement process can be described as

$$y = \Phi x,$$  \hspace{1cm} (1)

where $x \in \mathbb{R}^n$ is the original signal, $y \in \mathbb{R}^m$ is the measurement signal, and $\Phi \in \mathbb{R}^{m \times n}$ ($m < n$) is the measurement matrix. The underdetermined system (1) has an infinite number of solutions for the reason of $m < n$. In other words, the measurement signal $y$ cannot be accurately reconstructed through the original signal $x$.

However, the natural signals are usually sparse in a set of appropriate transform bases although they are not sparse in the time domain. Therefore, the original signal $x$ can be expressed as

$$x = \Psi s,$$  \hspace{1cm} (2)
where $\Psi \in \mathbb{R}^{n \times n}$ is an orthonormal basis matrix and $s \in \mathbb{R}^n$ is a $k$ sparse vector. In other words, $\|s\|_0 = k$ and $\|s\|_0$ indicates the number of non-zeros in $s$.

By plugging (2) into (1), the signal measurement process can be described as

$$y = \Phi x = \Phi \Psi s = As,$$

where matrix $A = \Phi \Psi \in \mathbb{R}^{m \times n}$ is usually referred to as a sensing matrix. Although the underdetermined system (1) is ill-posed, its sparsest solution can be obtained by solving the following $l_0$ optimization problem:

$$\min \|x\|_0 \quad \text{s.t.} \quad Ax = y.$$  

(4)

However, the $l_0$ optimization problem (4) is a NP-Hard problem\cite{20}. In practice, it is usually relaxed as the following $l_1$ optimization problem:

$$\min \|x\|_1 \quad \text{s.t.} \quad Ax = y.$$  

(5)

A key result in the compressive sampling theory states that if the sensing matrix $A$ satisfies the restricted isometry property (RIP) of order $2k$ and the restricted isometry constant (RIC) $\delta_{2k} \leq \sqrt{2} - 1$, then, for all $k$ sparse signals, the solutions of (4) and (5) are equal as long as the number of measurements satisfies

$$m \geq c \mu^2 (\Phi, \Psi) k \log \left( \frac{n}{k} \right),$$

(6)

where $\mu(\Phi, \Psi)$ stands for the mutual coherence between matrices $\Phi$ and $\Psi$ \cite{21}. The commonly used algorithms for solving the compressive sampling problem include the optimization algorithms, the greedy algorithms, and the iterative thresholding algorithms. In the optimization algorithms, a series of convex optimization methods are adopted to search the global optimal solution of the optimization problem (5). The representative optimization algorithms include the basis pursuit algorithm \cite{22}, the gradient projection algorithm \cite{23}, and the interior point algorithm \cite{24}. In the greedy algorithms, the support set of the signals is selected based on some kind of greedy selection criteria. This category includes the matching pursuit algorithm \cite{25}, the orthogonal matching pursuit algorithm \cite{26}, the regularized orthogonal matching pursuit algorithm \cite{27}, the compressive sampling matching pursuit algorithm \cite{28}, and the subspace matching pursuit algorithm \cite{29}. In the iterative thresholding algorithms, the optimal sparse solution is obtained through hard or soft thresholding operations in a number of iterations. The typical iterative thresholding algorithms include the iterative hard thresholding algorithm \cite{30}, the hard thresholding pursuit algorithm \cite{31}, and the soft thresholding algorithm \cite{32}.

2.2. The Iterative Hard Thresholding Algorithm. The iterative hard thresholding algorithm belongs to the set of iterative algorithms. Its core idea is to approximate the inverse of the sensing matrix $A$ using the transverse of that matrix; that is, let $A^T \approx A^{-1}$.

By multiplying $A^T$ on the two sides of the constraint condition of the $l_1$ optimization problem (5), we can get

$$A^T A x = A^T y.$$  

(7)

Then we can obtain

$$x = (I - A^T A) x + A^T y,$$  

(8)

by transposition and thus the following iterative equation:

$$x^{(t+1)} = x^{(t)} + A^T (y - A x^{(t)}).$$  

(9)

In each round of the above iteration, we choose the largest (in magnitude) $k$ elements of the current iterative vector $x$ and set other elements to zero because the original signal $x$ is $k$ sparse in the orthonormal basis $\Psi$. The aforementioned selection operation is executed using the following hard thresholding operator:

$$H_K(x)_i = \begin{cases} x_i & x_i \in \{x_{\text{Top-}|k|}\} \\ 0 & x_i \notin \{x_{\text{Top-}|k|}\} \end{cases}.$$  

(10)

where $x_{\text{Top-}|k|}$ represents the set of the largest $k$ elements of $x$ in magnitude. Therefore, the iterative equation (9) becomes

$$x^{(t+1)} = H_K(x^{(t)}) + A^T (y - A x^{(t)}).$$  

(11)

After a number of iterations, the results of the iterative hard thresholding algorithm will converge to the optimized solution of $l_1$ optimization problem (5).

3. The System Model

Considering a static WSN in a two-dimensional space as shown in Figure 1 where a number of wireless sensor nodes
are distributed uniformly and randomly in the surveillance area, the agent node in the figure can be either a cluster head or a sink node.

We assume that there are $p$ agent nodes and $m$ sensor nodes that have been deployed in the WSN and all agent nodes know their own locations. The agent node which is located closest to the center of the network is appointed as the administrator agent and other agent nodes are appointed as the member agent nodes.

The sensed data in a WSN at a certain point of time constitute a $k$ sparse vector $x \in \mathbb{R}^n$ because there is spatial correlation among the surveillance data in the monitored area. Each agent node $i \in \{1, \ldots, p\}$ takes $m_i$ linear measurements $y_i \in \mathbb{R}^{m_i \times 1}$ of $x$ using its subsensing matrix $A_i \in \mathbb{R}^{m_i \times n}$. Thus, the set of all agent nodes will equivalently obtain $m = m_1 + m_2 + \cdots + m_p$ linear measurements $y \in \mathbb{R}^{m \times 1}$ using global sensing matrix $A \in \mathbb{R}^{m \times n}$, where

$$A = \begin{bmatrix} A_1 \\ \vdots \\ A_p \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_p \end{bmatrix}. \quad (12)$$

Therefore, the problem of sensed data collection in the set of all agent nodes can be ascribed to the $l_1$ optimization problem (5). By reconstructing the original sensed data vector in a distributed manner, all agent nodes can obtain the sensed data of whole sensor network completely.

4. The Distributed Data Collection Scheme

The three-phase top-$|k|$ query based distributed data collection scheme consists of two correlated algorithms: the distributed iterative hard thresholding algorithm and the three-phase top-$|k|$ query algorithm. The first algorithm implements the iterative hard thresholding algorithm in a distributed and cooperative manner while the second algorithm provides the distributed version of the top-$|k|$ operator $H_{|k|}$ in (10) for the first algorithm.

4.1. The Distributed Iterative Hard Thresholding Algorithm

Firstly, the initial value of the original sensed data vector $x^{(0)}$ and the iteration time $t$ at each agent node are set to zero vector $0$ and $0$, respectively. Then, each agent node $i \in \{1, \ldots, p\}$ would calculate the intermediate result $z_i^{(0)}$ using its own subsensing matrix $A_i$ and measurement projection $y_i$ by executing

$$z_i^{(0)} = \begin{cases} x^{(0)} + A_i^T (y_i - A_i x^{(0)}) & i = 1 \\ A_i^T (y_i - A_i x^{(0)}) & i \neq 1 \end{cases} \quad (13)$$

in each iteration $t$. In particular, the administrator agent node and every member agent node would calculate $x^{(0)} + A_1^T (y_1 - A_1 x^{(0)})$ and $A_i^T (y_i - A_i x^{(0)})$, respectively. Then, all the agent nodes would execute the three-phase top-$|k|$ query algorithm to be described in the next subsection and the administrator agent node would get

$$x^{(t+1)} = H_K \left( \sum_{i=1}^{p} z_i^{(t)} \right). \quad (14)$$

Finally, the administrator agent node would send the recovered vector $x^{(t+1)}$ to each member agent node and all the agent nodes will continue to the next round of iteration.

We will show through Theorem 1 that the iterative results (14) in the distributed iterative hard thresholding algorithm are equal to the iterative results (11) in the centralized iterative hard thresholding algorithm. Therefore, after a number of iterations, the intermediate result $x^{(t+1)}$ will converge to the optimized solution of the $l_1$ optimization solution (5).

**Theorem 1.** The intermediate result $x^{(t+1)}$ in the centralized iterative hard thresholding algorithm is the same as that in the distributed iterative hard thresholding algorithm.

**Proof.** According to (13) and (14), the intermediate result $x^{(t+1)}$ in the distributed iterative hard thresholding algorithm is

$$x^{(t+1)} = H_K \left( \sum_{i=1}^{p} z_i^{(t)} \right)$$

$$= H_K \left( x^{(t)} + \sum_{i=1}^{p} A_i^T (y_i - A_i x^{(t)}) \right) \quad (15)$$

$$= H_K \left( x^{(t)} + \left[ A_1^T \ldots A_p^T \right] \begin{bmatrix} y_1 - A_1 x^{(t)} \\ \vdots \\ y_p - A_p x^{(t)} \end{bmatrix} \right)$$

$$= H_K \left( x^{(t)} + A^T (y - A x^{(t)}) \right).$$

Therefore, the iterative results in the distributed iterative hard thresholding algorithm are equal to that in the centralized iterative hard thresholding algorithm.

The pseudocode of the distributed iterative hard thresholding algorithm is presented in Algorithm I.

4.2. Three-Phase Top-$|k|$ Query Algorithm

The three-phase top-$|k|$ query algorithm is proposed to realize the hard thresholding operator $H_{|k|}$ in a distributed manner. Unlike the classical top-$k$ query algorithm which is only suitable to monotonic aggregation functions, our proposed algorithm can perform the largest $k$ absolute elements selection operation which is a nonmonotonic aggregation function in a distributed way. The basic idea of this algorithm is to prune unnecessary elements in the distributed system by only exchanging partial elements among administrator agent node and member agent nodes in three phases.
The three-phase top-\(|k|\) query algorithm includes the following three phases.

(1) The initial data set selection phase: in this phase, the initial dataset \(L\) is selected and distributed among the agent nodes.

(2) The candidate data set selection phase: in this phase, the candidate set \(S\) is built by computing the upper bounds and lower bounds of sums for the top-\(|k|\) elements.

(3) The top-\(|k|\) elements selection phase: in this phase, the precise sums for the top-\(|k|\) element are finally determined based on the candidate data set \(S\).

In each round of iteration in the distributed iterative hard thresholding algorithm, every agent node \(i\) can obtain its own intermediate result \(z_i^{(t)}\) through (13). Before executing the three-phase top-\(|k|\) query algorithm, every agent node \(i\) should sort \(z_i^{(t)}\) by value in a descending order and would get \(L_i = \{j, v_i(j), j = 1, \ldots, n_i\}\), where \(j\) is the element index, \(v_i(j)\) is the corresponding element value, and \(n_i\) is the number of elements in agent node \(i\).

In the three-phase top-\(|k|\) query algorithm, the whole sum \(V(j)\), the partial sum \(P(j)\), the upper bound of the whole sum \(U(j)\), and the lower bound of the whole sum \(L(j)\) for each element are computed according to the following formulas:

\[
V(j) = \sum_{i=1}^{p} v_i(j),
\]

\[
P(j) = \sum_{i=1}^{p} v_i'(j),
\]

\[
v_i'(j) = \begin{cases} v_i(j) & \text{if } v_i(j) \text{ is known} \\ 0 & \text{if } v_i(j) \text{ is unknown} \end{cases}
\]

\[
U(j) = \sum_{i=1}^{p} \overline{v}_i(j),
\]

\[
\overline{v}_i(j) = \begin{cases} v_i(j) & \text{if } v_i(j) \text{ is known} \\ \overline{T}_i & \text{if } v_i(j) \text{ is unknown} \end{cases}
\]

\[
L(j) = \sum_{i=1}^{p} \underline{v}_i(j),
\]

where \(\overline{T}_i\) and \(\overline{T}_i\) are the smallest positive local value and the largest negative local value in \(L\) among all elements in the initial data set \(L\), respectively. If an element in \(L\) does not appear in \(L\), just set it to 0.

The partial sum \(P(j)\) is used to represent the sum of top-\(|k|\) elements roughly. The upper bound of the whole sum \(U(j)\) and the lower bound of the whole sum \(L(j)\) are used to estimate the range of the sum of the top-\(|k|\) elements. By comparing the partial sum with the upper bound and the lower bound of the whole sum, we can filter out unnecessary elements and construct the candidate set \(S\). Thus, the amount of transmitted data among agent nodes can be effectively reduced. Finally, the whole sum \(V(j)\) is used to compute the sums of each top-\(|k|\) element in the aforementioned candidate set \(S\).

The three-phase top-\(|k|\) query algorithm is presented in Algorithm 2 where it is worth noting that the administrator agent node and the member agent nodes all follow the same element selection rules.

The interactive process between administrator agent node and member agent nodes in one iteration of the proposed scheme is shown in Figure 2. Although the three-phase top-\(|k|\) query algorithm includes three phases, the computation complexity of the algorithm is trivial. By analyzing the algorithm thoroughly, we can see that sorting and summation are the two nontrivial parts in the proposed algorithm. If the number of involved elements in each agent is \(n\), then the computational complexities of sorting and summation are \(O(n\lg n)\) and \(O(n)\), respectively. Therefore, the computational complexity of the three-phase top-\(|k|\) query algorithm is \(O(n\lg n)\). The communicational complexity of the three-phase top-\(|k|\) query algorithm will be analyzed through simulations in Section 5.

4.3. An Example of the Three-Phase Top-\(|k|\) Query Algorithm.

We now describe the three-phase top-\(|k|\) query algorithm.
Input: the descending list $L_i = \{(j, v_j(j)), j = 1, \ldots, n_i\}$.
Output: the sums of each top-\(|k|\) elements.

The initial data set selection phase
1. If node is a member agent node Then
2. Sends first $k$ positive and last $k$ negative elements in $L_i$ to administrator agent node;
3. Else
4. Computes partial sums for received elements according to (17);
5. $L \leftarrow \{\text{the largest } k \text{ positive sums and the smallest } k \text{ negative sums}\}$;
6. Sends $L$ to all member agent nodes;

The candidate data set selection phase
1. If node is a member agent node Then
2. Sends unsent elements in $L_i$ whose values $\geq T_i$ or $\leq \sum_j$ combined with $T_i$ and $\sum_j$ to the administrator agent node;
3. Else
4. Computes the partial sum for received elements according to (17);
5. $\mathcal{P} \leftarrow \text{the } k\text{th highest positive element}$;
6. $\mathcal{T} \leftarrow \text{the } k\text{th lowest negative element}$;
7. Computes upper bounds of whole sum for received elements according to (18);
8. Computes lower bounds of whole sum for received elements according to (19);
9. $S_1 \leftarrow \{j \mid U_j(j) \geq \mathcal{P}\text{ or } L_j(j) \leq \mathcal{T}\}$;
10. Sends $S_1$ to all member agent nodes;

The top-\(|k|\) elements selection phase
1. If node is a member agent node Then
2. Sends unsent elements in $S_1$ to administrator agent node;
3. Else
4. Computes whole sums for received elements according to (16);
5. $S_{\text{top-}k} \leftarrow \{\text{the largest } k \text{ elements in magnitude}\}$;

Algorithm 2: The three-phase top-\(|k|\) query algorithm.

| Administrator agent node | Member agent node 1 | Member agent node 2 |
|---------------------------|---------------------|---------------------|
| (1, 10)                   | (3, 10)             | (6, 12)             |
| (4, 9)                    | (1, 8)              | (1, 8)              |
| (6, 3)                    | (6, 2)              | (7, 5)              |
| (5, 1)                    | (5, 1)              | (7, 2)              |
| (3, −1)                   | (7, −1)             | (4, 1)              |
| (9, −2)                   | (8, −3)             | (9, −1)             |
| (8, −6)                   | (4, −5)             | (2, −6)             |
| (2, −9)                   | (2, −10)            | (3, −7)             |

Through an example in which the elements in the administrator agent node and in the member agent nodes are shown in Table 1. These elements are sorted by value in a descending order. In this case, the number of agent nodes $p = 3$ and the number of selected elements $k = 2$. In other words, we query top-\(|2|\) elements in a distributed manner.

Table 2 lists the computed results.

An explanation of the example follows. In the initial data set selection phase, member agent nodes 1 and 2 send $\{(3, 10), (1, 8)\}$ and $\{(6, 12), (1, 8)\}$ to the administrator agent node, respectively. After receiving the elements, the administrator agent node begins to compute the partial sums according to (17). For element 1, $P(1) = 26$ because $v_1(1) = 10, v_2(1) = 8,$ and $v_3(1) = 8$ are all known. For element 3, $P(3) = 3$ because $v_2(3) = 10$ and $v_3(3) = −7$ are known, but $v_1(3)$ is located in neither the first 2 elements nor the last 2 elements in the administrator agent node. Following the similar computing process, we can get $P(6) = 12, P(4) = 4, P(8) = −6,$ and $P(2) = −25$. Therefore, the initial data set $L = \{(1, 6, 8, 2)\}$ is selected and transmitted to all the member agent nodes.

In the candidate data set selection phase, the smallest positive local value and the largest negative local value in $L_i$ among all elements in $L$ are $\overline{T}_1 = 3, \overline{T}_2 = 2, \overline{T}_3 = 8$ and $\underline{T}_1 = −6, \underline{T}_2 = −3, \underline{T}_3 = −1$, respectively. Therefore, member agent nodes 1 and 2 transmit unsent elements $\{(6, 2), (8, −3)\}$ and $\{(9, −1)\}$ to the administrator agent node. After receiving the elements, the administrator agent node begins to compute the partial sums as well as the upper and the lower bounds of the whole sum according to (17)–(19). The positive threshold $\mathcal{P} = 17$ and the negative threshold $\mathcal{T} = −9$ are the second highest positive element and the lowest negative element. Therefore, candidate data set $S = \{(1, 6, 8, 9, 2)\}$ is selected and transmitted to all the member agent nodes.

In the top-\(|k|\) elements selection phase, there are no unsent elements in $S$. The administrator agent node computes the whole sum according to (16) and selects $S_{\text{top-}k} = \{(1, 26), (2, −25)\}$. Finally, it sends $S_{\text{top-}k}$ to all the member agent nodes.
Administrator agent node

Member agent node

The initial data set selection phase

Compute intermediate result $z_i^{(t)}$

(1) Send first $k$ positive and last $k$ negative elements in $L_i$

Compute initial data set $L_i$

(2) Send initial data set $L_i$

The candidate data set selection phase

(3) Send unsent elements in $L_i$ whose values $\geq T_i$ or $\leq T_i$

Compute candidate data set $S_i$

(4) Send candidate set $S_i$

The top-$|k|$ elements selection phase

(5) Send unsent elements in $S_i$ whose values $\geq T_i$ or $\leq T_i$

Compute top-$|k|$ set $S_{top-|k|}$

(6) Send top-$|k|$ set $S_{top-|k|}$

Figure 2: The process of interaction.

Table 2: The computed results.

| Phase                  | The initial data set selection phase | The candidate data set selection phase | The top-$|k|$ elements selection phase |
|------------------------|-------------------------------------|---------------------------------------|--------------------------------------|
| $L_i$                  | $\{1, 6, 8, 2\}$                    | $\bar{T}_1 = 3, T_2 = 2, T_3 = 8$   |                                       |
| $T_i$                  | $T_1 = -6, T_2 = -3, T_3 = -1$      |                                       |                                       |
| $S_i$                  | $T = 17$                            |                                       |                                       |
| $S_{top-|k|}$          | $\{1, 6, 8, 9, 2\}$                |                                       | $\{(1, 26), (2, -25)\}$             |

5. Experiment and Analysis

We perform the experiment through simulation to investigate the performance of the proposed three-phase top-$|k|$ query based distributed data collection scheme under various conditions. The performance matrices that we have evaluated include data reconstruction accuracy, communication complexity, and computational complexity. The simulation is carried out using Matlab R2011b on a Dell desktop computer with dual core E7500 CPU and 3.0 G memory and involves 50–200 wireless sensor nodes randomly distributed in a $400 \times 400$ m$^2$ area. Multiple uncorrelated two-dimensional Gaussian distribution has been assumed to simulate the spatial correlated data source. The number of agent nodes is 4–6
The amount of sensor nodes

50 100 150 200

SNR

50 55 60 65 70 75 80 85 90 95 100

Figure 3: The performance on data reconstruction at different data sampling rates.

and the Gaussian random matrix is selected as the sensing matrix. Finally, the simulation repeats 500 times and the final results are the averages of the 500 results.

5.1. Performance Analysis. We now analyze the performance of the three-phase top-$|k|$ query based distributed data collection scheme. Included in our performance analysis are data reconstruction, the amount of transmitted data, and that of computed data. The amount of transmitted data and that of computed data are the basis for measuring the communication and computational complexity of the scheme in which we count the accumulated amount of transmitted and computed data in all the three phases of the proposed scheme.

Figures 3, 4, and 5 show the signal noise ratio (SNR) of the reconstructed data, the amount of transmitted data, and the amount of computed data of the proposed scheme, respectively, at different data sampling rates. The SNR is computed using the following formula:

$$SNR = 20 \cdot \log_{10} \left( \frac{|x|}{|\hat{x}|} \right)$$  \hspace{1cm} (20)

to measure the data reconstruction error, where $x \in \mathbb{R}^n$ is the original data and $\hat{x} \in \mathbb{R}^n$ is the reconstructed data. We can draw the conclusion from the figures that the performance on data reconstruction improves along with the increase on the data sampling rate. At the same time, the amount of transmitted data and that of computed data would also increase. The reason is obvious since any increase on the data sampling rate would cause more data to be transmitted and computed while leading to better performance on data reconstruction.

Figures 6, 7, and 8 show the SNR of the reconstructed data, the amount of transmitted data, and the amount of computed data of the proposed scheme, respectively, in different surveillance environments. In the experiment, the number of random data sources varies between 2 and 4. We can draw the conclusion from the figures that the performance on data reconstruction of the proposed scheme would degrade when the number of data sources increases. At the same time, the amount of transmitted data and that of computed data would still increase. The reason is that any increase in the number of data sources will cause an increase in the sparsity of the sensed data set in the network, which would degrade the performance on data reconstruction and cause more data to be transmitted and computed in the agent nodes.

Figure 4: The amount of transmitted data at different data sampling rates.

Figure 5: The amount of computed data at different data sampling rates.
Figures 6, 7, and 8 show the SNR of the reconstructed data, the amount of transmitted data, and that of computed data of the proposed scheme, respectively, with different numbers of data sources. In the experiment, the number of agent nodes varies between 4 and 6. We can see from the figures that the performance on data reconstruction has no obvious difference when the number of agent nodes changes. However, the amount of transmitted data and that of computed data would increase along with the increase in the number of agent nodes. The reason is that the number of agent nodes has little influence on the performance of data reconstruction. However, more data should be transmitted and computed when more agent nodes are involved in the scheme.

5.2. Performance Comparison. We compare the performance on data reconstruction between our proposed algorithm and two similar schemes, that is, the iterative hard thresholding (IHT) algorithm [30] and the absolute threshold algorithm (ATA) based distributed data recovery algorithm [33], in this subsection.

In the ATA based distributed data recovery algorithm, the top-$|k|$ elements are selected based on computing the absolute
The amount of sensor nodes

The number of transmitted data

Figure 10: The amount of transmitted data with different numbers of agent nodes.

The amount of sensor nodes

The number of computed data

Figure 11: The amount of computed data with different numbers of agent nodes.

thresholds for each element in the list \( L_i \) in a distributed manner. The algorithm processes the sorted lists from both the top and the bottom in one single instance. In each direction, the administrator agent node requests an (index, value) pair from each member agent node \( i \) and updates the corresponding top-\( |k| \) sums it has received so far for agent node \( i \). Meanwhile, the administrator agent node also records the largest value \( \tau_i \) for each agent node \( i \). As long as the administrator agent node has received \( k \) values which are larger than \( \sum_{i=1}^{p} \tau_i \), the algorithm terminates. However, multiple interactions among agent nodes are needed while the exact number of interactions cannot be determined in advance. In addition, interactions among agent nodes are not very efficient because only the values that correspond to the same index are transmitted in an interaction.

Figure 12 shows the performances on data reconstruction of the three algorithms in which the number of agent nodes is 5, the number of data sources is 3, and the data sampling rate is 50%. We can see from the figure that the three algorithms exhibit little difference in the performance of data reconstruction.

Figures 13 and 14 show the number of interactions and the amount of transmitted data of the three algorithms, respectively. Since IHT is not a distributed algorithm, all the

Figures 10 and 11 show the performances on data reconstruction of the three algorithms in which the number of agent nodes is 4, the number of data sources is 5, and the data sampling rate is 50%. We can see from the figure that the three algorithms exhibit little difference in the performance of data reconstruction.
data in the member agent nodes need to be transmitted to the administrator agent node for processing. Thus, the number of iterations is 1. However, since no data pruning is performed before data are transmitted to the administrator agent node, the amount of transmitted data is significantly more than the other two algorithms. Conversely, since ATA and our proposed algorithm are distributed, multiple interactions are needed. We can see clearly from Figure 13 that the number of interactions in ATA increases with the number of sensor nodes. However, the number of interactions in our proposed algorithm always stays at 3, making it a fixed number which is generally far smaller than that in ATA. Meanwhile, we can see from Figure 14 that the total amount of transmitted data of our proposed algorithm is the smallest among three algorithms, thus consuming the least amount of energy among the three algorithms.

In summary, the above experiment and performance analysis show that our proposed three-phase top-[k] query based distributed data collection scheme can effectively reduce the amount of transmitted data through limiting the number of interactions while maintaining a similar performance on data reconstruction compared to some comparable schemes.

6. Conclusion

In this paper, we proposed the three-phase top-[k] query based distributed data collection scheme designed for clustered or multisink WSNs. In the scheme, the cluster heads or sink nodes can reconstruct the sensed data of the whole network by collecting local information and exchanging intermediate results among them. The distributed hard thresholding operator is realized by pruning unnecessary elements between the administrator agent node and the member agent nodes in three phases.

The performance on data reconstruction as well as the overhead incurred by the proposed scheme was analyzed through experiment and compared to some existing compressive sampling theory based data collection schemes to demonstrate its clear advantages. The computation and communication overheads of the proposed scheme were also analyzed in detail. Our future research includes the design of the compressive sampling theory based data collection schemes through utilizing the inherited spatial and temporal correlations in the sensed data.

Conflict of Interests

The authors declare that they have no competing interests.

Acknowledgments

The work in this paper has been supported by the National Natural Science Foundation of China (Grant nos. 61402094, 61272500, and 61300195), the Research Fund for the Doctoral Program of Higher Education of China (Grant no. 2012042120009), the Fundamental Research Funds for the Central Universities of China (Program no. N120423005), the Natural Science Foundation of Hebei Province, China (Grant nos. F2012501014 and F2014501078), Beijing Natural Science Foundation (Grant no. 4142008), the General Project of Liaoning Province Department of Education Science Research (Grant no. L2013099), and the Science and Technology Support Program of Northeastern University at Qinhuangdao (Grant no. XNK201401).

References

[1] R. V. Kulkarni, A. Förster, and G. K. Venayagamoorthy, “Computational intelligence in wireless sensor networks: a survey,” IEEE Communications Surveys & Tutorials, vol. 13, no. 1, pp. 68–96, 2011.
[2] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” Computer Networks, vol. 52, no. 12, pp. 2292–2330, 2008.
[3] H. Jiang, S. Jin, and C. Wang, “Prediction or not? An energy-efficient framework for clustering-based data collection in wireless sensor networks,” IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 6, pp. 1064–1071, 2011.
[4] 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,” Computer Communications, vol. 36, no. 9, pp. 1054–1066, 2013.
[5] G. Li and Y. Wang, “Automatic ARIMA modeling-based data aggregation scheme in wireless sensor networks,” Eurasip Journal on Wireless Communications and Networking, vol. 2013, no. 1, article 85, 2013.
[6] S.-T. Cheng, J.-S. Shih, C.-L. Chou, G.-J. Horng, and C.-H. Wang, “Hierarchical distributed source coding scheme and optimal transmission scheduling for wireless sensor networks,” Wireless Personal Communications, vol. 70, no. 2, pp. 847–868, 2013.
[7] W. Wang, D. Peng, H. Wang, H. Sharif, and H.-H. Chen, “Cross-layer multitrate interaction with distributed source coding in...
wireless sensor networks,” *IEEE Transactions on Wireless Communications*, vol. 8, no. 2, pp. 787–795, 2009.

[8] D. L. Donoho, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[9] S. Qaisar, R. M. Bilal, W. Iqbal, M. Naureen, and S. Lee, “Compressive sensing: from theory to applications, a survey,” *Journal of Communications and Networks*, vol. 15, no. 5, pp. 443–456, 2013.

[10] L. Xiang, J. Luo, and C. Rosenberg, “Compressed data aggregation: energy-efficient and high-fidelity data collection,” *IEEE/ACM Transactions on Networking*, vol. 21, no. 6, pp. 1722–1735, 2013.

[11] C. Zhao, Y. Xu, and H. Huang, “Weighted centroid localization based on compressive sensing,” *Wireless Networks*, vol. 20, no. 6, pp. 1527–1540, 2014.

[12] W. U. Bajwa, J. D. Haupt, A. M. Sayeed, and R. D. Nowak, “Joint source-channel communication for distributed estimation in sensor networks,” *IEEE Transactions on Information Theory*, vol. 53, no. 10, pp. 3629–3653, 2007.

[13] S. Pudlewski, A. Prasanna, and T. Melodia, “Compressed-sensing-enabled video streaming for wireless multimedia sensor networks,” *IEEE Transactions on Mobile Computing*, vol. 11, no. 6, pp. 1060–1072, 2012.

[14] L. Xu, Q. Liang, X. Cheng, and D. Chen, “Compressive sensing in distributed radar sensor networks using pulse compression waveforms,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2013, no. 1, article 36, 2013.

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

[16] J. Wang, S. Tang, B. Yin, and X.-Y. Li, “Data gathering in wireless sensor networks through intelligent compressive sensing,” in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM ’12)*, pp. 603–611, IEEE Press, Orlando, FL, USA, March 2012.

[17] L. Xu, X. Qi, Y. Wang, and T. Moschibroda, “Efficient data gathering using compressed sparse functions,” in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM ’13)*, M. Marsan, Ed., pp. 310–314. IEEE Press, Turin, Italy, 2013.

[18] W. Lu, Y. Liu, and D. Wang, “A distributed secure data collection scheme via chaotic compressed sensing in wireless sensor networks,” *Circuits, Systems, and Signal Processing*, vol. 32, no. 3, pp. 1363–1387, 2013.

[19] X. Wu, Y. Xiong, P. Yang, S. Wan, and W. Huang, “Sparsest random scheduling for compressive data gathering in wireless sensor networks,” *IEEE Transactions on Wireless Communications*, vol. 13, no. 10, pp. 5867–5877, 2014.

[20] R. G. Baraniuk, “Compressive sensing,” *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–124, 2007.

[21] M. F. Duarte and Y. C. Eldar, “Structured compressed sensing: from theory to applications,” *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4053–4085, 2011.

[22] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.

[23] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, “Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems,” *IEEE Journal on Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 586–597, 2007.