Sensors **2014**, *14*, 8330-8349; doi:10.3390/s140508330

OPEN ACCESS

sensors

ISSN 1424-8220
www.mdpi.com/journal/sensors

Article

An Adaptive Data Collection Algorithm Based on a Bayesian Compressed Sensing Framework

Zhi Liu *, Mengmeng Zhang and Jian Cui

College of Information Engineering, North China University of Technology, Beijing 100144, China;
E-Mails: zmm@ncut.edu.cn (M.Z.); landrew@126.com (J.C.)

* Author to whom correspondence should be addressed; E-Mail: lzliu@ncut.edu.cn;
Tel.: +86-010-8880-1519.

Received: 2 March 2014; in revised form: 23 April 2014 / Accepted: 28 April 2014 / Published: 9 May 2014

Abstract: For Wireless Sensor Networks, energy efficiency is always a key consideration in system design. Compressed sensing is a new theory which has promising prospects in WSNs. However, how to construct a sparse projection matrix is a problem. In this paper, based on a Bayesian compressed sensing framework, a new adaptive algorithm which can integrate routing and data collection is proposed. By introducing new target node selection metrics, embedding the routing structure and maximizing the differential entropy for each collection round, an adaptive projection vector is constructed. Simulations show that compared to reference algorithms, the proposed algorithm can decrease computation complexity and improve energy efficiency.

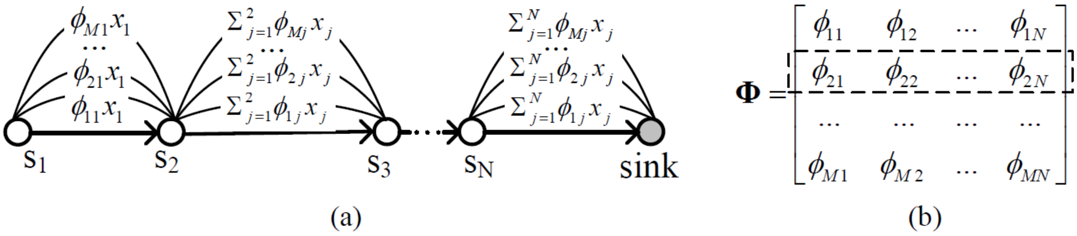
Keywords: Bayesian compressed sensing; routing structure; wireless sensor networks

1. Introduction

For data collection Wireless Sensor Networks (WSN), since sensor nodes are always densely distributed, there exists abundant redundancy in data from neighboring nodes. During the data collection process, besides how to route data to the sink, one of the key problems is how to remove redundancy and improve energy efficiency. There are a large number of algorithms that have integrated data compression into data collection [1–6]. Work described in [1,2] uses aggregation functions (e.g., $Max(\cdot)$, $Min(\cdot)$, $Average(\cdot)$, *etc.*) to extract the required information during data routing. However, while data size is reduced, data structure is lost. In [3,4] the authors adopt lifting schemes to compress data during routing. These schemes need to exchange intermediate coefficients between nodes during compression, which results in a waste of energy. Many traditional data compression algorithms such as KLT [5], wavelet [6], data mining [7], have been used in data collection. These algorithms have high computational complexity, and are difficult to integrate with routing processes. Compressed Sensing (CS) [8] is a new emerging theory in signal processing which has promising prospects to be applied in WSN [9–16]. Firstly, in CS, data sampling and compression can be integrated harmoniously into one step, which meets the requirements of WSN energy efficiency. Secondly, in compressed sensing, the computational complexity of decoding and encoding is highly asymmetrical; encoding complexity is very low, which implies a limitation for sensor nodes with weak computational power and limited energy supplies. Finally, during the encoding process in CS, no original or intermediate data is required to exchange among nodes, which is beneficial for distributed sensor network design.

However, it is not feasible to straightforwardly apply compressed sensing to WSN. Traditional projection matrices are always dense. From Figure 1, one can find that, if the projection matrix is dense, the number of packets transmitted among nodes will be very large. To increase energy efficiency, a sparse projection matrix must be utilized.

Figure 1. Relationship between routing and projection matrix: (**a**) Routing process; (**b**) Projection matrix.


$\phi_{M1}x_1$
…
$\phi_{21}x_1$
$\phi_{11}x_1$
$\Sigma_{j=1}^{2}\phi_{Mj}x_j$
…
$\Sigma_{j=1}^{2}\phi_{2j}x_j$
$\Sigma_{j=1}^{2}\phi_{1j}x_j$
$\Sigma_{j=1}^{N}\phi_{Mj}x_j$
…
$\Sigma_{j=1}^{N}\phi_{2j}x_j$
$\Sigma_{j=1}^{N}\phi_{1j}x_j$
s_1
s_2
s_3
s_N
sink
(a)
$\Phi = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1N} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2N} \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{M1} & \phi_{M2} & \cdots & \phi_{MN} \end{bmatrix}$
(b)

There is some work devoted to the design of optimized projection matrices [9–16]. In [12] an iterative algorithm to build a sparse projection matrix and decrease its coherency with the transform matrix was proposed. In each collection round, one randomly chooses one node to begin, and then selects the next node from the previous one's neighbors which can minimize the coherency between these two matrices as the next hop. This procedure loops until a complete route to a sink is formed. Work in [14] defines a metric named mutual-coherency to optimize the projection matrix. Based on this metric, a new iterative scheme is proposed to reduce the coherence step by step. Unlike the aforementioned algorithms, work in [15] suggests the use of Bayesian theory to solve the problem of projection matrix building. The main idea is that, based on some initial observed samples, one may obtain the posterior probability of the underlying signal to aid the construction of the projection vector for the next collection round. The optimization object for the new vector is to maximize the information retrieved in one collection round. However, the projection matrix formulated in this algorithm is always dense, with no routing structure embedded. Based on the framework proposed by [15], work in [16] proposes a greedy algorithm to build a sparse projection vector, which maximizes the information retrieved in one unit of energy consumption. Since the algorithm is greedy, its computation complexity is very high.

Furthermore, this algorithm tends to select only one node from a sink's one hop neighbors to build the projection vector, therefore, its reconstruction performance is degraded.

In this paper, on account of these issues, a new adaptive algorithm to design a projection vector to meet the requirements of sparseness and routing structure is proposed. The rest of this paper is organized as follows: a brief description of the Bayesian compressive sensing framework is provided in Section 2. In Section 3, the complete idea of the proposed algorithm is presented. Section 4 gives the simulation results and Section 5 concludes the work.

2. Bayesian Compressive Sensing Framework

For an underlying signal vector $\mathbf{x} \in \mathbf{R}^{N\times 1}$, which is sparse on basis $\mathbf{\Psi}$:

$$\begin{gathered}\mathbf{x} = \mathbf{\Psi z} \\ \text{s.t. } \|\mathbf{z}\|_0 = K, K \ll N\end{gathered} \tag{1}$$

if $\mathbf{x}$ is noise contaminated, and the observation process does not introduce noise, then the observed data $\mathbf{y} \in \mathbf{R}^{M\times 1}$ using projection matrix $\mathbf{\Phi} \in \mathbf{R}^{M\times N}$ can be expressed as:

$$\mathbf{y} = \mathbf{\Phi}(\mathbf{x} + \mathbf{e}_s) \tag{2}$$

where $M \geq C \cdot K \cdot \log N$, $M \ll N$, C is a constant. $\mathbf{e}_s$ is independent and identically distributed sensing noise.

Based on observed $\mathbf{y}$, adopting Bayesian learning theory, the differential entropy of the underlying signal $\mathbf{x}$ can be obtained as [15,16]:

$$h(\mathbf{x}) = -\frac{1}{2}\log|\mathbf{A} + \sigma^{-2}\mathbf{\Psi}^T\Phi^T(\Phi\Phi^{\boldsymbol{T}})^{-1}\Phi\mathbf{\Psi}| + const \tag{3}$$

where *const* is irrelevant to projection matrix, and:

$$\sum = (\mathbf{A} + \sigma^{-2}\mathbf{\Psi}^T\Phi^T(\Phi\Phi^{\boldsymbol{T}})^{-1}\Phi\mathbf{\Psi})^{-1} \tag{4}$$

$$\mathbf{A} = \begin{bmatrix} \alpha_1 & 0 & ... & 0 \\ 0 & \alpha_2 & ... & 0 \\ 0 & 0 & ... & ... \\ 0 & 0 & ... & \alpha_N \end{bmatrix} \tag{5}$$

Here $\alpha_i, (i = 1,\cdots \text{N})$ is independent and identically distributed hyper-parameter. σ^2 is the noise power. Assuming there are M_0 initial observed samples, to obtain the $(M_0 + 1)$-th sample, a new projection vector $\mathbf{r}_{M_0+1}$ must be constructed, and the new projection matrix becomes:

$$\Phi_{new} = \begin{bmatrix} \Phi \\ \mathbf{r}_{M_0+1} \end{bmatrix} \tag{6}$$

The information retrieved in the $(M_0 + 1)$-th collection round can be expressed as [16]:

$$\Delta h(\mathbf{x}) = \frac{1}{2}\log(1 + \frac{1}{\sigma^2}\frac{\mathbf{r}_{M_0+1}^T\mathbf{V}\mathbf{V}^T\mathbf{\Psi\Sigma\Psi}^T\mathbf{V}\mathbf{V}^T\mathbf{r}_{M_0+1}}{\mathbf{r}_{M_0+1}^T\mathbf{V}\mathbf{V}^T\mathbf{r}_{M_0+1}}) \tag{7}$$

which indicates the reduction of uncertainty of $\mathbf{x}$ after a new sample y_{M_0+1} has been received. In Equation (7), $\mathbf{V} \in \mathbf{R}^{N\times M_0}$ is a matrix acquired from Singular Value Decomposition of $\mathbf{\Phi}$:

$$\mathbf{\Phi} = \mathbf{U}\begin{bmatrix}\mathbf{S} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\bar{\mathbf{V}}^T \\ \mathbf{V}^T\end{bmatrix} \tag{8}$$

It is straightforward that in order to maximize the information retrieved in round $M_0 + 1$, $\mathbf{r}_{M_0+1}$ should be obtained by solving the following optimization problem:

$$\mathbf{r}_{M_0+1} = \arg\max_{\mathrm{r}} \frac{\mathbf{r}^T\mathbf{V}\mathbf{V}^T\mathbf{\Psi}\sum\mathbf{\Psi}^T\mathbf{V}\mathbf{V}^T\mathbf{r}}{\mathbf{r}^T\mathbf{V}\mathbf{V}^T\mathbf{r}} \tag{9}$$

It should be noted that, $\mathbf{r}_{M_0+1}$ obtained using the above algorithm is always dense, which means it is not energy efficient. Another problem is that this new derived projection vector does not have a routing structure embedded.

3. Adaptive Projection Vector Construction Algorithm

Firstly, we give out some assumptions for the WSN. All sensor nodes are static, and the network is fully connected. The sink knows the neighboring context of each node. As nodes are densely deployed, data from neighboring nodes is highly correlated.

Secondly, we present the constraints an optimal projection vector $\mathbf{r}_{M_0+1}$ should meet: (1) The vector should be sparse. (2) Nodes selected in the vector can form a complete routing structure to sink. (3) For all vectors with the same number of nodes selected, the vector constructed by the proposed algorithm can retrieve the maximum information. (4) To meet energy conservation constraint, $\mathbf{r}_{M_0+1}$ should be normalized:

$$\left\{\mathbf{r}_{M_0+1} \in \mathbf{R}^N, \|\mathbf{r}_{M_0+1}\|_2 = 1\right\} \tag{10}$$

Since the problem described by (1), (3), (4) is NP-hard [16] and constraint (2) makes it more difficult to solve, we try to find a solution for each constraint and to get a sub-optimal solution for the whole problem. For constraint (1), the key is to select optimal number of nodes—target nodes. Since each non-zero element in the projection vector corresponds to one sensor node, the task here is to find out the nodes from which to harvest information in $(M_0 + 1)$-th collection round. A proper metric should be defined for node selection. Constraint (2) is supplementary to the first one. When the vector obtained in the first step cannot form a complete routing structure, we need to add the least supplementary nodes to complete it. For condition (3), it is equivalent to determine the optimal projection coefficients for each selected nodes. This can be solved through maximizing the differential entropy of signal $\mathbf{x}$.

3.1. Determine the Target Nodes

Given a certain number of initial observed samples, in order to determine which nodes need to be harvested the most, we define two metrics from different points of view.

3.1.1. Principle Component Analysis Based Metric

Under the Bayesian compressive sensing framework, one can get the differential entropy of the underlying signal **x**, and then obtain a new projection vector by maximizing the reduction of uncertainty of **x**. However, the vector obtained by this means always has too many non-zero items. Since the direction of a vector is mainly determined by its principle components, we can find out the indices of L elements which have the most amplitude in the vector to represent target nodes. For a normalized vector **r** and a given threshold E_{th}, L is determined in the following way:

$$\begin{gathered} L = \min\ l \\ s.t.\ \sum_{k=1}^{l} | r_{i_k} |^2 > E_{th},\ \ i_k \in \{1, \cdots N\} \end{gathered} \tag{11}$$

3.1.2. Node's Total Coefficients Energy Based Metric

In compressive sensing theory, transform matrix **Ψ** and projection matrix **Φ** of the underlying signal **x** should be incoherent in order to improve reconstruction performance. At the same time, in order to sparse signal **x**, **Ψ** and **x** should have good coherency. That is to say, ideally, **Φ** and **x** should be completely incoherent. Therefore, for all nodes, the statistic feature of their coefficients in **Φ** should be identical, and there is no clue to find out from **Φ** that which node is more or less important. Based on this observation, for each node, we can study its projection coefficients by defining node's total coefficients energy. An example to illustrate the node's total coefficients energy for node 2 is illustrated in Figure 2.

Figure 2. Explanation of node 2's total coefficients energy.

$$\mathbf{\Phi} = \left[\begin{array}{c:c:cc} \phi_{11} & \phi_{12} & \cdots & \phi_{1N} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2N} \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{M_0 1} & \phi_{M_0 2} & \cdots & \phi_{M_0 N} \end{array}\right]$$

Define (node's total coefficients energy): Given a projection matrix $\Phi = \varphi_1, \varphi_2 \cdots \varphi_{\mathrm{N}}$, we define the total coefficients energy of node i to be the sum of absolute value of i-th column elements in matrix Φ:

$$P_i = \sum_{j=1}^{M} | \phi_{ji} | \tag{12}$$

Total coefficients energy P_i indicates the contribution of data from node i to the observed samples. Ideally, total coefficients energy for each node should be identical in a statistical sense, however, in an adaptive compressive sensing algorithm, the initial projection matrix has a relatively small number of rows, making the total coefficient energy of some nodes much less than that of others. Therefore, when a new projection vector is to be built, it is reasonable to select these nodes first.

It should be noted that, most projection matrices currently proposed meet the attributes we presented here, *i.e.*, their total coefficients energy for each node is identical in a statistical sense. For example:

(1) Gaussian random matrix: every element in the matrix follows Gaussian distribution with zero mean and 1/*N* variance.
(2) Rademancher matrix: elements of the matrix are selected randomly from set {+1, −1}.
(3) Partial Fourier matrix: randomly select *M* rows from *N*x*N* Fourier matrix.
(4) Cyclic matrix: each row of the matrix is a different arrange of a same set.

3.2. Build Routing Structure

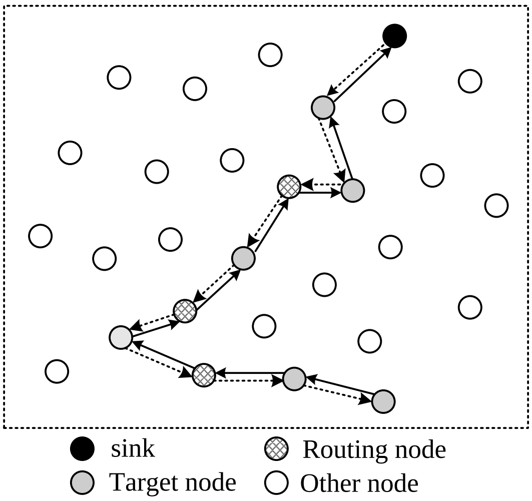
3.2.1. Style of the Routing Structure

As stated in the previous subsection, the target nodes selected can't guarantee formulation of a complete routing structure, therefore, some supplementary nodes are needed. These nodes can be used in two styles. One is to use them as routing nodes, and no data is collected from them. The other is to use them both as routing nodes and data collection nodes.

Traditional data collection schemes usually adopt the first style, as adding data collection nodes will increase the transmission packet size, and consume more energy. But in compressive sensing, in one collection round, all data will be projected into a fixed length buffer regardless of the amount of sensors, which means collecting data from routing nodes will not distinctly increase the power consumption. Moreover, there is tangible benefit for this style as it can increase the information collected and decrease the coherency between projection vectors. Therefore, we adopt the second style.

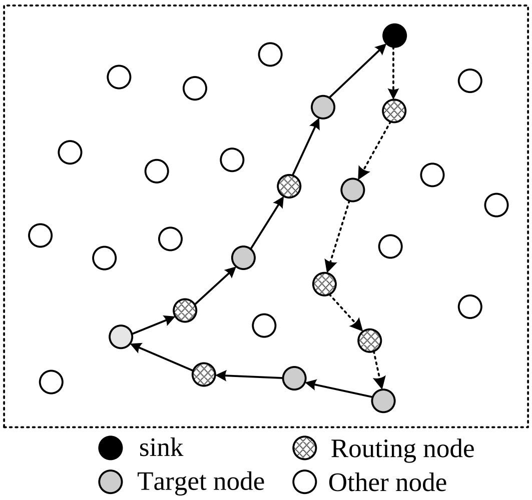
Next, let's discuss how to arrange the selected nodes to form a complete routing structure. In one data collection round, the sink needs to send out packets containing projection coefficients to selected nodes; then the corresponding node will project its data to the projection field of the packet; finally this packet will be routed back to sink. There are several routing structures to complete this procedure, shown in Figures 3 and 4.

Figure 3. Linear structure.



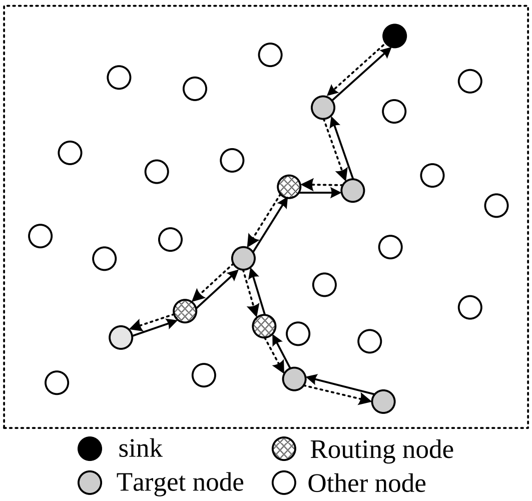
In Figure 3, all selected nodes are arranged in a linear structure; the back and forth paths of the packet are the same. In this style, except for the endpoint node, each node will receive and send the packet twice; however, only one time they can project their information into it, which means energy waste. As is shown in Figure 3, there are in total 18 transmissions, but only nine nodes' information is collected.

Figure 4. Single-ring structure.



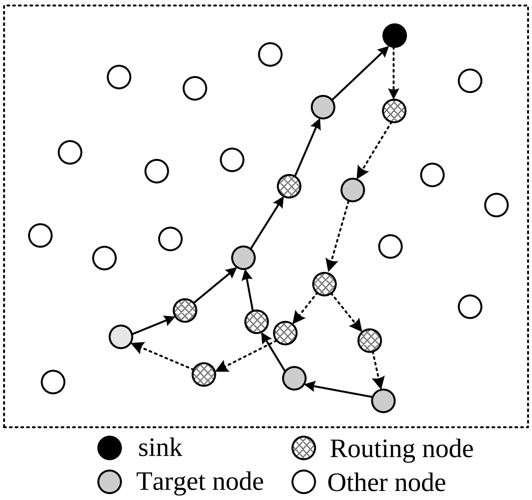
To improve this, the structure shown in Figure 4 is another option. In this style, all selected nodes are arranged in a ring structure. Every node on the path only needs to transmit the packet once. In the figure, there are totally 13 transmissions, and 12 nodes' information is collected. It's more energy efficient than structure 1. However, this structure leads to a pretty long routing path, making the time needed for one round very long in case of large scale wireless sensor networks.

Figure 5. Tree structure.



To shorten the time spent for one collection round, the structure shown in Figure 5 utilizes a tree structure. Like structure 1, data packets pass a same node twice. The structure shown in Figure 6 uses a multi-ring structure to arrange the selected nodes. Packets are duplicated into multiple copies in case of necessity utilizing the broadcast property of wireless channels to speed up the collection process. Meanwhile, multiple copies of the packet are reunited when necessary to save energy. This structure takes the advantage of the structures proposed in Figures 4 and 5, and has the virtue of both energy efficiency and short collection path. In this paper, we propose to design a multi-ring routing structure as shown in Figure 6.

Figure 6. Multi-ring structure.



3.2.2. Build the Routing Structure

However, how to build an efficient routing structure containing rings shown in Figure 6 is a problem. Fortunately, this structure can be split into tree structures. Without loss of generality, we take the structure shown in Figure 6 as an example. For each routing path with a ring, if we cut off the ring at the target node which is the most far to sink among all target nodes on that ring, and append this target node to the two new endpoints, we can get two routing trees. Each tree is rooted at sink, and has the same target nodes as leaf nodes. These trees have different direction. One tree is used to transmit packets from the sink to each selected node, and the other is used to transmit packets toward the sink. The former is called downlink tree, and the latter is called uplink tree.

Consequently, the problem of building multi-ring routing structures is transformed into building two multicast trees. There are many classic algorithms to build multicast trees, for example [17,18]. However, it should be noted that, there is some different between the proposed multicast tree and the classic one. Firstly, each of the trees built here does not need to cover all the target nodes; only the combination of the two trees to cover all target nodes is required. This makes the optimization process more complex than in a classic one. Secondly, two trees built are node disjoint, except the leaf nodes. Finally, as the path for each packet is a ring, there is no strict upstream or downstream relationship among nodes; therefore, a new packet format must be defined.

Formal Description of the Problem (Node Disjoint Double Multicast Trees, NDDMT)

Given a graph $G = (V, E), V = \{v_i, i = 1,2, \dots, N\}$ is the set of sensor nodes and E is the *set* of edges. Define D as the target nodes set. The object is to build two node disjoint multicast trees

$T_1, T_2, (T_1 \in V, T_2 \in V)$, which has minimum cost and can cover all target nodes in *D*. To build a sparse projection vector, the cost here is defined as the total number of nodes on the two multicast trees.

Define ς_i to indicate whether node v_i is in target nodes set *D*:

$$\begin{cases} \varsigma_i = 1, & v_i \in D \\ \varsigma_i = 0, & v_i \notin D \end{cases} \tag{13}$$

Define $\tau_{i,j}$ to indicate whether node v_i is on the multicast tree j, $j \in \{1, 2\}$:

$$\begin{cases} \tau_{i,1} = 1, & v_i \in T_1 \\ \tau_{i,2} = 1, & v_i \in T_2 \\ \tau_{i,j} = 0, & others \end{cases} \tag{14}$$

Define η_i to indicate whether node v_i is a leaf node:

$$\begin{cases} \eta_i = 1, & v_i \text{ is not a leaf node} \\ \eta_i = 0, & v_i \text{ is a leaf node} \end{cases} \tag{15}$$

and, define h_i to indicate the hops from v_i to sink.

Then, the optimization problem can be expressed as:

$$\begin{gathered} \min\ |T_1 \cup T_2| \\ \text{s.t. } \forall v_i \in D, \quad \sum_{j \in \{1,2\}} \tau_{i,j} > 0 \\ \forall v_i \in T_1 \cup T_2, \quad \eta_i + \sum_{j \in \{1,2\}} \tau_{i,j} = 2 \\ \forall v_i \in \{v_j \mid \eta_j = 0\}, \quad \varsigma_i \cdot (1 - \eta_i) = 1 \\ \forall v_i \in V - (T_1 \cup T_2), \quad \sum_{j \in \{1,2\}} \tau_{i,j} = 0 \end{gathered} \tag{16}$$

The object function means to find two trees with least nodes on them. Condition 1 means each node in set *D* must be covered by at least one tree. Condition 2 has two implications. For a leaf node, it must be covered by both two trees; for a non-leaf node, it is covered by only one tree. Condition 3 means that only node in the target set can be used as leaf node. Condition 4 refers to the node outside of the multicast trees.

Property. NDDMT problem is NP-hard.

Proof. Study one of its sub-problems. If this sub-problem is NP-hard, then the original problem is NP-hard. Let's reduce the optimization space to the problem of building two trees separately. First, we build the downlink tree T_1 to cover all the nodes in *D*, which is denoted as problem P_1. Then, we build the uplink tree T_2 to cover the leaf nodes of T_1, which is denoted as problem P_2. Since both problem P_1 and P_2 are standard Steiner tree problems [19], which is NP-hard, NDDMT is NP-hard.

Algorithm to Build NDDMT

Since the NDDMT problem is NP-hard, we design a two steps heuristic algorithm to solve this problem. Firstly, in graph $G = (V, E)$, build multicast tree T_1 rooted at the sink to cover all target nodes. Secondly, in graph $G' = (V', E')$, build a tree T_2 which can cover all leaf nodes on T_1, where V' is the set of nodes in V except non-leaf nodes on tree T_1. The optimization object for each tree is to minimize the number of nodes on it. Since the nodes on tree T_1 is excluded when building tree T_2, the two trees are node disjoint. As sensor nodes are always densely deployed and the target nodes set D is always small, so after deleting nodes on tree T_1, the nodes in V' still have a high probability to build tree T_2. If some nodes in the graph are unconnected when building tree T_2, we can reuse nodes in tree T_1 to ensure the success of the building process.

The Steiner tree building problem is to let nodes on the tree share their path as much as possible, so as to minimize energy cost. There are many classic Steiner tree building algorithms, such as MPH [17], ADH [18]. In this paper, if we define the cost of each edge to be 1, then the problem of building Steiner tree with minimum cost is equivalent to the problem of building a multicast tree with minimum nodes.

Uplink tree (T_1) building algorithm:

Let $T_{1,t}$ denote the partial tree built at step *t*, and $C_{1,t}$ be the cost of $T_{1,t}$. Let $D_{1,t}$ denote the set of target nodes that haven't joined tree T_1 till step *t*. Denote *s* to represent sink.

INPUT: graph $G = (V, E)$, target nodes set D.

OUTPUT: tree T_1 and its cost C_1.

(1) Initially, $t = 0$, $T_{1,0} = \{s\}, C_{1,0} = 0, D_{1,0} = D$.
(2) At step *t*, for each $v_i \in D_{1,t}$, use Dijkstra algorithm to calculate its cost (denoted as c_i) to tree $T_{1,t}$ Define $T_{1,t}^{'}$ to be a virtual tree if v_i joins $T_{1,t}$, then the total cost of $T_{1,t}^{'}$ is $C_{1,t} + c_i$.
(3) Based on their distance to $T_{1,t}^{'}$, sort the reminder target nodes in $(D_{1,t} - v_i)$ from near to far, and calculate the total cost $\overline{c}_i$ for them to join tree $T_{1,t}^{'}$.
(4) Choose the node v_i which can minimize $C_{1,t} + c_i + \overline{c}_i$ to join tree $T_{1,t}$.
(5) Let $t = t + 1, D_{1,t} = D_{1,t} - v_i$, go to (2) unless all target nodes have joined the tree.

Building the algorithm for tree T_2 is similar to T_1. It is omitted here due to space limitations.

Complexity

First, calculate the shortest path from each target node to every other node in advance. The complexity for this step is $O(LN^2)$. In the process of multicast tree building, each iteration will try to add one target node to the tree; computational complexity for each iteration is $O(L^2)$. In total there are L iterations, so the computational complexity for all iterations is $O(L^3)$. In conclusion, the computational complexity for double multicast trees building is $O(LN^2 + L^3)$.

3.2.3. Packet Structure

Assume the wireless channel is ideal. From the previous subsection, we are informed that in compressive sensing, one routing structure will be maintained only during one collection round, and there is no distinct downstream or upstream relationship between nodes on a path. Therefore, the routing problem here is different from that in a classic WSN. To solve this problem, a new data packet structure is required.

Figure 7. Packet structure.

Sender ID	SeqNo	ProjectionValue	Routing & Coefficients

As is shown in Figure 7, the *SenderID* field contains the ID of the node that sends the current packet. *SeqNo* field is used to uniquely identify one collection round. Due to the broadcast nature of the wireless channel, all neighbors of a sender can receive its packet. This feature can be utilized to improve energy efficiency. When receiving a packet, a node (assumed to be v_i) checks the *Routing & Coefficients* field to determine if it is one of the children of the sender. If it is false, the packet will be dropped, otherwise v_i finds out its projection coefficient r_i in the packet and updates the *Projection Value* field using:

$$projection_{new} = (projection_{old} + x_i \cdot r_i) / child_i \tag{17}$$

where $child_i$ indicates the number of children of v_i. The reason to divide the projection value by $child_i$ is that the packet from v_i will be received by all its children, which is illustrated in Figure 8 as an example.

Figure 8. An example of the routing structure.

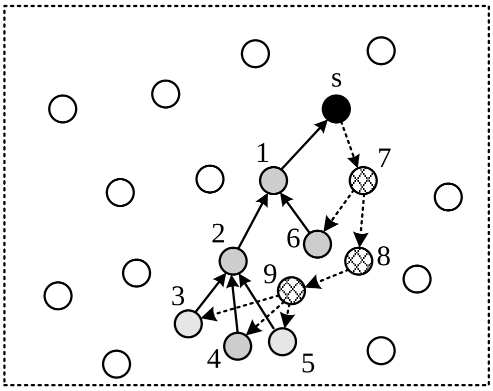


Table 1. Routing & coefficients table of Figure 8.

Node	Upstream	Downstream	Coeff.
s	1	7	N.A.
1	2,6	s	r1
2	3,4,5	1	r2
3	9	2	r3
4	9	2	r4
5	9	2	r5
6	7	1	r6
7	s	6,8	r7
8	7	9	r8
9	8	3,4,5	r9

Next, v_i updates the *Routing & Coefficients* field by deleting the expired information, and broadcasts the packet to its children. (Table 1 illustrates the routing & coefficients table built from Figure 8). If a node is the child of multiple nodes, it reunites the packets received from all of its parents to one packet before sending it to its child.

3.2.4. Determine Projection Coefficients

Once the nodes for one collection round are determined, we need to compute the projection coefficients for them. Equation (9) can be used to accomplish this task. Due to the constraint of sparseness, projection vector $\mathbf{r}_{M_0+1}$ has only small part of elements with non-zero value. To obtain $\mathbf{r}_{M_0+1}$, first delete the rows and columns in $\mathbf{VV}^T\mathbf{\Psi}\sum\mathbf{\Psi}^T\mathbf{VV}^T$ and $\mathbf{VV}^T$ corresponding to the position of unselected nodes; then use the shrunk matrix pencil to determine the non-zero elements in $\mathbf{r}_{M_0+1}$.

4. Simulation and Discussion

4.1. Simulation Scenarios

Assume nodes are uniformly distributed. The sink is located at the center of the area. The normalized communication radius of each node is r_{c}, $r_{\mathrm{c}} = 1.5$. Assume the communication media is ideal, with no channel noise or access collision. To evaluate the proposed algorithm, three related algorithms are selected as reference, including adaptive CS [16], adaptive Laplace BCS (adaptive BCS) [20], and Compressed Sensing with Random Walk routing (CS-RW) [21].

In the simulation, the maximum length of observation vector $\mathbf{y}$ is $M = 120$. M includes two parts: (1) initial observation vector with length M_0; (2) adaptive observation vector with length $M - M_0$. The initial observation vector is used as the basis for the adaptive collection stage. For adaptive BCS, the initial projection matrix is generated at the sink before collection, and then transmitted to all sensor nodes using a traditional routing mechanism. During the adaptive collection process, for each collection round, the sink determines a projection coefficient for each node and sends it out using the same routing path. For CS-RW, the probability to choose a node is fixed at 0.5.

Data used in simulation is generated using a computer. The model for data generation only assumes the data is compressible, while its sparse structure is unknown [22]. It is described as follows: assume nodes are uniformly distributed in the area, and $d_{i,j}$ is the distance between node i, j. Define the correlation between data from node i, j to be $c_{ij=e^{-\beta d_{ij}}}$, where β indicates the degree of correlation. Then, the matrix $\mathbf{C}$ describes the correlation is:

$$C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1N} \\ c_{21} & c_{22} & \cdots & c_{2N} \\ \vdots & & & \vdots \\ c_{N1} & c_{N2} & \cdots & c_{NN} \end{bmatrix} \tag{18}$$

Perform Cholesky decomposition to, $\mathbf{C}$:

$$\mathbf{C} = \mathbf{G} \cdot \mathbf{G}^T \tag{19}$$

and assume $\mathbf{n}_0 \in \mathbf{R}^N$ is a uniform random vector, with each element independent and identically distributed. Then, data vector $\mathbf{x}$ is generated as:

$$\mathbf{x} = \mathbf{G} \cdot \mathbf{n}_0 + \mathbf{n}_s \tag{20}$$

where $\mathbf{n}_s$ denotes sensing noise.

4.2. Evaluation Metric

4.2.1. Reconstruction Error

Assume the original data vector is $\mathbf{x}$ ($x_i, 1 \le i \le N$), and the reconstruction data vector is $\hat{\mathbf{x}}$ ($\hat{x}_i, 1 \le i \le N$), then the reconstruction error is defined as:

$$err = \frac{\| \mathbf{x} - \hat{\mathbf{x}} \|_2^2}{\| \mathbf{x} \|_2^2} = \frac{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2}{\sum_{i=1}^{N} x_i^{\ 2}} \tag{21}$$

4.2.2. Energy Cost

Energy cost is a key metric for wireless sensor networks. In the simulation, only the energy used in data transmission is counted, and the energy used for sensing and computation is not included. Energy consumed by the sink is not included either. Assume the transmission power is fixed, then the energy cost of a node is proportional to the number of bytes it sent.

4.2.3. Computation Complexity

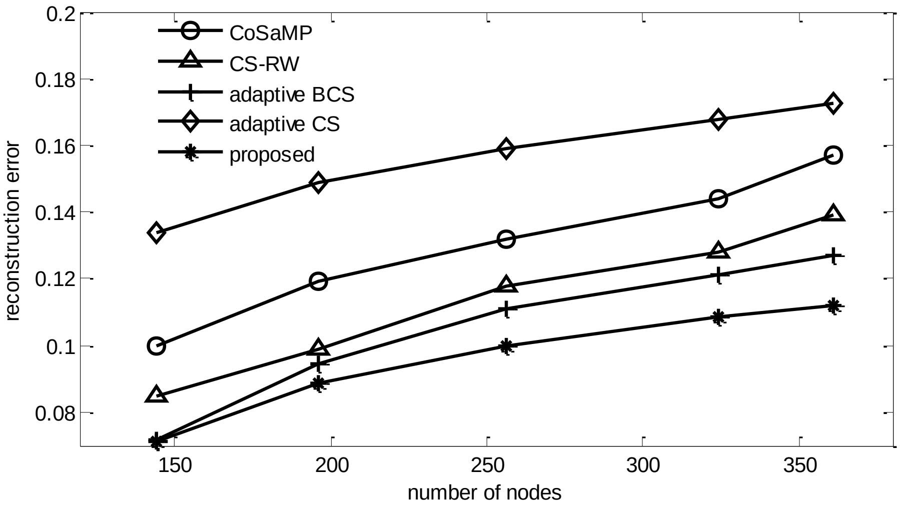
For compressive sensing, a node only needs to perform one multiplication and one addition operation in one projection round; therefore, the computational complexity for a node is negligible. However, the computation complexity for the sink, which includes three parts: BCS reconstruction, projection route building and projection vector computation, is much higher. In the simulation, the time elapsed in one simulation is used as metric to evaluate an algorithm's computational complexity. The simulation performed on a PC equipped with an Intel E2200 dual core processor.

4.3. Simulation Results

4.3.1. Reconstruction Error

From Figure 9, it can be found that, when the data is contaminated by noise, reconstruction Bayesian-based algorithms outperform CoSaMP [23]. Meanwhile, when the number of collection rounds is fixed, the reconstruction error will increase with the increase of node number *N*. Among all five algorithms, the proposed algorithm performs the best. When *N* = 400, it can reduce the reconstruction error by 2.3% in comparison with adaptive BCS. Although the adaptive method is utilized in adaptive BCS too, its performance is worse than that of the proposed algorithm, due to the fact that the Bayesian learning process is likely to choose the same basis, which leads the algorithm to fall into a local minimum [24]. Although a random noise disturbance can be added to the vector to help the learning process jump out of local minima, the performance improvement is limited while the energy cost increases dramatically. In the figure, it can be found that adaptive CS always performs the worst among the five algorithms. This is due to the fact that this algorithm tends to choose only one node in a sink's one hop neighbors as its target node. The reason for this is that adaptive CS uses $\Delta h(\mathbf{p}) / E(\mathbf{p})$ as a metric to select the path. As $E(\mathbf{p})$ is counted in the hop, when the path length increases from 1 hop to 2 hop, $E(\mathbf{p})$ increases once. However, as data from neighbor nodes has a strong correlation, in most cases, the increase of information $\Delta h(\mathbf{p})$ can't increase once. Therefore, adaptive CS tends to choose only one node among the sink's one hop neighbors.

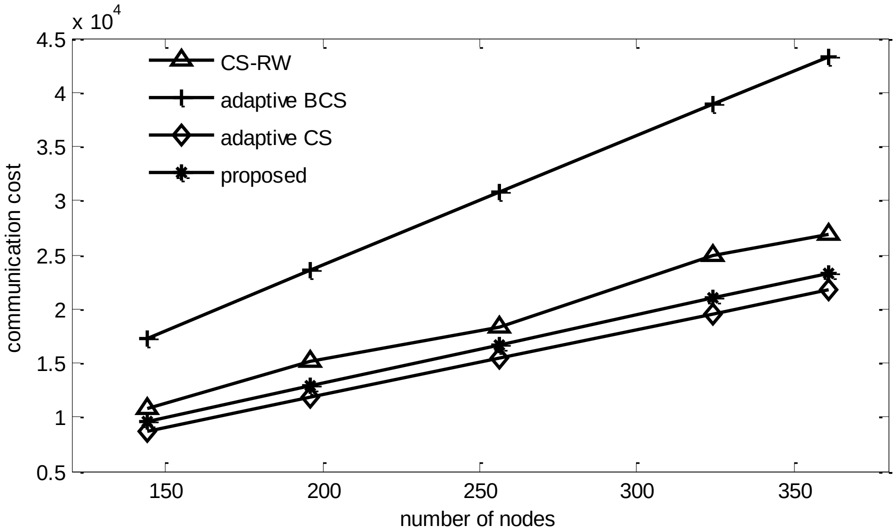
Figure 9. Reconstruction error *vs.* number of nodes.



4.3.2. Energy Cost

It can be found from Figure 10 that the cost of adaptive BCS surpasses the others.

Figure 10. Communication cost *vs.* number of nodes.

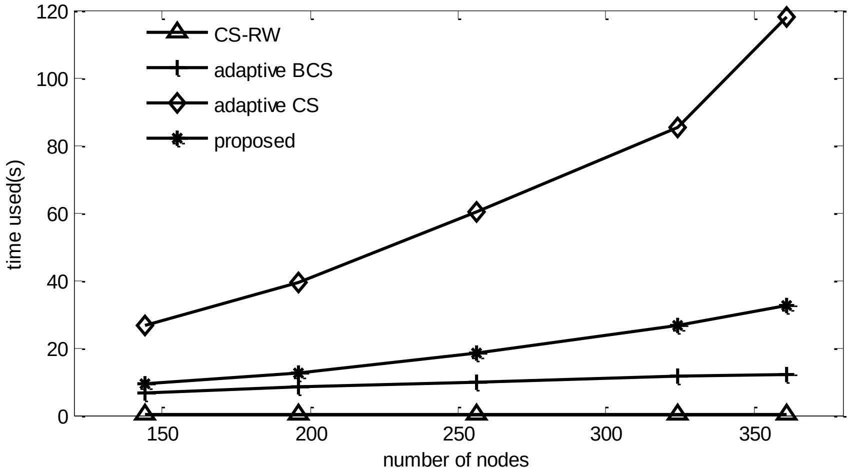


This is due to the fact adaptive BCS adds a random disturbance noise to improve the performance, which leads to a dense projection matrix. Meanwhile the projection vector built by the proposed algorithm has the majority part of elements in the projection vector as zeros, and can save 50% energy cost compared to adaptive BCS. As shown in the figure, adaptive CS consumes the minimum energy among all the algorithms, since it only collects information from one node in most collection rounds. The proposed algorithm consumes 8% more energy than adaptive CS, while the improves reconstruction performance by 6%.

4.3.3. Computation Complexity

Since CS-RW does not use the adaptive mechanism and only needs to perform one reconstruction procedure, its simulation time is the lowest among the four algorithms. Adaptive BCS performs one reconstruction for each collection round, so its simulation time increases dramatically compared to BCS, while the adaptive CS, in addition to performing the reconstruction for each collection round, should calculate routing and projection vectors, so its simulation time increases dramatically with the number of nodes. From Figure 11, it can be found that the computational complexity of the proposed algorithm is much lower than that of adaptive CS.

Figure 11. Computation complexity *vs.* number of nodes.

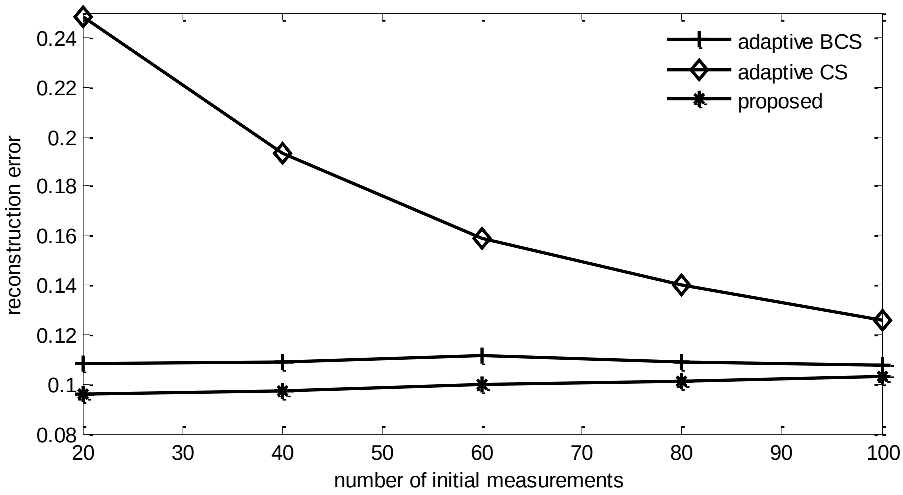


4.3.4. Impact of Initial Observation Length (M_0)

From Figure 12, it can be found that the reconstruction error of all listed algorithms increases with the increase of M_0.

This is due to the fact that if the total round of observations is fixed, increasing initial observations length will decrease the number of adaptive collection rounds, therefore, the performance will be degraded. It also can be found that the reconstruction error of adaptive CS will reduce with the increase of the initial observation length. This is also due to the reason explained previously. Adaptive CS tends to choose only one node in sink's one hop neighbors to collect information, therefore, the more rounds of adaptive collection performed, less information will be collected.

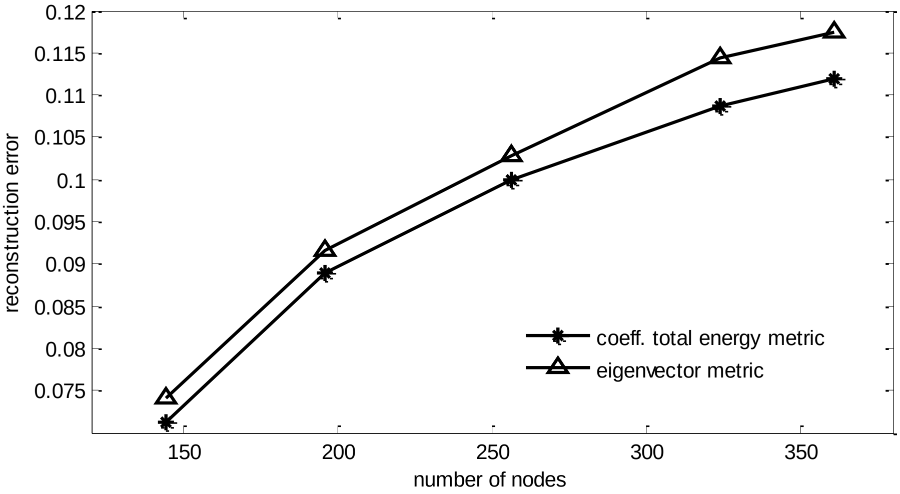
Figure 12. Reconstruction error *vs.* initial observation length.



4.3.5. Impact of Target Node Selection Metric

In this paper, two target nodes selection metric are proposed. Here we compare their performance. It can be found from Figure 13 that, the reconstruction error using node's total coefficients energy as metric is smaller than using principle components of the eigenvector as metric. The reason lies in that the Bayesian learning process tends to select the same basis, which will reduce the difference between projection vectors and degrade reconstruction performance. On the contrary, node's total coefficients energy metric always can build a different projection vector.

Figure 13. Reconstruction error for different node selection metrics.



4.3.6. Number of Target Nodes

Figures 14 and 15 show the impact of the target nodes number for tree building.

It can be found that with the increase of target nodes, there is some degree of decrease in reconstruction error. However, the communication cost increases rapidly too. From the figure we can find that, when target nodes increase from 2 to 10, communication costs increase by 12%, while the reconstruction error only decreases 0.6%. Therefore, a proper number of target nodes should be selected. In the simulations, this number is fixed at four.

Figure 14. Energy cost *vs.* number of target nodes.

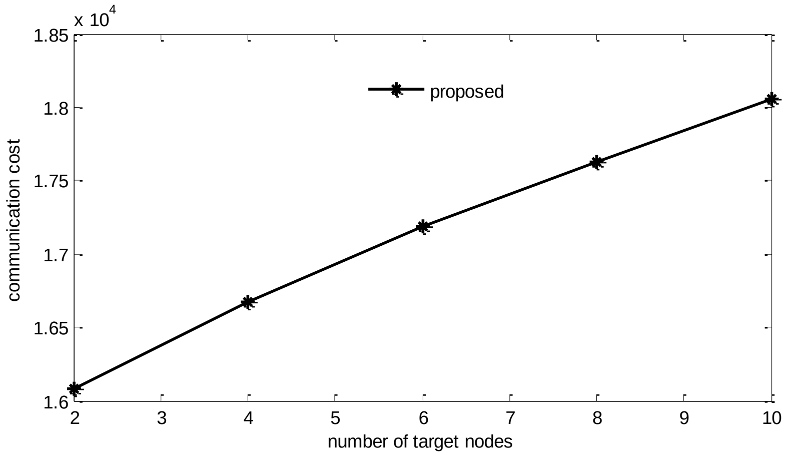


Figure 15. Reconstruction error *vs.* number of target nodes.

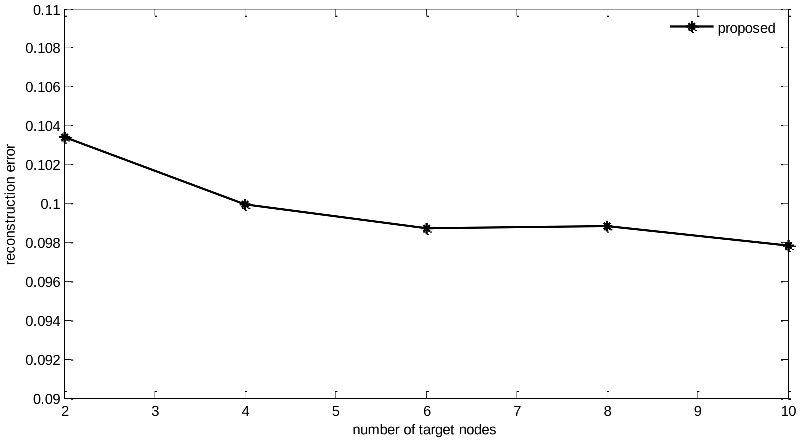
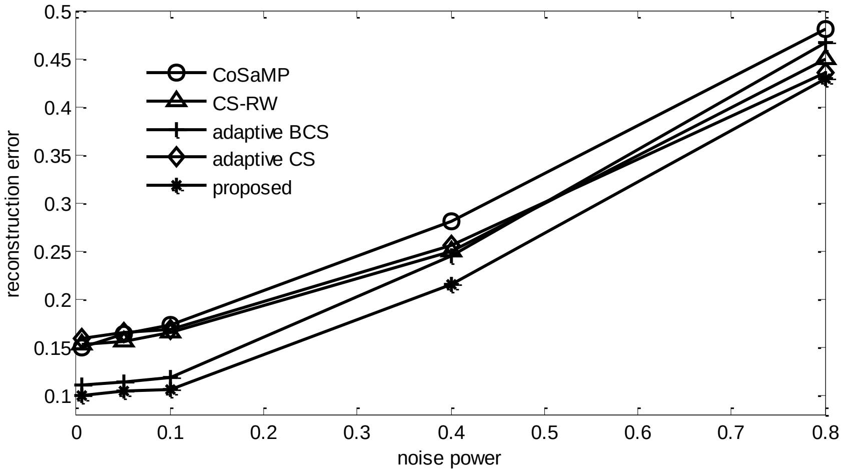


Figure 16. Impact of noise power.



4.3.7. Noise Power

From Figure 16, it can be found that, with increasing noise power, the reconstruction error of all algorithms will increase. Among them, the proposed algorithm has the best reconstruction performance in all experiments.

5. Conclusions

In this paper, a new adaptive data collection algorithm based on a Bayesian compressive sensing framework is proposed. Since the problem of finding the optimum projection vector with routing structure embedded is NP-hard, we propose a heuristic algorithm with three steps, including target nodes selection, routing structure building and projection coefficients computation. Firstly, by defining two metrics, named principle component of eigenvector and node's total coefficients power, the sub-problem of target nodes selection is solved. Secondly, we transform the problem of routing structure building to the problem of building double node disjoint broadcast trees. Finally, utilizing the theory of maximizing the differential entropy, the optimum coefficient for each selected node is obtained. Simulation results show that, compared with the reference algorithms, the proposed algorithm has better reconstruction performance with lower communication cost. Future research will study the problem of adaptive compressive sensing in a lossy communication medium and to parallelize the reconstruction process in a multi-core processor platform.

Acknowledgment

This work is partially supported by the National Natural Science Foundation of China (No.61370111 and No.61103113) and Beijing Municipal Education Commission General Program (KM201310009004).

Author Contributions

Zhi Liu designed the study, conducted the experiments and data analysis and prepared the manuscript; Mengmeng Zhang participated in data analysis and manuscript preparation; Jian Cui aided the development of software program and provided reviewing for this article.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Roy, S.; Conti, M.; Setia, S.; Jajodia, S. Secure Data Aggregation in Wireless Sensor Networks: Filtering out the Attacker's Impact. *IEEE Trans. Inf. Forens. Secur.* **2014**, *9*, 681–694.
2. Krishna, M.B.; Vashishta, N. Energy Efficient Data Aggregation Techniques in Wireless Sensor Networks. In Proceedings of the 5th International Conference on Computational Intelligence and Communication Networks (CICN), Mathura, India, 27–29 September 2013; pp. 160–165.

3. Beaudaux, J.; Gallais, A.; Montavont, J. LIFT: Layer Independent Fault Tolerance Mechanism for Wireless Sensor Networks. In Proceedings of the IEEE 73rd Vehicular Technology Conference (VTC Spring), Yokohama, Japan, 15–18 May 2011; pp. 1–5.
4. Temel, S.; Unaldi, N.; Kaynak, O. On Deployment of Wireless Sensors on 3-D Terrains to Maximize Sensing Coverage by Utilizing Cat Swarm Optimization with Wavelet Transform. *IEEE Trans. Syst. Man. Cybern. Syst.* **2014**. *44*, 111–120.
5. Shen, G.; Narang, S.K.; Ortega, A. Adaptive distributed transforms for irregularly sampled Wireless Sensor Networks. In Proceedings of the IEEE International Conference on 2009 Acoustics, Speech and Signal Processing, Taipei, Tanwan, 19–24 April 2009; pp. 2225–2228.
6. Wang, C.; Ma, H.D.; He, Y.; Xiong, S.G. Adaptive Approximate Data Collection for Wireless Sensor Networks. *IEEE Trans. Parallel. Distrib. Syst.* **2012**, *23*, 1004–1016.
7. Abdelmoghith, E.M.; Mouftah, H.T. A data mining approach to energy efficiency in Wireless Sensor Networks. In Proceedings of the 2013 IEEE 24th International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), London, UK, 8–11 September 2013; pp. 2621–2626.
8. Donoho, D.L. Compressed Sensing. *IEEE Trans. Inf. Theory* **2006**, *52*, 1289–1306.
9. Barcelo-Llado, J.E.; Perez, A.M.; Seco-Granados, G. Enhanced Correlation Estimators for Distributed Source Coding in Large Wireless Sensor Networks. *IEEE Sens. J.* **2012**, *12*, 2799–2806.
10. Karakus, C.; Gurbuz, A.C.; Tavli, B. Analysis of Energy Efficiency of Compressive Sensing in Wireless Sensor Networks. *IEEE Sens. J.* **2013**, *13*, 1999–2008.
11. Li, S.J.; Qi, H.R. Distributed Data Aggregation for Sparse Recovery in Wireless Sensor Networks. In Proceedings of the 2013 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), Cambridge, MA, USA, 20–23 May 2013; pp. 62–69.
12. Lee, S.; Pattem, S.; Sathiamoorthy, M.; Krishnamachari, B.; Ortega, A. *Compressed Sensing and Routing in Multi-Hop Networks*; Technical Report; University of Southern California: Los Angeles, CA, USA, 2009.
13. Moghaddam, B.; Weiss, Y.; Avidan, S. Generalized spectral bounds for sparse LDA. *Proc. Int. Conf. Mach. Learn.* **2006**, doi:10.1145/1143844.1143925.
14. Quer, G.; Masiero, R.; Munaretto, D. On the interplay between routing and signal representation for Compressive Sensing in wireless sensor networks. In Proceedings of the Information Theory and Applications Workshop, San Diego, CA, USA, 9–14 February 2009; pp. 206–215.
15. Ji, S.H.; Xue, Y.; Carin, L. Bayesian Compressive Sensing. *IEEE Trans. Signal Process.* **2008**, *56*, 2346–2356.
16. Chun, T.C.; Rana, R.; Wen, H. Energy efficient information collection in wireless sensor networks using adaptive compressive sensing. In Proceedings of the IEEE 34th Conference on Local Computer Networks, LCN 2009, Zurich, Switzerland, 20–23 October 2009; pp. 443–450.
17. Takahashi, H.; Matsuyama, A. An approximate solution for the steiner problem in graphs. *Math. Jpn.* **1980**, *24*, 573–577.
18. Rayward-Smith, V.J. The computation of nearly minimal steiner trees in graphs. *Int. J. Math. Educ. Sci. Technol.* **1983**, *14*, 15–23.

19. Karp, R.M. Reducibility among combinatorial problems. In *Complexity of Computer Computations*; Miller, R.E., Thatcher, J.W., Bohlinger, J.D., Eds.; Plenum Press: New York, NY, USA, 1972; pp. 85–103.
20. Babacan, S.D.; Molina, R.; Katsaggelos, A.K. Bayesian Compressive Sensing Using Laplace Priors. *IEEE Trans. Image. Process.* **2010**, *19*, 53–63.
21. Minh, T.N. Minimizing Energy Consumption in Random Walk Routing for Wireless Sensor Networks Utilizing Compressed Sensing. In Proceedings of the 8th International Conference on System of Systems Engineering, Maui, HI, USA, 2–6 June 2013; pp. 297–301.
22. Muttereja, A.; Raghunathan, A.; Ravi, S.; Jha, A.K. Active Learning Driven Data Acquisition for Sensor Networks. In Proceedings of the 11th IEEE Sym. on Computers and Communications (ISCC'06), Cagliari, Italy, 26–29 June 2006; pp. 929–934.
23. Needell, D.; Tropp, J.A. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmonic. Anal.* **2008**, *26*, 301–321.
24. Tipping, M.E.; Faul, A.C. Fast marginal likelihood maximisation for sparse Bayesian models. In Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, Key West, FL, USA, 3–6 January 2003; pp. 3–6.