Energy and Latency Analysis for In-network Computation with Compressive Sensing in Wireless Sensor Networks

Haifeng Zheng[+§], Shilin Xiao[+], Xinbing Wang[*], Xiaohua Tian[*]
[+]State Key Lab. of Adv. Opt. Commun. Syst. and Networks, Shanghai Jiao Tong University, China
[*]Dept. of Electronic Engineering, Shanghai Jiao Tong University, China
[§]College of Physics and Information Engineering, Fuzhou University, China
Email: {zhenghf, slxiao, xwang8, xtian}@sjtu.edu.cn

***Abstract*—In this paper, we study data gathering with compressive sensing from the perspective of in-network computation in random networks, in which n nodes are uniformly and independently deployed in a unit square area. We formulate the problem of data gathering to compute multiround random linear function. We study the performance of in-network computation with compressive sensing in terms of energy consumption and latency in centralized and distributed fashions. For the centralized approach, we propose a tree-based protocol for computing multiround random linear function. The complexity of computation shows that the proposed protocol can save energy and reduce latency by a factor of $\Theta(\sqrt{n/\log n})$ for data gathering comparing with the traditional approach, respectively. For the distributed approach, we propose a gossip-based approach and study the performance of energy and latency through theoretical analysis. We show that our approach needs fewer transmissions than the scheme using randomized gossip.**

I. Introduction

Wireless sensor networks (WSNs) consisting of a large number of nodes, are usually deployed in a large region for a variety of monitoring tasks, network diagnosis, etc [1]. Such networks are typically designed to sense a field of interest, process sensed values, and transport data to a designed node (sink). It is inefficient in many situations for sensor nodes to directly transmit all the raw data to the sink. In particular, a sensing field usually exhibits high correlation between the measured data and can be compressible in some transform domains. It is possible to deliver less data to the sink without sacrificing the salient information. Therefore, it is desired to cooperate between the nodes and process the data in the networks so that the transport load can be reduced. However, conventional aggregation techniques only capture some limited statistical qualities, such as maximum, average of the data. Fortunately, compressive sensing (CS) provides an approach to process and transport correlated data in an efficient manner [2].

In this paper, we consider the application of compressive sensing in a data gathering scenario, where the data collected by the sink is assumed to be correlated. We take a new look at this recurring issue from the perspective of distributed function computation [3]. We attempt to formulate the problem of data gathering to compute multiround random linear function. Such a function has much lower dimensions than the original signal, which reduces the number of measurements that need to be transported in the network. In this paper, we study the performance of routing and computing functions in random geometric network in terms of energy consumption and latency. To the best of our knowledge, this is the first work to analyze the performance of energy consumption and latency for data gathering with compressive sensing from the perspective of in-network computation.

The focus of this paper concentrates on how to efficiently compute multiround random linear function in wireless sensor networks. To this end, we propose two protocols: tree-based protocol and gossip-based protocol. In the tree-based protocol, sensor nodes receive the data from the children, compute the function and forward the result to the parent. Finally, the computation result is obtained at the sink which completes one round of in-network function computation. However, the tree-based protocol is susceptible to the failure of nodes and links. Specially, the failure of the sink node will cause the loss of the computation results. To solve this problem, we further propose a gossip-based protocol, where the computation results can be available in each sensor node. Our contributions are summarized as follows:

- We formulate the problem of data gathering into in-network function computation. We construct multiround random linear function, and devise protocols for evaluating such function computing in wireless sensor networks.
- We propose a tree-based protocol with efficient energy consumption to compute multiround round linear function. We analyze the scaling laws for energy and latency in random geometric networks. We show that this approach is efficient, which can significantly save energy and reduce latency comparing with the traditional transmission approach.
- We propose a gossip-based protocol to compute multiround linear function, which is robust to topology changes. We derive the bounds of energy consumption and latency based on the eigenstructure of the underlying graph in random geometric networks.

The remainder of this paper is organized as follows. In Section II, we give the problem formulation and the network model used in the paper. In Section III and IV, we propose a tree-based protocol and a gossip-based protocol for computing multiround random linear function, respectively. Then we analyze the performance of computation in terms of energy consumption and latency for each protocol. Finally, we conclude the paper in Section V.

II. Problem Formulation and Network Model

A. Problem Formulation

Considering the scenario where the sink needs to collect the data from n sensor nodes in the network. At a sampling instant, sensor node i takes a measurement x_i. Let $\mathbf{x} = (x_1, \cdots, x_n)$ denote the vector of measurements sampled by sensor nodes, where $\mathbf{x}$ is compressible. The processing of data gathering with compressive sensing consists of two parts: collecting random projections $\mathbf{y}$ and recovering the signal $\mathbf{x}$ from $\mathbf{y}$. In fact, the former part can be viewed as the problem of in-network function computation. The target function can be represented as a multiround random linear function, which has the following form

$$\mathcal{F} : \mathbf{x}^n \rightarrow \mathbf{y} \tag{1}$$

where $\mathbf{y}$ is the vector of random projections received by the sink, i.e., $\mathbf{y} = \{y_1, \cdots, y_m\}$. The function $\mathcal{F}_j$ can be written as $\sum_{i=1}^{n} \Phi_{ij} x_i$ when the jth random projection y_j is computed, where Φ_{ij} can be the entries of a random Gaussian or Bernoulli matrix. To simplify the analysis, we assume that $m = s \log n = \Theta(\log n)$ random projections are sufficient to recover an s-sparse signal $\mathbf{x}$ when n is large. In this paper, we focus on designing protocols to perform computation of such a function in a random network. To measure the efficiency of a protocol, we consider energy consumption and latency of a protocol, which are measured by the number of transmissions and the number of time slots that the protocol takes to compute one round of a function, respectively.

B. Network Model

In this paper, we model our wireless sensor network as a random geometric graph $G(V, E)$, which consists of n nodes randomly deployed in a unit square area. We assume all nodes share a common wireless channel and the transmission range of the nodes is denoted by $r(n)$. We adopt the protocol model [7]. Furthermore, we divide the unit square area into cells with side length $c_n = \sqrt{2 \log n / n}$. A K^2-TDMA cell scheduling scheme is adopted in this paper, where K^2 colors are used to schedule cells transmissions. Each time slot corresponding to each color is assigned to one of K^2 cells in a super cell, which is composed of $K \times K$ cells. If $K \geq 2 + (1 + \Delta) 2\sqrt{2}$, there exists a TDMA scheme such that one node per cell with the same color can simultaneously transmit a packet to all nodes in adjacent cells successfully, where Δ is the size of the guard zone to prevent interference.

III. Tree-Based Computation Protocol with Compressive Sensing

In this section, we propose a tree-based protocol to compute multiround random linear function. The protocol performs intra-cell and inter-cell computations to deliver the fused results from the nodes to the sink.

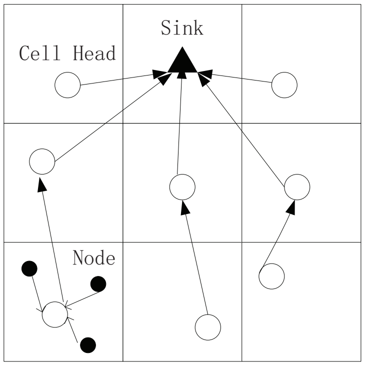

Fig. 1. A spanning tree for computation protocol.

A. Protocol

Now we consider the protocol in details. As mentioned in Section II, since the unit square area is divided into cells with side length $c_n = \sqrt{2 \log n / n}$, the total number of cells is $l = \lceil \sqrt{n / 2 \log n} \rceil^2$. Each cell contains $\Theta(\log n)$ nodes with high probability [8]. In each cell, a node is randomly selected as a cell head. We form a spanning tree, as shown in Fig. 1, where the sink is designated as the root, the vertices include all the cell heads and the links connect only between the adjacent cell heads. The proposed protocol is composed of two protocols: an intra-cell protocol and an inter-cell protocol.

Intra-cell protocol:

In each cell, one node is randomly selected as a cell head. We denote H_j as the cell head and n_j as the number of nodes in the jth cell C_j where $j = 1, \cdots, l$. For each time slot, the nodes in the jth cell take turns to transmit their data to the cell head H_j. Note that there are $n_j = \Theta(\log n)$ nodes in each cell, and thus the cell head H_j has $\Theta(\log n)$ measurements including its own measurement in its transmitting buffer.

Inter-cell protocol: In this stage, the ith random projection y_i is computed and transmitted along the tree to the sink. Computation is performed from the bottom of the tree to the root. The computation processing is illustrated in Fig. 2. Let d_j^k with $k = 1 \cdots n_j$ be the data collected by a cell head including its own packet in the cell C_j, where k is the number index of nodes. After the cell head H_j receives the data $y_{i,j-1}$ from the child H_{j-1} in the cell C_{j-1}, the cell head generates n_j random coefficients $\Phi_{i,j}^k$, computes the value $\sum_{k=1}^{n_j} \Phi_j^k d_j^k$ and updates the received data by computing

$$y_{i,j} = y_{i,j-1} + \sum_{k=1}^{n_j} \Phi_{i,j}^k d_j^k \tag{2}$$

and sends out $y_{i,j}$ to the parent H_{j+1} in the cell C_{j+1}. In this way, data is aggregated and computed along the tree to the sink. This process is repeated for m times and the sink receives m random projections. Finally, the sink recovers the data from the received random projections.

B. Analysis

We first analyze the computation complexity in the intra-cell protocol in terms of the number of transmissions and time

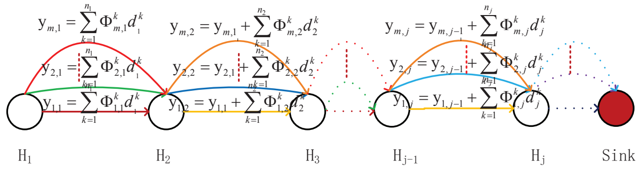
$y_{m,1} = \sum_{k=1}^{n_1} \Phi_{m,1}^k d_1^k$
$y_{m,2} = y_{m,1} + \sum_{k=1}^{n_2} \Phi_{m,2}^k d_2^k$
$y_{m,j} = y_{m,j-1} + \sum_{k=1}^{n_j} \Phi_{m,j}^k d_j^k$
$y_{2,1} = \sum_{k=1}^{n_1} \Phi_{2,1}^k d_1^k$
$y_{2,2} = y_{2,1} + \sum_{k=1}^{n_2} \Phi_{2,2}^k d_2^k$
$y_{2,j} = y_{2,j-1} + \sum_{k=1}^{n_j} \Phi_{2,j}^k d_j^k$
$y_{1,1} = \sum_{k=1}^{n_1} \Phi_{1,1}^k d_1^k$
$y_{1,2} = y_{1,1} + \sum_{k=1}^{n_2} \Phi_{1,2}^k d_2^k$
$y_{1,j} = y_{1,j-1} + \sum_{k=1}^{n_j} \Phi_{1,j}^k d_j^k$
H_1
H_2
H_3
H_{j-1}
H_j
Sink

Fig. 2. Illustration for computing random projections in the inter-cell protocol.

slots. In the intra-cell protocol, each node needs to transmit the data to the cell head in the same cell. Since there are $l = \Theta(\frac{n}{\log n})$ cells in the network and $n_i = \Theta(\log n)$ nodes in each cell, it is easy to check that $\Theta(n)$ transmissions suffice to complete all the transmissions using the intra-cell protocol. According to the K^2-TDMA scheduling scheme, simultaneous transmissions can occur in different super cells, where each time slot is allocated to each node. It is easy to know that there are $\Theta(K^2 \log n)$ nodes in each super cell. Therefore, it requires $\Theta(K^2 \log n)$ time slots for all the transmissions in this stage. Since K is a constant, the intra-cell protocol can be completed in this stage in $T_1 = \Theta(\log n)$ time slots using $E_1 = \Theta(n)$ transmissions.

Now we consider the computation complexity in the inter-cell protocol. There are $\Theta(\frac{n}{\log n})$ nodes in the spanning tree which only consists of cell heads. For computing one random projection y_i, each cell head only transmits once. Therefore, it requires $\Theta(\frac{n}{\log n})$ transmissions for the sink to compute one random projection. Hence, to compute m random projections, $\Theta(\frac{mn}{\log n})$ transmissions in total are needed. Now we consider the cell scheduling for the inter-cell protocol. The scheduling starts from the bottom of the tree since the nodes at a level can not be scheduled before all the children at this level are scheduled. Note that each cell head has a constant number of children nodes and the maximum time required to schedule a cell head in one level of the tree is K^2 in this stage. The depth of the tree is $\Theta(\sqrt{\frac{n}{\log n}})$. Hence, the inter-cell protocol requires $\Theta(\sqrt{\frac{n}{\log n}})$ time slots to compute one random projection. Therefore, to compute m random projections, $\Theta(m\sqrt{\frac{n}{\log n}})$ time slots are needed. When $m = \Theta(\log n)$, the inter-cell protocol requires $E_2 = \Theta(n)$ transmissions and $T_2 = \Theta(\sqrt{n \log n})$ time slots.

Summarizing the above analysis, we can conclude that the proposed protocol requires $E = E_1 + E_2 = \Theta(n)$ transmissions and $T = T_1 + T_2 = \Theta(\sqrt{n \log n})$ time slots. Therefore, we have the following theorem:

Theorem 1: In a random geometric network, the multi-round random linear function for compressive sensing can be computed with $\Theta(n)$ transmissions and $\Theta(\sqrt{n \log n})$ time slots.

C. Discussion

To compare with the above result, we consider the traditional transmission approach, where the nodes collect the data and forward it to the sink without any computations being performed at the relay nodes. The data is directly transported to the sink via multihop transmissions through the shortest path routing strategy. The computation complexity of this approach has been analyzed in [4] for computing the max function. It is shown that it requires $\Theta(n\sqrt{n/\log n})$ transmissions and $\Theta(n)$ time slots. Therefore, our approach can significantly save the energy and reduce the latency for data gathering by a factor of $\Theta(\sqrt{n/\log n})$, respectively.

IV. Gossip-Based Computation Protocol with Compressive Sensing

In the above section, we have presented a tree-based protocol for computing multiround random linear function. However, the tree-based protocol is susceptible to the failure of nodes or links. In this section, we propose a gossip-based computation protocol and analyze the performance of the protocol in terms of energy consumption and latency.

A. Protocol

The proposed protocol combines broadcast gossip algorithm with cell scheduling. The cell scheduling described in Section III is adopted in the protocol. We first describe how the tth random projection y_t is computed and spread to each node. The protocol operates as follows: Firstly each node obtains its measured value x_i at a sampling instance. At each time slot, one cell in a super cell actives and one node in the cell is randomly selected as a cell head. The cell head broadcasts a message within distance $r(n)$ from it, where $r(n)$ is the transmission range with $r(n) = \sqrt{16 \log n/n}$. Once neighboring nodes receive the message, a group is formed with the cell head as group head and neighboring nodes as group members. Then, the neighboring nodes compute $\omega_i = n\Phi_{i,t}x_i$, and transmit the results to the group head. The group head collects all the values from these neighboring nodes, computes the average value and broadcasts it to the neighboring nodes. The neighboring nodes receive the average value and update their values with it. When the computation results are within some desired accuracy range, the gossip algorithm stops and continues to compute the next random projection.

B. Analysis

In this section, we analyze the performance of gossip-based protocol in terms of energy consumption and latency. The analysis of our gossip-based protocol is based on the random grouping algorithm [9]. However, we consider the transmission scheduling and compute multiround random linear function in our paper, which makes our analysis different. Before proceeding our analysis, we present some preliminaries.

Definition 1: Consider a connected undirected graph $G(V, E)$ with n nodes. Let $\mathbf{x} = (x_1, \cdots, x_n)$ denote the measured value vector of n nodes, where x_i is the measured value of node i. To begin an instance of gossip, each node i initializes the value $\omega_i = n\Phi_{i,t}x_i$, where $\Phi_{i,t}$ are i.i.d random variables which take the values of $\pm 1/\sqrt{n}$ with probability $1/2$. The *potential* of the graph G is defined as

$$\phi = \sum_{i=1}^{n} (\omega_i - \bar{\omega})^2 = \sum_{i=1}^{n} {\omega_i}^2 - n\bar{\omega}^2, \tag{3}$$

where $\bar{\omega} = \sum_i \omega_i / n$ is the average value on a node. Note that $\phi = 0$ if and only if $\boldsymbol{\omega} = (\bar{\omega}, \cdots, \bar{\omega})$.

Definition 2: *Convergence Rate*. Let ϕ and ϕ' denote the potential before and after the invocation of the algorithm, respectively. Let $\delta\phi$ denote the decrement of the potential $\phi - \phi'$. The convergence rate is defined as $\delta\phi/\phi$.

Let $\delta\varphi_i$ denote the potential decrement of the group g_i after executing one iteration of the algorithm

$$\delta\varphi_i = (\sum_{j \in g_i} {\omega_j}^2) - \frac{(\sum_{j \in g_i} {\omega_j}^2)}{J} = \frac{1}{J} \sum_{j,k \in g_i} (\omega_j - \omega_k)^2, \quad (4)$$

where J is the number of nodes in the group g_i.

Furthermore, we introduce some linear algebraic concepts in this paper which are used in our analysis. Let A denote the adjacency matrix of G and D denote the diagonal matrix $(d_{i,i})$ where $d_{i,i}$ is the degree of node i. The matrix $L = D - A$ is the Laplacian Matrix of G. The eigenvalues of L are $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$. The eigenvalue λ_2 is the algebraic connectivity of G.

As mentioned before, the K^2-TDMA scheduling scheme is adopted in our algorithm. For each time slot, the cells with the same color are active and a node in each active cell is randomly selected as a cell head. The number of simultaneously active group is $\frac{n}{2K^2 \log n}$. Therefore, the probability that the node i is selected as a cell head to form a group g_i is $P_i = \frac{1}{2K^2 \log n}$.

Lemma 1: The convergence rate

$$E(\frac{\delta\phi}{\phi}) \geq \frac{\lambda_2}{2K^2 d \log n} \quad (5)$$

where d is the maximum degree of the graph G.

Proof:

$$\begin{aligned} E(\delta\phi) &= \sum_{i \in V} Pr(i \in g_i) \times (\delta\varphi_i) \\ &= \sum_{i \in V} P_i \times \frac{1}{d_{i,i} + 1} \sum_{j,k \in g_i} (\omega_j - \omega_k)^2 \\ &\geq \sum_{i \in V} \frac{1}{2K^2 d \log n} \times \sum_{j,k \in g_i} (\omega_j - \omega_k)^2, \end{aligned} \quad (6)$$

where we use $d_{i,i} + 1 \approx d_{i,i} \leq d$. Note that $\phi = \sum_{i \in V} (\omega_i - \bar{\omega})^2$. Therefore,

$$\begin{aligned} E(\frac{\delta\phi}{\phi}) &\geq \frac{1}{2K^2 d \log n} \frac{\sum_{j,k \in V} (\omega_j - \omega_k)^2}{\sum_{i \in V} (\omega_i - \bar{\omega})^2} \\ &= \frac{1}{2K^2 d \log n} \left(\frac{\sum_{j,k \in V} ((\omega_j - \bar{\omega}) - (\omega_k - \bar{\omega}))^2}{\sum_{i \in V} (\omega_i - \bar{\omega})^2} \right). \end{aligned} \quad (7)$$

Let $z_i = \omega_i - \bar{\omega}$ and $\mathbf{z} = (z_1, \cdots, z_n)^T$. Hence,

$$\begin{aligned} E(\frac{\delta\phi}{\phi}) &\geq \frac{1}{2K^2 d \log n} \left(\frac{\sum_{j,k \in V} (z_j - z_k)^2}{\sum_{i=1}^n {z_i}^2} \mid \sum_{i=1}^n z_i = 0, \mathbf{z} \neq 0 \right) \\ &\geq \frac{1}{2K^2 d \log n} \left(\frac{\mathbf{z}^T L \mathbf{z}}{\mathbf{z}^T \mathbf{z}} \mid \sum_{i=1}^n z_i = 0, \mathbf{z} \neq 0 \right). \end{aligned} \quad (8)$$

Since $\sum_{i=1}^n z_i = 0$, $\mathbf{z}$ is orthogonal to the eigenvector $\mathbf{u} = (1, \cdots, 1)$ of the matrix L, which corresponds to the eigenvalue λ_1. Then, using the Courant-Fischer Minimax Theorem [6]

$$\lambda_2 = \min_{\mathbf{z}} \left(\frac{\mathbf{z}^T L \mathbf{z}}{\mathbf{z}^T \mathbf{z}} \mid \mathbf{z} \perp \mathbf{u}, \mathbf{z} \neq 0 \right), \quad (9)$$

it follows that

$$E(\frac{\delta\phi}{\phi}) \geq \frac{\lambda_2}{2K^2 d \log n}. \quad (10)$$

For convenience, we assume that

$$\gamma = \frac{\lambda_2}{2K^2 d \log n}. \quad (11)$$

■

Lemma 2: Let $m_1, m_2, \ldots, m_k$ be the independent random variables representing the simultaneous group distributions after the invocation of the algorithm at iteration $1, 2, \ldots, k$. Let $\phi_1, \phi_2, \ldots, \phi_k$ be the random variables representing the potentials after the invocation of the algorithm at iteration $1, 2, \ldots, k$. Let $E_{m_k}(\phi_k)$ be the expected value of ϕ_k computed over all possible group distributions at iteration k given the potential ϕ_{k-1} at the previous iteration $k-1$. Let $E(\phi_k)$ be the expected value of ϕ_k computed over all possible group distribution to $m_1, \ldots, m_k$, given the initial potential ϕ_0. We have $E(\phi_k) \leq (1-\gamma)^k \phi_0$.

Proof: From Lemma 1, $E_{m_k}(\phi_k) \leq (1-\gamma)\phi_{k-1}$.

$$\begin{aligned} E(\phi_k) &= E_{m_1, m_2, \ldots, m_k}(\phi_k) \\ &= E_{m_1}(E_{m_2}(\cdots E_{m_{k-1}}(E_{m_k}(\phi_k)))) \\ &\leq (1-\gamma) E_{m_1}(E_{m_2}(\cdots E_{m_{k-1}}(\phi_{k-1}))) \\ &\vdots \\ &\leq (1-\gamma)^k \phi_0. \end{aligned} \quad (12)$$

■

Let ϕ_k be the potential after the invocation of the algorithm at the iteration k. If $\phi_k \leq \varepsilon^2$, then the algorithm stops. Now we derive the bound of the number of iterations that the algorithm requires before it stops.

By Lemma 2,

$$E(\phi_k) \leq (1-\gamma)^k \phi_0 \leq \varepsilon^2. \quad (13)$$

Taking logarithms on the two right terms and applying the inequality $-\ln(1-\gamma) \geq \gamma$ for $-1 \leq \gamma < 1$, we obtain

$$k \geq \frac{1}{\gamma} \log(\frac{\phi_0}{\varepsilon^2}). \quad (14)$$

Also, $\phi_0 = \sum_{i=1}^n (n\Phi_{i,t} x_i)^2 - n(\frac{1}{n} \sum_{i=1}^n n\Phi_{i,t} x_i)^2 = n \sum_{i=1}^n {x_i}^2 - n(\sum_{i=1}^n \Phi_{i,t} x_i)^2$. By Khintchine's inequality [11], $n(\sum_{i=1}^n \Phi_{i,t} x_i)^2 \leq 2^{3/4} e^{-1} \cdot 2 \sum_{i=1}^n x^2$. Since the signal $\mathbf{x}$ is compressible, $\mathbf{x} = \Psi\boldsymbol{\theta} = \sum_{i=1}^n \theta_i \psi_i$ where θ_i is the coefficients of $\mathbf{x}$ in the basis Ψ. The ith largest transformation coefficient satisfies $|\theta_i| \leq R i^{-1/p}, R > 0, p \in (0, 1]$. By orthonormality, $||\mathbf{x}||_2^2 = ||\boldsymbol{\theta}||_2^2 = \sum_{i=0}^n {x_i}^2 \leq R^2 \sum_{i=1}^n i^{-2/p}$. The summation $\sum_{i=1}^n i^{-2/p}$ is Riemann zeta function which converges to a constant when $0 < p \leq 1$. Thus, $\phi_0 \leq n \sum_{i=0}^n {x_i}^2 = O(n)$. Assuming that the algorithm stops when the potential ϕ reaches at a small constant value ε, so $\varepsilon^2 = O(1)$. By Markov inequality,

$$Pr(\phi_k > \varepsilon^2) < \frac{E(\phi_k)}{\varepsilon^2} \leq \frac{(1-\gamma)^k \phi_0}{\varepsilon^2}. \quad (15)$$

Therefore, we can choose $k = \frac{c}{\gamma} \log(\frac{\phi_0}{\varepsilon^2})$, where $c \geq 2$, such that

$$Pr(\phi_k > \varepsilon^2) < e^{-\log(\frac{\phi_0}{\varepsilon^2})^c} \frac{\phi_0}{\varepsilon^2} = (\frac{\varepsilon^2}{\phi_0})^{c-1} \longrightarrow 0. \quad (16)$$

Thus,

$$Pr(\phi_k \leq \varepsilon^2) \geq 1 - (\frac{\varepsilon^2}{\phi_0})^{c-1} \longrightarrow 1. \tag{17}$$

Therefore, with high probability, the number of iterations requires

$$k = O\left(\frac{c}{\gamma}\log(\frac{\phi_0}{\varepsilon^2})\right). \tag{18}$$

Furthermore, it is shown that λ_2 is bounded by the following function of the diameter $diam(G)$ of the graph [10]:

$$\frac{4}{n \cdot diam(G)} \leq \lambda_2 \leq \frac{8d}{diam(G)^2}\log_2^2 n. \tag{19}$$

For the random geometric graph, to guarantee full connectivity, $d = O(\log n)$. The diameter of the the random geometric graph is $\Theta(\sqrt{n/\log n})$, so the bounds of λ_2 are

$$\Omega(\frac{\sqrt{\log n}}{n^{3/2}}) = \lambda_2 = O(\frac{\log^4 n}{n}). \tag{20}$$

Combining (11), (18) and (20), we can get the following lower bound and upper bound of the computation iterations

$$\Omega(\frac{n}{\log n}) = k = O(n^{3/2}(\log n)^{5/2}). \tag{21}$$

The above result is only for computing one random projection. Furthermore, for each round of algorithm, each cell head needs $\Theta(\log n)$ time slots to collect the data from the neighboring nodes in the group. Hence, to compute m random projections, the total number of time slots needed is $km\log n$. When $m = \Theta(\log n)$, we have the following theorem:

Theorem 2: Given a connected undirected graph $G(V, E)$, the bounds of the total number of time slots T_g needed for computing m random projections in a node within an accuracy $\varepsilon = O(1)$ are

$$T_g = \Omega(n\log n) \tag{22}$$

$$T_g = O(n^{3/2}(\log n)^{9/2}). \tag{23}$$

We now consider the energy consumption by the number of transmissions for computing random projections. We have the following theorem:

Theorem 3: Given a connected undirected graph $G(V, E)$, the total expected energy consumption E_g in terms of number of transmissions needed for computing m random projections in a node within an accuracy $\varepsilon = O(1)$ is

$$E(E_g) = \Omega(n^2) \tag{24}$$

$$E(E_g) = O(n^{5/2}(\log n)^{7/2}). \tag{25}$$

Proof: Since there are $\Theta(\frac{n}{\log n})$ groups simultaneously compute random projections for each iteration, the total expected number of transmissions needed for computing m random projections is $E(E_g) = T_g \times \Theta(\frac{n}{\log n})$. By Theorem 2, we obtain the above theorem. ■

C. Discussion

We compare the performance of our gossip-based approach to the performance of the scheme using randomized gossip [5], where a node randomly selects one neighboring node to exchange random measurements and compute the average value for each iteration of computation. In [5], it has been shown that the number of transmissions needed to compute m random projections is $\Theta(mn^2) = \Theta(n^2\log n)$ within an accuracy $\varepsilon = O(1)$ when $m = \Theta(\log n)$. In our work, we show that our broadcast gossip-based approach requires $\Omega(n^2)$ transmissions, which indicates that our approach can improve upon the randomized gossip-based approach.

V. Conclusion and Future Work

In this paper, we have studied the application of compressive sensing for data gathering from the perspective of in-network computation in wireless sensor networks. we formulated the problem of data gathering to compute multiround random linear function. We designed two protocols for computing such a function in centralized and distributed fashions, respectively. For these two computation protocols, we studied the performance of in-network computation in terms of energy consumption and latency. For future work it is interesting to study delay and capacity tradeoff for data gathering with compressive sensing in mobile sensor networks as [12].

VI. Acknowledgment

This work is supported by National Fundamental research grant (2011CB302701), NSF China (No. 60832005, 60972032); China Ministry of Education New Century Excellent Talent (No. NCET-10-0580); China Ministry of Education Fok Ying Tung Fund (No. 122002); Qualcomm Research Grant; Shanghai Basic Research Key Project (No. 11JC1405100); National Key Project of China (2010ZX03003-001-01) and the National “863” Hi-tech Project of China.

References

[1] Y. Liu, K. Liu, and M. Li, “Passive Diagnosis for Wireless Sensor Networks,” *IEEE/ACM Transactions on Networking* , vol. 18, No. 4, pp. 1132-1144, Aug. 2010.

[2] D. Donoho, “Compressed sensing,” *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289-1306, Apr. 2006.

[3] A. Giridhar and P. R. Kumar, “Computing and communicating functions over sensor networks,” *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 455-764, Apr. 2005.

[4] N. Khude, A. Kumar, and A. Karnik, “Time and energy complexity of distributed computation in wireless sensor networks,” *IEEE Trans. Mobile Computing*, vol. 7, no. 5, pp. 617-632, May. 2008.

[5] M. Rabbat, J. Haupt, A. Singh, and R. Nowak, “Decentralized compression and predistribution via random gossiping,” in *IPSN*, pp. 51-59, Apr. 2006.

[6] B.Mohar and S. Poljak, “Eigenvalues in combinatorial optimization,” in R. A. Brualdi, S. Friedland, and V. Klee, editors, *Combinatorial and Graph-Theoretical Problems in Linear Algebra, volume 50 of IMA Volumes in Mathematics and its Applications*, pp. 107-151, Berlin, Springer-Verlag, 1993.

[7] P. Gupta and P. R. Kumar. “The capacity of wireless network,”*IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388-404, Mar. 2000.

[8] F. Xue and P. R. Kumar, “The Number of Neighbors Needed for Connectivity of Wireless Networks,” *Wireless Networks*, vol. 10, no. 2, pp. 169-181, Mar. 2004.

[9] J. Y. Chen, G. Pandurangan, and D. Xu, “Robust Computation of Aggregates in Wireless Sensor Networks: Distributed Randomized Algorithms and Analysis,” *IEEE Trans. Parallel and Distributed Systems*, vol. 17, no. 9, pp. 987-1000, Sept. 2006.

[10] T. Aysal, M. Yildiz, A. Sarwate, and A. Scaglione, “Broadcast gossip algorithms for consensus,” *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2748-2761, Jul. 2009.

[11] H. Rauhut, “Compressive sensing and structured random matrices,” in *Theoretical Foundations and Numerical Methods for Sparse Recovery*, edited by Fornasier, M. Radon Series Comp. Appl. Math., vol. 9, pp. 1-92. deGruyter, 2010.

[12] X. Wang, W. Huang, S. Wang, J. Zhang, C. Hu, “Delay and Capacity Tradeoff Analysis for MotionCast,” *IEEE/ACM Transactions on Networking*, vol. 19, no. 5, pp. 1354-1367, Oct. 2011.