Efficient Agile Sink Selection in Wireless Sensor Networks based on Compressed Sensing

Mohammadreza Mahmudimanesh
Technical University of Darmstadt
Email: reza@cs.tu-darmstadt.de

Amir Naseri
Technical University of Darmstadt
Email: amir.naseri@stud.tu-darmstadt.de

Neeraj Suri
Technical University of Darmstadt
Email: suri@cs.tu-darmstadt.de

***Abstract*—Collection of the sensed data in a wireless sensor network at one or more sink(s) is a well studied problem and there are a lot of efficient solutions for a variety of wireless sensor network configurations and application requirements. These methods are often optimized towards collection of the sensed data at a predetermined base station or sink. This inherently reduces the agility of the wireless sensor network as the flow of information is not easily changeable after the establishment of the routing and data collection algorithms. This paper presents an efficient data dissemination method based on the compressed sensing theory that allows each sensor node to take the role of a sink. Agile sink selection is especially advantageous in scenarios where the sink or the end user of the wireless sensor network is mobile. The proposed method allows availing the global state of the environment by fetching a small set of data from any arbitrary node. Our evaluations prove the better performance of our technique over existing methods. Also a comparison with an oracle-based approach gives sufficient experimental evidences of a nearly optimal performance of our method.**

I. Introduction

A Wireless Sensor Network (WSN) is a distributed sensory system consisting of interconnected Sensor Nodes (SN) that is employed for large-scale monitoring of the environment. [1], [2]. In a conventional WSN, the sensed data are gathered at a dedicated node called *sink*. The sink then processes the received data and prepares it for the end user. This paper studies the problem of availing the sensed data at *any* SN. This allows an *agile* sink selection based on the specific requirements of the end user. Agile sink selection gives the end user more flexibility and mobility.

The paradigm of WSN data handling techniques is characterized by three attributes of the SNs:

1) The power supply of a SN is commonly in form of battery, and hence, is a limited source of energy.
2) Data transmission is the main energy drain of a SN, thus reduction in the amount of transmissions is advocated [3], [4].
3) The processing power and the memory of a SN is often very limited.

Given these constraints, we propose a novel data dissemination technique, called *Comprensus*, that efficiently disseminates the sensed data to all SNs. The *global* state of the environment is obtainable by querying a small set of data from *any* SN. Comprensus draws its concepts from the well known Consensus methods. Similar to Consensus, the global state propagates in the network via local information exchange by neighboring nodes. Our work distinguishes itself from Consensus methods by considering the compressibility of the sensed data. Comprensus reduces the amount of information exchanges by applying the theory of Compressed Sensing (CS).

A. Problem statement

Consider a WSN consisting of n static SNs that are labeled from 1 to n. The value recorded by SN i is denoted by f_i. We make a vector $\mathbf{f} \in \mathbb{R}^n$ by stacking the values recorded by the SNs.

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \tag{1}$$

The goal of our dissemination method is to make the vector $\mathbf{f}$ available to all SNs within a certain time limit. The global state can be recovered by fetching a limited amount of data from any SN, e.g., when a mobile sink visits some SNs and estimates the global state of the environment by extracting their data.

There are two challenges to achieve this goal. First, the number of transmissions by the SNs must be minimized in order to meet bandwidth limitations and also save battery. Second, the dissemination protocol must be light-weight such that it can be easily implemented on the basic hardware platforms of the SNs.

B. Contributions

Our approach has two advantages over the straightforward solution that gathers data at a stationary sink and sends it to the mobile end user.

1) Comprensus allows each SN to be a potential sink.
2) There is no need for the end user to be in radio range of the stationary sink.

The mobile end user extracts the global state of the environment from any SN in its vicinity. This is especially useful for in-door applications where the base station is not necessarily accessible from any arbitrary location in the environment.

Our evaluations prove better performance of Comprensus in comparison to the state of the art methods such as compression and predistribution via randomized gossiping [5]. We also compare the performance of Comprensus with an oracle-based approach and show that it performs nearly optimal in a noiseless environment. According to the CS theory, in presence of the additive noise, the accuracy of the disseminated data is degraded not much more than the noise magnitude.

II. Background and related work

This section briefly reviews the fundamentals of the CS theory and the terminology that is required for description of our technique. It also reviews a closely related work by Rabbat et al., namely the decentralized compression and predistribution via randomized gossiping [5].

A. Compressed sensing

CS allows a more efficient acquisition of a signal by taking advantage of its compressibility. A signal of size n is denoted by a vector $\mathbf{f} \in \mathbb{R}^n$. Each element of the vector $\mathbf{f}$ corresponds to a *sample* of the signal.

Definition 1. *We call a vector* $\boldsymbol{\phi} \in \mathbb{R}^n$ *a* sensing vector*, and the inner product of a sensing vector and vector* $\mathbf{f}$ *is called a* measurement.

Let $y_1, y_2, \ldots, y_m$ be m measurements such that

$$y_j = \boldsymbol{\phi}_j{}^T \mathbf{f} + z_j \quad , \quad j \in \{1, 2, \ldots, m\} \tag{2}$$

where $\boldsymbol{\phi}_j$ are the sensing vectors, $\{z_j\}$ is the white noise sequence with variance σ^2. This can be also written using matrix notations:

$$\mathbf{y} = \boldsymbol{\Phi}\mathbf{f} + \mathbf{z} \tag{3}$$

where $\mathbf{y} = [y_1 \ y_2 \ \ldots \ y_m]^T$ is the measurement vector, $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1 \ \boldsymbol{\phi}_2 \ \ldots \ \boldsymbol{\phi}_m]^T$ is the measurement matrix and $\mathbf{z} = [z_1 \ z_2 \ \ldots \ z_m]^T$ is the noise vector.

According to the CS theory, it is possible to recover vector $\mathbf{f}$ from $m < n$ measurements under certain conditions for $\mathbf{f}$ and the sensing vectors as follows.

We assume that $\mathbf{f}$ can be sparsely represented under a linear projection using an orthonormal matrix $\boldsymbol{\Psi}$. Suppose that $\mathbf{f} = \boldsymbol{\Psi}\mathbf{x}$ for a suitably chosen orthonormal matrix $\boldsymbol{\Psi}$ such that $\mathbf{x}$ is sparse. Vector $\mathbf{x}$ is called a sparse vector when it has $s \ll n$ non-zero components and all its other $(n-s)$ components are zero. Sparsity plays an important role in the CS theory. The sparser the vector $\mathbf{x}$ is, the fewer measurements are required to recover $\mathbf{f}$. The spatial signals recorded by a WSN admit a nearly sparse representation under an orthonormal linear transform such as Fourier, Discrete Cosine Transform (DCT) or Discrete Wavelet Transform (DWT) [6].

1) Isotropy and Incoherence: *Isotropy* and *incoherence* are the other necessary conditions in order to recover $\mathbf{f}$ from $\mathbf{y}$ [7]. Let $\boldsymbol{\phi} \in \mathbb{R}^n$ be a random sensing vector with independent and identically distributed components drawn from distribution F, i.e., $\boldsymbol{\phi} \overset{iid}{\sim} F$.

Definition 2. [7] *Distribution* F *has the* isotropy property*, when* $\boldsymbol{\phi}\boldsymbol{\phi}^T$ *is expected to be the identity matrix. Mathematically,*

$$E[\boldsymbol{\phi}\boldsymbol{\phi}^T] = \mathbf{I} \quad , \quad \boldsymbol{\phi} \sim F. \tag{4}$$

The isotropy condition can be weakened to *near isotropy*, i.e., $E[\boldsymbol{\phi}\boldsymbol{\phi}^T] \approx \mathbf{I}$ and still $\mathbf{f}$ is accurately recoverable from the measurement vector $\mathbf{y}$ [7].

Definition 3. [7] Coherence parameter μ *is defined as the smallest value* μ *such that*

$$|\boldsymbol{\phi}_j^T \boldsymbol{\psi}_i|^2 \ \leq \ \mu \tag{5}$$

for all sensing vectors $\boldsymbol{\phi}_j$ *and columns* $\boldsymbol{\psi}_i$ *of matrix* $\boldsymbol{\Psi}$*,* $j \in \{1, 2, \ldots, m\}$ *and* $i \in \{1, 2, \ldots, n\}$.

We say that the sensing vectors are more *incoherent* if the value of μ is a smaller number. According to [7], the more incoherent the sensing vectors are, the less random measurements are required for accurate recovery. Candes et al. in their RIPless theory of CS [7] discuss some of the random distributions F obeying the isotropy condition. These include the Gaussian distribution, Rademacher distribution and random Fourier sampling [7]. Randomized sampling brings a key benefit for WSNs by eliminating the need for centralized coordination [8], [9].

It is shown in [7] that if the isotropy condition holds and the number of measurements m is in the order of $O(\mu s \log n)$, then $\mathbf{f}$ can be recovered from $\mathbf{y}$ with an overwhelming probability. Therefore, we need a basis[1] Ψ and a set of sensing vectors with isotropy property such that $\mathbf{f}$ is compressible under the Ψ-transform and the columns of the transformation matrix $\boldsymbol{\Psi}$ have the least coherence with the sensing vectors. In this paper, we propose a novel network coding technique that fulfills these conditions.

2) Signal recovery: In order to recover $\mathbf{f}$ from $\mathbf{y}$, first we need to solve the following convex optimization problem [7].

$$\underset{\tilde{\mathbf{x}} \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\boldsymbol{\Phi}\boldsymbol{\Psi}\tilde{\mathbf{x}} - \mathbf{y}\|_2^2 \ + \ \lambda\sigma\|\tilde{\mathbf{x}}\|_1 \tag{6}$$

where $\lambda = 10\sqrt{m \log n}$, $\|\cdot\|_1$ is the norm-1 operator and $\|\cdot\|_2$ is the norm-2 operator[2].

Some of the efficient and accurate algorithms for solving this problem can be found in [10] and [11]. If $\hat{\mathbf{x}}$ is the solution to the convex optimization problem in Equation 6, then $\hat{\mathbf{f}} = \boldsymbol{\Psi}\hat{\mathbf{x}}$ will estimate the original signal $\mathbf{f}$ with an error bounded by $\text{polylog}(n)(s/m)\sigma^2$ [7]. In practice, signal $\mathbf{f}$ is not strictly sparse under the Ψ-transform. Instead, $\mathbf{x}$ has a few components with larger magnitudes and most of its remaining components are nearly zero. Suppose that $\mathbf{x}_s$ is a sparse approximation of $\mathbf{x}$ by keeping s most significant components of $\mathbf{x}$ and zeroing its remaining $(n-s)$ components. It is shown in [7] and [12] that, in such a case the recovery error will not grow much more than $O(\|\mathbf{x} - \mathbf{x}_s\|_2)$.

B. Distributed compression and predistribution via randomized gossiping

Randomized gossiping is a data dissemination approach which employs average consensus algorithm [5]. It makes use of gossiping average consensus technique to compute and

[1] A basis for $\mathbb{R}^n$ is a set of vectors $\boldsymbol{\psi}_i, i \in \{1, \ldots, n\}$, such that any vector $\mathbf{f} \in \mathbb{R}^n$ can be represented as $\mathbf{f} = \sum_{i=1}^n x_i \boldsymbol{\psi}_i$ where x_i are called the coefficients of $\mathbf{f}$ in basis Ψ. The corresponding transformation matrix $\boldsymbol{\Psi}$ is made by putting the vectors $\boldsymbol{\psi}_i$ in a matrix, i.e., $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1 \ \boldsymbol{\psi}_2 \ \ldots \ \boldsymbol{\psi}_n]$ and $\mathbf{f} = \boldsymbol{\Psi}\mathbf{x}$ where $\mathbf{x} = [x_1 \ x_2 \ \ldots \ x_n]^T$.

[2] For a real vector $\mathbf{v} \in \mathbb{R}^n$, norm-1 of $\mathbf{v}$ is defined as $\|\mathbf{v}\|_1 = \sum_{i=1}^n |v_i|$ and norm-2 of $\mathbf{v}$ is defined as $\|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^n |v_i|^2}$

distribute random projections of the sensed data. These random projections are the measurements y_j where $j \in \{1, 2, ..., m\}$.

Let $\boldsymbol{\alpha}_i$ refer to the i^{th} column of measurement matrix $\boldsymbol{\Phi}$. Note that $\boldsymbol{\alpha}_i$ is generated in a decentralized manner by employing a pseudo-random number generator. Suppose that SN i computes $f_i\boldsymbol{\alpha}_i$ and stores the result in an array of m real numbers. Let $\mathbf{w}_i[t] \in \mathbb{R}^m$ refer to the content of the array inside SN i at time t. SN l_1 is activated uniformly at random at time t and chooses one of its neighbors l_2 uniformly at random. SN l_1 and SN l_2 exchange $\mathbf{w}_{l_1}[t]$ and $\mathbf{w}_{l_2}[t]$ and update $\mathbf{w}_{l_1}[t+1] = \mathbf{w}_{l_2}[t+1] = (\mathbf{w}_{l_1}[t] + \mathbf{w}_{l_2}[t])/2$.

According to the work by Rabbat et al. [5], when $t \to \infty$, $\mathbf{w}_i \to \boldsymbol{\Phi}\mathbf{f}$ for all $i \in \{1, 2, \ldots, n\}$ [5]. Therefore, after *sufficiently many* iterations of this protocol, the content of array $\mathbf{w}_i$ in all SNs will *get close enough* to $\mathbf{y} = \boldsymbol{\Phi}\mathbf{f}$. According to the discussion in Section II-A2, signal $\mathbf{f}$ can be recovered at any SN after solving Equation 6. Since the size of $\mathbf{y}$ and also the array $\mathbf{w}_i$ is in the order of $\mu s \log n$, randomized gossiping requires $O(\mu s \log n)$ transmissions per iteration. The number of required iterations depends on the network topology [5].

In practice, achieving a negligible recovery error requires a large number of iterations. This is one important drawback of randomized gossiping. Comprensus proves to be more efficient in terms of number of iterations and transmissions required for data dissemination. The novelty of the Comprensus is its efficient network coding technique, which makes a faster convergence to the minimum recovery error possible.

This paper proposes a novel network coding mechanism which is still as simple as consensus, nevertheless, requires less time and communications to disseminate the measurements. In randomized gossiping [5], the SNs run a protocol such that all of them *converge* to a measurement vector $\mathbf{y}$ which is common among all SNs.

In Comprensus, each SN receives a different measurement vector $\mathbf{y}_i$, $i \in \{1, 2, \ldots, n\}$. We show that for all of these measurement vectors, the isotropy and incoherence properties hold. Therefore, Comprensus does not need too many iterations for convergence to the same measurement vector among all SNs. Instead, our proposed method guarantees that the same signal $\mathbf{f}$ is accurately recoverable from each individual measurement vector $\mathbf{y}_i$ received by SN i, $i \in \{1, 2, \ldots, n\}$.

III. The Comprensus protocol

In this section, we explain Comprensus, a simple distributed protocol to disseminate random linear measurements in a WSN with static topology. We assume that the network topology corresponds to a connected regular graph of degree d. It is easy to create a regular graph topology in a WSN when $n \times d$ is an even number. For a given degree d each SN selects at most d neighbors with the highest Received Signal Strength Indicator (RSSI) [13] assuming that each SN has at least d SNs in its communication range. At the end of this process, we will have a topology corresponding to a regular graph of degree d.

We start by defining the variables and definitions used in our protocol. Suppose that each SN is equipped with two pseudo-random number generators as defined below.

- *Rademacher random generator* produces either $+1$ or -1 each with probability $1/2$.
- *Bernoulli random generator* produces 1 with probability $p = k/n$ and 0 with probability $1 - p$.

We assume that SN i keeps a real number u_i in its internal memory. SN i also keeps a list L_i of real numbers in its memory. The data type of the elements of L_i is the same as the data type of u_i. Memory requirement for this list is $O(\mu s \log n)$ items. We will see shortly that μ will be a small constant. This list actually holds the random linear measurements which are used thereafter for signal recovery. One can have an estimation of s in an appropriate basis Ψ based on a previous knowledge about the data gathered from the WSN. Since this estimation is not necessarily accurate, it is recommended to use a worst case estimation for s in a real-world deployment of Comprensus.

A. Distributed Comprensus algorithm

The Comprensus protocol is executed in three phases: *Initialization*, *Dissemination* and *Recovery*. The instructions described below will be executed in parallel by every SN i, $i \in \{1, 2, \ldots, n\}$.

1) Initialization: First, the list L_i is emptied. Then, SN i reads the value f_i from its sensor and stores it into variable u_i. We assume that each SN is given a unique id and initializes the seeds of the Rademacher and Bernoulli random generators by its id. By choosing an efficient and reliable pseudo-random number generator we minimize the chance that two SNs generate the same sequence of random values [14].

2) Dissemination: This phase is repeated r times in parallel by all n SNs. At each iteration $t \in \{1, 2, \ldots, r\}$ all of the SNs execute Algorithm 1 simultaneously.

- $h_i[t]$ is the value generated by the Rademacher random generator of SN i at iteration t.
- $b_i[t]$ is the value generated by the Bernoulli random generator of SN i at iteration t.

Algorithm 1 Dissemination phase of Comprensus

$u_i \leftarrow h_i[t] \cdot u_i$
if $b_i[t] = 1$ **then**
  Transmit u_i
else
  for all SN j in neighborhood of SN i **do**
    if SN j is transmitting the value $u_j[t]$ **then**
      $u_i \leftarrow u_i + u_j[t]/n$
    end if
  end for
end if
if at least one neighbor has transmitted **then**
  add u_i to the rear of L_i
end if

Remark: We assume that the SNs cannot transmit and listen at the same time. Also the wireless channel of two adjacent nodes cannot be used simultaneously. Therefore, if a set of adjacent SNs want to transmit at the same iteration, they transmit one by one according to the descending order of their ids. They aggregate their received measurements in a temporary variable and update their corresponding u_i only after all of these concurrent transmissions are completed.

If the SNs are perfectly synchronized, aggregation by signal superposition helps to perform the dissemination phase faster. Signal superposition allows multiple nodes to transmit simultaneously and the receiver accumulates the received values at the same time [15], [16].

3) Recovery: SN i derives a vector $\mathbf{y}_i$ by stacking the entries in list L_i. When all SNs agree on a common random generator algorithm, the linear combinations that led to the values in L_i are reproducible as described in Section II-A. These linear measurements are then placed in Equation 6 to recover $\mathbf{f}$. We show in Section III-C that the linear measurements acquired in the dissemination phase obey the isotropy condition and have low coherence with DCT.

Line 1 of Algorithm 1 generates a new Rademacher value and multiplies it by the current value of u_i which is first set to f_i in the initialization phase. Line 2 decides whether SN i is to transmit in this iteration or not. Since $b_i[t]$ returns 1 with probability k/n, this is equivalent to the case that almost k out of n SNs select themselves uniformly at random to transmit. Executing the line 3 consumes the most amount of battery power, as using the radio in transmitting mode is the major energy drain of a SN [17]. If SN i is not in transmitting mode at iteration t, i.e., $b_i[t] = 0$, then it listens to the communication channel and accumulates the values sent by neighboring nodes onto u_i after dividing them by n as instructed in lines 5 through 9. Summing the received values from neighboring nodes can be done arithmetically by using a simple Time Division Multiple Access (TDMA) mechanism [18]. A faster alternative is signal superposition when the SNs are perfectly synchronized [15], [16]. It can also happen that no neighbor of SN i does a transmission at iteration t. In this case, no value is added to the list L_i. This condition is checked in line 11, and thus, line 12 is executed only when at least one neighboring node has transmitted. We will explain shortly why this restriction is necessary.

B. Matrix representation of the distributed protocol

In this section we examine the network-wide implication of Algorithm 1 by using the equivalent matrix representation of Comprensus.

Let N_i denote the set of the d neighbors of SN i.

Definition 4. *Transition matrix $\mathbf{M}_t$ at iteration t is an $n \times n$ real matrix with the following attributes.*

1) $M_t[i,i] = h_i[t]$ for $1 \le i \le n$.

2) $M_t[j,i] = h_i[t]/n$ when $j \in N_i$ and $b_i[t] = 1$.

It is easy to verify that after iteration t of the dissemination phase,

$$\begin{pmatrix} u_1[t] \\ u_2[t] \\ \vdots \\ u_n[t] \end{pmatrix} = \left(\mathbf{M}_t \times \mathbf{M}_{t-1} \times \cdots \times \mathbf{M}_1\right)\mathbf{f} \tag{7}$$

describes the contents of variables u_i , $i \in \{1, 2, \ldots, n\}$. We address the effect of noise at the end of our matrix analysis. We also define the $n \times n$ matrix $\mathbf{Q}_t$ as

$$\mathbf{Q}_t := \begin{pmatrix} \mathbf{q}_{1,t} \\ \mathbf{q}_{2,t} \\ \vdots \\ \mathbf{q}_{n,t} \end{pmatrix} := \mathbf{M}_t \times \mathbf{M}_{t-1} \times \cdots \times \mathbf{M}_1 \tag{8}$$

where $\mathbf{q}_{1,t},\ \mathbf{q}_{2,t}\ ,\ \ldots\ ,\ \mathbf{q}_{n,t}$ are the rows of matrix $\mathbf{Q}_t$.

We define a set R_i as

$$R_i := \{t \quad | \quad \exists j \in N_i \ : b_j[t] = 1\} \tag{9}$$

to refer to the set of iterations in which at least one neighboring node of SN i is transmitting. We also define matrix $\boldsymbol{\mathcal{A}}_i$ as

$$\boldsymbol{\mathcal{A}}_i := [\mathbf{q}_{i,t_1}^T \ \ \mathbf{q}_{i,t_2}^T \ \ \ldots \ \ \mathbf{q}_{i,t_{m(i)}}^T]^T \tag{10}$$

where $m(i) = |R_i|$ is the number of measurements received by SN i and $\{t_1, t_2, \ldots, t_{m(i)}\} = R_i$. The number of received measurements may differ from one SN to other. Nevertheless, when the network topology corresponds to a regular graph, all of the nodes are expected to receive almost the same amount of measurements, since each SN has an equal chance to transmit and receive messages. It can be shown that the measurement vector $\mathbf{y}^{(i)}$ made by stacking the values in list L_i will be

$$\mathbf{y}^{(i)} = \boldsymbol{\mathcal{A}}_i \mathbf{f} + \mathbf{z} \tag{11}$$

where $\mathbf{z}$ is the additive noise. The noise is added either by the communication channel or can be regarded as a side effect of low precision floating pointing storage and processing inside the SNs. We model $\mathbf{z}$ by a white Gaussian noise vector in our simulations and experiments.

If the rows of $\boldsymbol{\mathcal{A}}_i$ obey the isotropy property and have low coherence with a compressive basis, then $\mathbf{f}$ can be recovered at SN i from $\mathbf{y}^{(i)}$ with high probability as detailed in Section II-A2. Now the reason for the restriction in Line 11 of Algorithm 1 becomes clear. We only let *newly received* measurements to be aggregated and added to the measurement list L_i. Otherwise, $\boldsymbol{\mathcal{A}}_i$ will have at least two rows which are linearly dependent, and thus, $\boldsymbol{\mathcal{A}}_i$ is not full rank. In other words, we will have redundant measurements stored in L_i if we do not check the condition in Line 11 of Algorithm 1.

Suppose that $\bar{m}$ is the average number of measurements received per SN. $\bar{m}$ should be in order of $O(\mu s \log n)$ to allow successful recovery. When these conditions are fulfilled, the signal vector $\mathbf{f}$ can be recovered at every SN after running the Comprensus protocol. Next, we examine isotropy and incoherence properties of our measurement matrix $\boldsymbol{\mathcal{A}}_i$ for $i \in \{1, 2, \ldots, n\}$.

C. Numerical experiments

In this section, we investigate the isotropy and incoherence of our measurement method through numerical experiments on simulated WSNs. We perform comprehensive numerical experiments on simulated WSNs consisting of $n = 128$ SNs. The network graph is a random regular graph of degree $d = 5$ which is freshly generated in each experiment and the results are averaged over multiple simulation runs.

We let the SNs to generate their corresponding h_i and b_i random numbers and execute the Comprensus protocol for varying values of r and k. Each experiment is run several times and all of the results are averaged to eliminate randomness effects.

In Section II-A1, we have seen that even if the set of sensing vectors have the *near-isotropy* property, the signal $\mathbf{f}$ can be recovered from measurement vector $\mathbf{y}$. In Comprensus, the set of the sensing vectors for SN i are the rows of $\boldsymbol{\mathcal{A}}_i$ and the measurement vector for SN i is $\mathbf{y}^{(i)}$. We define a metric for *deviation from isotropy* and show that the rows of $\boldsymbol{\mathcal{A}}_i$ have a very low deviation from the isotropy property.

Definition 5. Deviation from isotropy *for a random sensing vector* $\mathbf{a}$ *is defined as* $\sum_{e \in \mathcal{E}_a} (1-e)^2$ *where* $\mathcal{E}_a$ *is the the set of eigenvalues of the square matrix* $E[\mathbf{a}\mathbf{a}^T]$.

This metric determines how much $E[\mathbf{a}\mathbf{a}^T]$ behaves like an identity matrix. In the ideal case, $E[\mathbf{a}\mathbf{a}^T] = \mathbf{I}$ and has only one eigenvalue, i.e., 1, and thus, the deviation from isotropy is zero. For random sensing vector $\mathbf{a}$, if the eigenvalues of $E[\mathbf{a}\mathbf{a}^T]$ are all very close to 1, $E[\mathbf{a}\mathbf{a}^T]$ behaves like an identity matrix and deviation from isotropy as defined in Definition 5 will be low.

In our numerical experiments, a large set of measurement matrices $\boldsymbol{\mathcal{A}}_i$ are generated. Our random sensing vectors are actually the rows of the randomly generated measurement matrices $\boldsymbol{\mathcal{A}}_i$. For each row $\mathbf{a}^T$ of these matrices, we calculate $\mathbf{a}\mathbf{a}^T$ and sum up all of the results. $E[\mathbf{a}\mathbf{a}^T]$ is then numerically calculated by dividing this summation by the total number of the randomly generated sensing vectors.

Observation 1 – near-isotropy: Deviation from isotropy according to Definition 5 is calculated over all randomly generated measurement matrices $\boldsymbol{\mathcal{A}}_i$, $i \in \{1, 2, \ldots, n\}$ and a full range of experiments with k varying from 1 to n and $r \in \{32, 64, 128\}$. The results as illustrated in Figure 1.a prove that our measurement mechanism obeys the near-isotropy property with negligible deviation, i.e., $E[\mathbf{a}\mathbf{a}^T] \approx \mathbf{I}$.

Observation 2 – low coherence: We set $\boldsymbol{\Psi}$ to be the inverse DCT matrix and calculate the coherence with the DCT basis according to Definition 3 for randomly generated measurement matrices with k varying from 1 to n and $r \in \{32, 64, 128\}$. The averaged results over a large set simulations as illustrated in Figure 1.b shows that the coherence factor is also low.

Next, we evaluate the performance of Comprensus and compare it with randomized gossiping and an oracle-based approach.

IV. Evaluation

We simulate the Comprensus protocol by distributed execution of Algorithm 1 for different values of r and k on a large set of synthesized spatial signals. After running each instance of the Comprensus protocol, the Signal to Noise Ratio (SNR) of the recovered signal is calculated. The SNR is measured in decibels (dB) and the recovery algorithm is run at a randomly chosen SN. This guarantees that the recovery is possible at any arbitrarily chosen node.

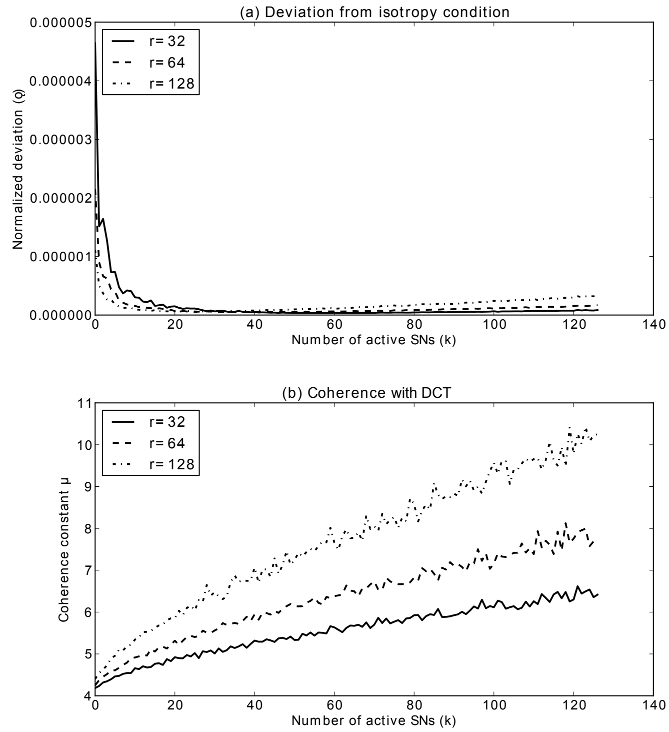


Fig. 1. Near-isotropy and low coherence of Comprensus measurement scheme

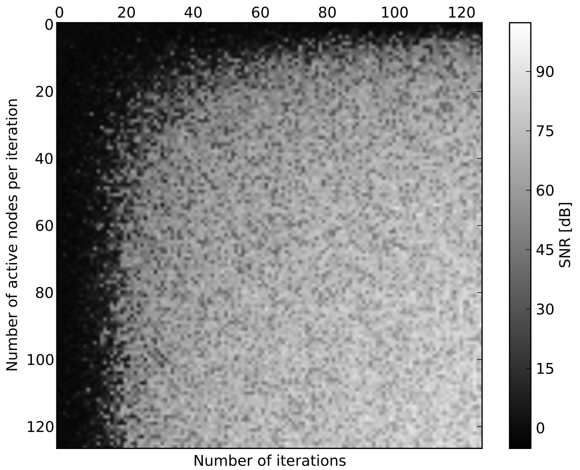
If $\mathbf{f}$ is the original signal and $\hat{\mathbf{f}}$ is the recovered signal, we define SNR as

$$\text{SNR} := 10 \log_{10} \left(\frac{\|\mathbf{f}\|_2^2}{\|\mathbf{f} - \hat{\mathbf{f}}\|_2^2} \right). \tag{12}$$

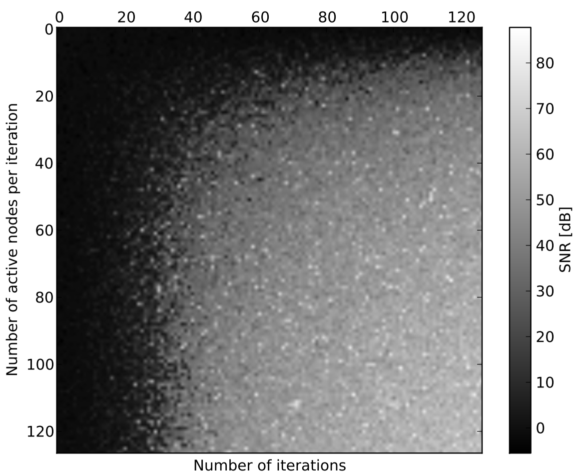
Figure 2 shows the results of simulating a network consisting of 128 nodes. Number of iterations and number of active nodes are varying from 1 to 128. Each point on the SNR diagrams corresponds to one simulation run. Brighter area shows higher SNR which means accurate signal recovery at an arbitrary SN and darker area shows lower SNR which means that signal recovery is not possible, and hence, the sensed data are not accessible from an arbitrary node.

An interesting observation in our evaluations is that the transition to the condition where accurate recovery is possible is relatively sharp. Looking at the SNR diagrams of Figure 2, the border between the bright area (successful dissemination) and dark area (non-recoverability) has a recognizable contrast. Sharp transition between recoverability and non-recoverability states in CS is comprehensively studied in the Donoho-Tanner universal phase transition inspections [19].

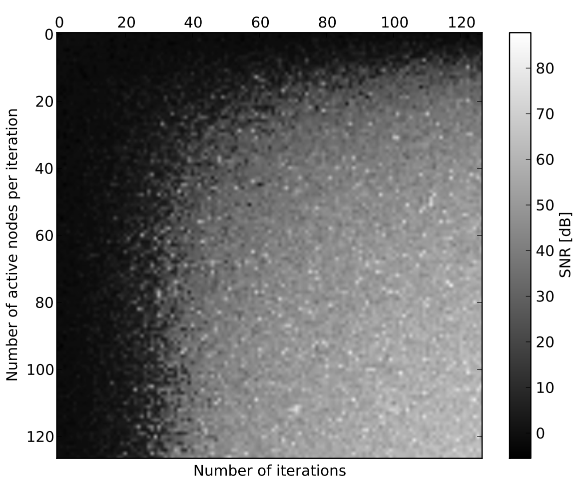
We conduct a large set of simulations with different configurations of the Comprensus protocol and for different values of the sparsity parameter s. Next, we compare Comprensus to randomized gossiping [5] for three illustrative values for s. Our evaluations show that Comprensus outperforms randomized gossiping in terms of message cost and the dissemination time.



(a) Signal recovery for $s = 5$



(b) Signal recovery for $s = 10$



(c) Signal recovery for $s = 20$

Fig. 2. Accuracy of signal recovery for different sparsity levels

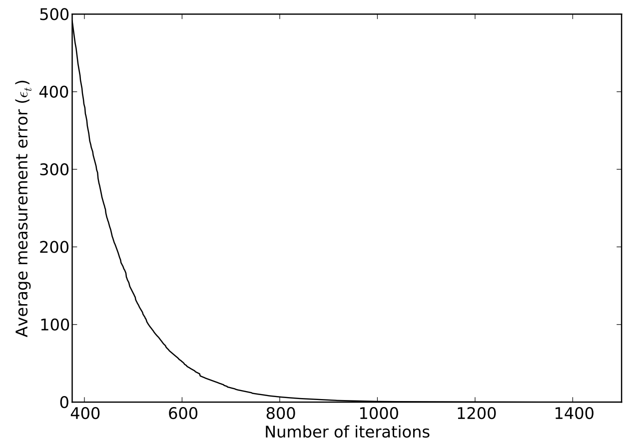


Fig. 3. Measurement error decay with iterations of randomized gossiping

In Section IV-B, we show that Comprensus performs close to the optimal case when no packet loss occurs.

A. Comparison to randomized gossiping methods

In this section, we compare Comprensus to decentralized compression based on randomized gossiping [5]. We set our SNR requirement for both protocols to 40 dB and compare their performance in achieving this requirement. Network topology and all other conditions are also the same for both protocols.

As described in Section II-B, randomized gossiping requires at least $s \log n$ data exchanges per iteration. The number of required iterations is dependent on the network topology and maximum allowed measurement error. Assume that $\mathbf{w}_i[t]$ is the content of the measurement vector of SN i at iteration t of randomized gossiping as defined in Section II-B. As we have seen in Section II-B, the difference between $\mathbf{w}_i[t]$ and $\mathbf{y} = \mathbf{\Phi f}$ shrinks to zero when $t \to \infty$.

We define the measurement error of SN i at iteration t as

$$\epsilon_{i,t} \; := \; \|\mathbf{w}_i[t] - \mathbf{y}\|_2. \tag{13}$$

We also define average measurement error at iteration t as $\epsilon_t := (\sum_{i=1}^{n} \epsilon_{i,t})/n$. From the arguments in Section II-A2 we know that for recovering $\mathbf{f}$ with SNR of at least 40 dB, the measurement error $\epsilon_{i,t}$ must be lower than $\|\mathbf{f}\|_2 \times 10^{-4}$ for the measurement vector received by SN i at time t.

Figure 3 shows how ϵ_t decays with the number of randomized gossiping iterations for a WSN consisting of $n = 128$ SNs with a network topology corresponding to a connected regular graph of degree $d = 5$. We observe that, average measurement error goes below our required threshold after almost 1200 iterations. We round down this number to 1000 iterations in favor of the randomized gossiping method.

Now we compare the total amount of transmissions in Comprensus and randomized gossiping. We consider the three test cases illustrated in Figure 2. For $s \in \{5, 10, 20\}$, randomized gossiping requires $s \log n \;\approx\; 2.1 \times s$ transmissions per iteration when $n = 128$, and thus, almost $2.1 \times 10^3 \times s$ transmissions in total, since it requires to execute 1000 iterations. Comprensus needs rk transmission corresponding to point (r, k) that falls on the bright part of the SNR diagram of

Figure 2. For s being 5, 10 and 20 we set $(r = 50, k = 30)$, $(r = 60, k = 40)$ and $(r = 80, k = 40)$ respectively. Looking at Figure 2, we see that these are rather conservative selections and signal recovery is possible with fewer numbers of transmissions. Nevertheless, we run Comprensus with these conservative settings and compare its performance to the randomized gossiping method. The comparison result is summarized in Table I.

TABLE I. COMPARING COMPRENSUS TO RANDOMIZED GOSSIPING

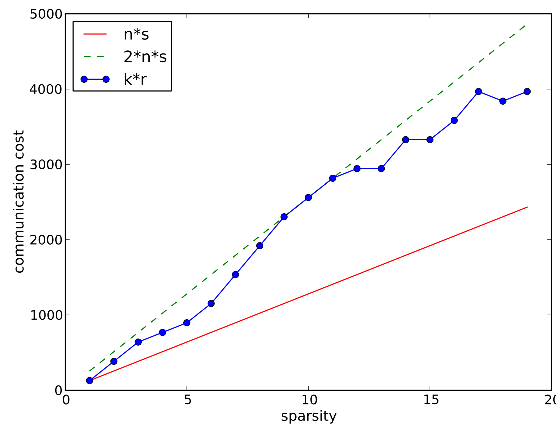
s	Total number of transmissions	
	Comprensus	Randomized gossiping
5	1.5×10^3	1.0×10^4
10	2.4×10^3	2.1×10^4
20	3.2×10^3	4.2×10^4

Comprensus proves to disseminate the random linear measurements not only in significantly less number of iterations, but also using much less amount of in-network transmissions. Using its efficient network coding technique, Comprensus disseminates compressible data with low latency and high quality while keeping the number of transmissions as low as possible in order to preserve more battery power of the SNs.

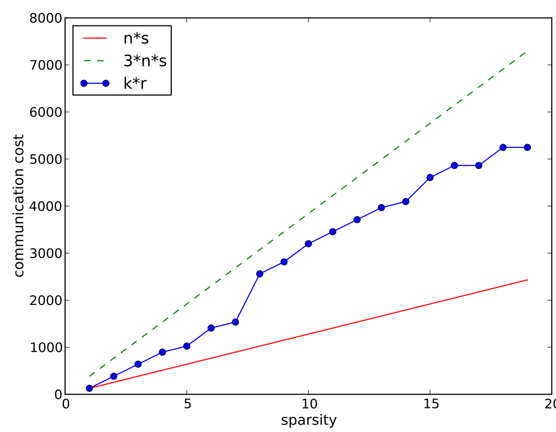
B. Comparison to oracle-based approach

Comparison with an oracle-based approach gives us a better understanding of Comprensus' performance in comparison to the optimal solution. We assume that an oracle knows all of the sensed data, i.e., the vector $\mathbf{f}$ in advance, and hence, it also knows its sparse transform, i.e., $\mathbf{x}$. The oracle broadcasts only the s significant coefficients of $\mathbf{x}$ into the network. Thus, the communication cost of the oracle-based approach is $O(sn)$. Note that broadcasting is performed hop by hop. For a limited number of neighboring hops, the broadcast of a single data item requires $O(n)$ transmissions, and thus, the total number of transmissions for broadcasting the s significant coefficients of $\mathbf{x}$ is $O(sn)$.

Figure 4.a and Figure 4.b demonstrate communication cost of Comprensus for different values of sparsity and for 35 and 40 dB SNR-thresholds respectively. We observe that, the communication cost of Comprensus is bounded between $O(s \times n)$ and $O(3 \times s \times n)$. These numerical experiments indicate that Comprensus functions almost optimally under the conditions that are applied for the simulation. A generalization of these results or a formal proof of optimality of Comprensus is regarded as an interesting direction for future work.



(a) Communication cost for SNR-threshold of 35 dB



(b) Communication cost for SNR-threshold of 40 dB

Fig. 4. Comparing the communication cost of Comprensus with the oracle-based approach

V. CONCLUSION

In this paper, we studied dissemination of compressible data using local information exchanges between the nodes in a wireless sensor network. Our approach is based on the theory of compressed sensing. We present a novel network coding protocol, named Comprensus, that enables each sensor node of a wireless sensor network to operate potentially as a sink. Our agile sink selection techniques can avail the full set of the sensed data by querying a small amount of measurements from *any* arbitrary node in the network. The techniques proposed in this paper are particulary suitable for scenarios where the sink of the wireless sensor network is mobile or when each node should access the global state of the environment.

The main advantage of the Comprensus protocol is its simple dissemination algorithm that is easily implementable on the scarce hardware resources of the sensor nodes. The complex part of the protocol, i.e., signal recovery is offloaded to an external sink or data collector that possess enough computation power.

Our evaluations show that Comprensus outperforms the state of the art methods for data dissemination in wireless sensor networks that are based on compressed sensing both in terms of communication cost and the time required for data dissemination. Our approach benefits from inherent resilience of compressed sensing to communication and measurement noise. Moreover, comprehensive simulation and experiments of our method shows a nearly optimal performance of the Comprensus protocol in a noiseless scenario. Our experiments provides a basis for further theoretical and practical investigations of our proposed method, especially for dynamic topologies.

Acknowledgment

The authors are partly supported by the Research Training Group (Graduiertenkolleg) 1362 of the German Research Foundation (Deutsche Forschungsgemeinschaft - DFG).

References

[1] K. Sohraby, D. Minoli, and T. Znati, "Wireless sensor networks, technology, protocols and applications," *Wiley-Interscience*, 2007.

[2] J. Elson and D. Estrin, "Sensor networks: A bridge to the physical world," in *Wireless Sensor Networks*, C. S. Raghavendra, K. M. Sivalingam, and T. Znati, Eds. Springer US, 2004, pp. 3–20.

[3] G. Pottie and W. Kaiser, "Wireless integrated network sensors," *Communication of ACM*, vol. 43, pp. 51–58, 2000.

[4] V. Raghunathan, C. Schurghers, S. Park, and M. Srivastava, "Energy-aware wireless microsensor networks," *IEEE Signal Processing Magazine*, pp. 40–50, 2002.

[5] M. Rabbat, J. Haupt, A. Singh, and R. Nowak, "Decentralized compression and predistribution via randomized gossiping," in *The 5th International Conference on Information Processing in Sensor Networks (IPSN), pages 51-59*, 2006.

[6] A. O. M. F. Duarte, G. Shen, and R. G. Baraniuk, "Signal compression in wireless sensor networks," *Philosophical Trans. of the Royal Society*, vol. 370, pp. 118–135, 2012.

[7] E. Candes and Y. Plan, "A probabilistic and ripless theory of compressed sensing," *IEEE Transaction on Information Theory*, vol. 57(11), pp. 7235–7254, 2011.

[8] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, "Joint source-channel communication for distributed estimation in sensor networks," *IEEE Trans. on Information Theory*, vol. 53, no. 10, pp. 3629–3653, 2007.

[9] C. Luo *et al.*, "Compressive data gathering for large-scale wireless sensor networks," in *Proc. of the 15th annual international conference on Mobile computing and networking (Mobicom)*, 2009, pp. 145–156.

[10] S. J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior point method for large-scale 11-regularized least squares," *Selected topics in Signal Processing, IEEE Journal*, vol. 1(4), pp. 606–617, 2007.

[11] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27(3), pp. 265–274, 2009.

[12] E. J. Candes, "Compressive sampling," in *Proc. of the International Congress of Mathematicians, volume 3, pages 1433-1452*, 2006.

[13] P. Santi, "Topology control in wireless ad hoc and sensor networks," *ACM Comput. Surv.*, vol. 37, no. 2, pp. 164–194, Jun. 2005.

[14] H. Niederreiter, "Recent trends in random number and random vector generation," *Annals of Operations Research*, vol. 31, no. 1, pp. 323–345, 1991.

[15] W. B. et al., "Compressive wireless sensing," in *Proc. of the 5th international conference on Information Processing in Sensor Networks (IPSN), pages 134-142*, 2006.

[16] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, "Joint source-channel communication for distributed estimation in sensor networks," *IEEE Trans. on Information Theory*, vol. 53(10), pp. 3629–3653, 2007.

[17] S. Mahfoudh and P. Minet, "Survey of energy efficient strategies in wireless ad hoc and sensor networks," in *Proc. of the 7th International Conference on Networking, pages 1-7*, 2008.

[18] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393 – 422, 2002.

[19] D. Donoho and J. Tanner, "Observed universality of phase transitions in high-dimensional geometry," *Philosophical Tran. of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 367, no. 1906, pp. 4273–4293, 2009.