Networked Computing for Structural Health Monitoring

Apoorva Jindal, *Member, IEEE,* Mingyan Liu, *Senior Member, IEEE,*

***Abstract*—This paper studies the problem of distributed computation over a wireless network of resource constrained sensor nodes. In particular, we focus our attention on sensor networks used for structural health monitoring. Within this context, the heaviest computation is to determine the singular value decomposition (SVD) to extract mode shapes (eigenvectors) of a structure. Compared to collecting raw vibration data and performing SVD at a central location, computing SVD within the network can result in a significantly smaller energy consumption and delay. Recent results have proposed methods to decompose SVD, which is a well-defined centralized operation, into components that can be carried out in a distributed way. What is missing, and is the focus of this paper, is how to determine a near-optimal communication structure that enables the distribution of this computation and the reassembly of the final results, with the objective of minimizing energy consumption subject to a computational delay constraint. We show that this reduces to a generalized clustering problem; a cluster forms a unit on which a component of the overall computation is performed. We establish that this problem is NP-hard. By relaxing the delay constraint, we derive a lower bound to this problem. We also show that the optimal solution to the unconstrained problem has a simple structure that reveals insights into the solution of the original constrained problem. We then propose an integer linear program (ILP) to solve the constrained problem exactly as well as an approximate algorithm with a proven approximation ratio. We also present a distributed version of the approximate algorithm. Numerical results are presented to demonstrate the effectiveness of the approximate and distributed algorithms.**

***Index Terms*—Networked Computing, Wireless Sensor Networks, Degree-Constrained Data Collection Tree, Singular Value Decomposition.**

I. Introduction

Over the past decade, the research community has made tremendous progress in understanding and using wireless sensor networks (WSNs). Of particular relevance to this paper are extensive studies on in-network processing, e.g., finding efficient routing strategies when data compression and aggregation are involved. However, many emerging applications, e.g., body sensor networks, structural health monitoring sensor network, and various other cyber-physical systems, require far more sophisticated data processing beyond data compression and collection, in order to enable real-time diagnosis and control.

This motivates the following question: how do we perform arbitrary (and likely complex) computational tasks using a distributed network of wireless sensors, whose inputs originate from this network of sensors, while each sensor individually has limited resources both in energy and in processing capability? The ultimate goal is *fully-automated in-network computing*, a natural progression from the networked sensing paradigm. This question poses the following two challenges. The first is the decomposition of complex computational tasks into smaller operations, each with its own input and output and collectively related through a certain data-flow or dependency graph. The second challenge is to distribute or place these operations among individual sensor nodes so as to incur minimal energy consumption and delay.

In this paper we will focus on the latter challenge within the context of using WSNs for structure health monitoring (SHM). This is an area of growing interest due to the growing need to provide low-cost and more timely monitoring and inspection of deteriorating infrastructure, but also as an appealing application of wireless sensor technologies.

The most common approach in SHM to detect damage is to collect vibration data using a set of wireless sensors in response to white/free input to the structure, and then use the singular value decomposition (SVD) to determine the set of modes [1]–[3]. A *mode* is a combination of a frequency and a shape, which is the expected curvature (or displacement) of a surface vibrating at a mode frequency.

In this study, we will use SVD as a primary example to illustrate an approach to determine how to perform such a complex computational task over a network of resource-constrained sensors. Compared to collecting raw vibration data (or the FFT of raw data) and performing the SVD computation at a central location, directly computing SVD within the network can result in a significant reduction in both energy consumption and computational delay. Conceptually, this reduction occurs because the output of SVD, a set of eigenvectors, is much smaller in size than its input, FFTs from individual sensor data streams, and evaluating multiple, smaller SVDs in parallel is much faster than evaluating the SVD on the input from all sensors.

How to decompose the SVD computation, which a well-defined classical centralized operation, into components that can be carried out in a distributed way was studied by Zimmerman *et al.* [4]. In this paper, we focus on the next step which is to determine a near-optimal communication structure that enables the distribution of this computation and the reassembly of the final results. We define an optimization framework that seeks the best computation and communication structure with the objective being to minimize energy consumption subject to a computational delay constraint. We show that this reduces to a generalized clustering problem; a cluster forms a unit on

A. Jindal and M. Liu are with the Department of Electrical Engineering and Computer Science at University of Michigan, Ann Arbor. E-mail: apoorvaj@umich.edu,mingyan@eecs.umich.edu.

which a component of the overall computation is performed.

Previous results on establishing the communication structure for in-network computation either consider only very simple functions like max/min/average/median [5]–[8] that do not fully represent the complex computational requirements demanded by practical engineering applications like SVD computation, or study scaling laws which do not yield algorithms to determine the optimal communication structure [9]. Finally, note that SVD computation is an essential ingredient in a broad class of signal processing applications, including classification, identification and detection [4], [10]–[14].

We first formally define the above problem and establish that it is NP-hard in Section II-E. By relaxing the delay constraint, in Section III we derive a lower bound to this problem, and show that the optimal solution to the unconstrained problem has a simple structure that reveals insights into the original problem. We then propose an integer linear program (ILP) to solve the constrained problem exactly as well as an approximate algorithm with a proven approximation ratio in Section IV. We also present a distributed version of the approximate algorithm. Numerical results are presented in Section V to demonstrate the effectiveness of the approximate and distributed algorithms. Note that even though our presentation, for the most part, is focussed specifically on SVD, the methodology itself is generic. In Section VI, we discuss how a similar approach can be used for other computational tasks in SHM like distributed optimization using simulated annealing. Finally, we conclude in Section VII.

Before we end this introduction, we present a simple example to illustrate that the optimal communication structure can differ depending on the computational objective, and hence, prior work on deriving the routing structure for data aggregation cannot be applied here. We will compare the optimal routing structure for data compression and that of computing the SVD. Assume that data compression converts 2 input streams of size R bits each to an output stream of $R+r$ where $r<R$ [15]. The SVD operator, as discussed in detail in Section IV, converts k input streams of size R bits each, to k eigenvectors of size r bits each with $r<R$. Now consider the simple 4-node topology of Figure 1 and the two possible communication structures, with node 0 being the base station (or data collector/processor) and assuming all links are of unit length/cost. As derived in [15], data compression requires an exchange (or incurs a cost) of $3R+3r$ (using successive encoding) and $4R+r$ bits respectively for the communication structures (a) and (b). Hence, if $R>2r$, the one on the left is better. On the other hand, in the case of SVD if we do not perform in-network computation, then sending all raw data to node 0 results in a cost of $6R$ and $5R$ over the two structures, respectively. If we perform in-network computation, then as detailed in Section IV, the resulting costs are $3R+6r$ and $3R+3r$ for the two structures, respectively. Hence, the second communication structure is always better for the SVD computation.

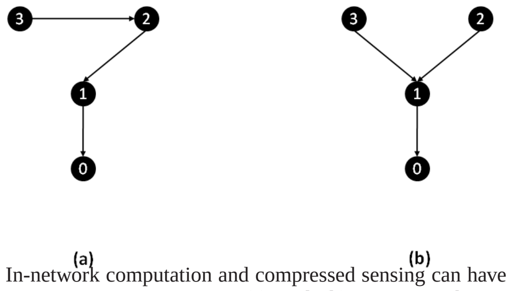


Fig. 1. In-network computation and compressed sensing can have a different optimal communication structure. (a) and (b) represent the two possible communication structures for a simple 4-node topology.

II. Problem Formulation

In this section, we first introduce the relevant background on structural health monitoring, then present the network model and formally introduce the problem.

A. Background on Structural Health Monitoring

During the past two decades, the SHM community has become increasingly focussed on the use of the structural vibration data for identifying degradation or damage within structural systems. The first step in determining if the vibration data collected by a set of sensors represents a healthy or an unhealthy structure is to decompose the spectral density matrix into a set of single degree of freedom systems. Assuming a broadband white input to the system, this can be accomplished by first obtaining an estimate of the output power spectral density (PSD) matrix for each discrete frequency by creating an array of frequency response functions using the Fast Fourier Transform (FFT) information from each degree of freedom in a system. Early studies in this field focussed on identifying changes in modal frequencies or the eigenvalues of the PSD matrix using the peak picking method [16] to detect damage in large structural systems [17]. More recent studies have observed that viewing changes in modal frequencies in combination with changes in mode shape information (eigenvector of the PSD matrix) makes it increasingly possible to both detect and locate damage within a variety of structural types and configurations [1]–[3]. One of the most widely used method for mode shape estimation is the frequency domain decomposition (FDD) method which was proposed by Brincker *et al.* [18]. This method involves computing the SVD[1] of the PSD matrix to extract the eigenvectors/mode shapes.

The most common implementation of the FDD method over a wireless sensor network is to have each sensor send its vibration data to a central sensor node which computes the SVD of the PSD matrix and then distributes the mode shapes back to each sensor. This method requires significant computational power and memory at the central sensor node as well as a significant energy consumption in the network to communicate all this data to the sensor node. For example, if there are 100 sensor nodes in the network, this implementation requires the central sensor node to compute the SVD of a 100×100 PSD matrix as well as having each of the 100 sensor nodes send all their vibration data to one central node.

Within a wireless sensing network, where processing power, energy and memory at each node is limited, Zimmerman *et al.* [4] proposed an alternative implementation by decomposing the computation of SVD using in-network computation

[1]Please see Appendix A for a detailed description of the SVD computation.

(graphically represented in Figure 2). Each sensor node is assumed to be aware of the eigenvalues of the PSD matrix (which have already been determined using the peak-picking method[2]) and the FFT of its own sensed data stream. Now, if a sensor has the FFT of $N \subset V, |N| > 1$ sensors and all the eigenvalues, then it can compute the SVD of the PSD matrix using $|N|$ sets of FFT results and determine $|N|$ eigenvectors. Let another sensor node be in possession of the FFT of $N' \subset V, |N'| > 1$ sensors. It can do a similar computation to determine $|N'|$ eigenvectors. To be able to combine results from these two computations to construct the $|N \cup N'|$ eigenvectors, one needs to be able to determine the appropriate scaling factors. We call two computations *combinable* if one can determine the appropriate scaling factors to combine them. A computation on N nodes and another computation on N' nodes is combinable if and only if either $N \cap N' \neq \phi$ (that is, there is at least one common sensor in N and N'), or there exists another computation on N'' nodes which is combinable with both N and N'.

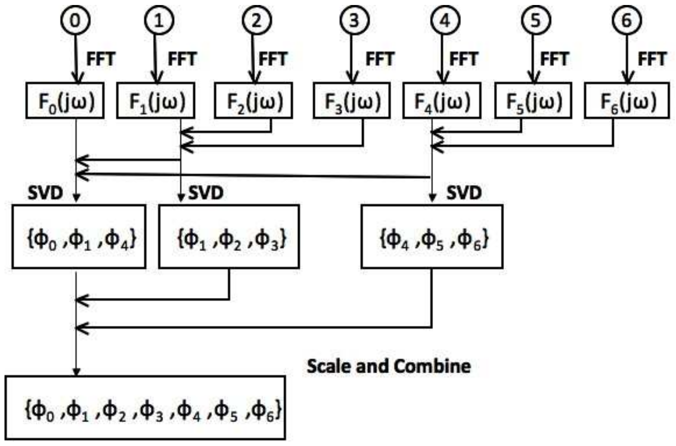

0
1
2
3
4
5
6
FFT
$F_0(j\omega)$
$F_1(j\omega)$
$F_2(j\omega)$
$F_3(j\omega)$
$F_4(j\omega)$
$F_5(j\omega)$
$F_6(j\omega)$
SVD
$\{\phi_0, \phi_1, \phi_4\}$
$\{\phi_1, \phi_2, \phi_3\}$
$\{\phi_4, \phi_5, \phi_6\}$
Scale and Combine
$\{\phi_0, \phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \phi_6\}$

Fig. 2. Decomposing the computation of SVD using in-network computation.

Each independent in-network computation requires message exchange with other in-network computations to make them combinable. Also, if R denotes the size in bits required to represent the FFT of a sensor stream and r denotes the size in bits to represent a eigenvector, each in-network computation which combines the FFT of k sensor streams reduces the number of bits in the network from kR to kr. Note that the size of the output stream does not depend on R.

Though Zimmerman *et al.* [4] proposed a new data-flow graph for SVD, they assumed a fixed routing structure and did not consider optimizing the communication structure, which will be the focus of this paper.

B. Network Model

A network of sensor nodes is represented with an undirected, unweighted graph $G(V,E)$. Each node in V acts as both a sensor and a relay. If two nodes can successfully exchange messages with each other, there exists an edge $e \in E$ between them. Let there be a weight $w_e \geq 0$ associated with each edge which denotes the energy expended in sending a packet across this edge and depends on the number of transmissions required to send a packet across that edge. Without loss of generality, we assume node 0 to be the central sensor node or the base station. We also assume that all sensors (including the base station) are identical in their processor and radio (and hence computational time and energy consumption per bit). This is done to keep the presentation simple and can be easily relaxed.

Each node has a local input vibration stream. The objective of the network is to evaluate the SVD of the PSD matrix formed by the input vibration streams of all the sensors. A sensing cycle is defined to be the time duration in which each sensor performs the sensing task to generate a vibration stream, the SVD is computed and the mode shapes are made known at the base station. The sensing rate depends inversely on the duration of one sensing cycle. Our objective, as will be more precisely discussed in Section IV, is to determine the optimal communication structure to minimize the energy consumption in a sensing cycle under a constraint on the maximum duration of a sensing cycle.

C. Metrics of Interest

Energy Consumption: this is defined to be the total communication energy consumed in the network in one sensing cycle. Let E_{Tx} and E_{Rx} denote the energy consumption to transmit and receive a bit of data. Then we assume that the energy consumed in transmitting a packet of B bits over an edge e is $w_e B(E_{Tx} + E_{Rx})$[3].

Computational Delay: this is defined to the maximum computational delay at a sensor node. As observed in [4], [21], the computational time is the chief contributor to delay as packet sizes in sensor systems tend to be very small. Thus, the duration of a sensing cycle depends chiefly on the maximum computational delay amongst all sensor nodes. In other words, the computational delay constraint imposes a constraint on the maximum duration of a sensing cycle.

D. Formal Definition

We now formally introduce the problem. Determining the optimal communication structure implies finding the set S of sensor nodes on which the SVD computation will take place, and for each $s \in S$, finding the corresponding set of sensors N_s whose FFT will be made available at s. Recall that R and r denote the number of bits required to represent the FFT from a single sensor and a single eigenvector respectively. The computational delay constraint imposes a constraint on the maximum number of FFT's which can be combined at a sensor node. Let $C(|N_s|)$ be the time it takes to compute

[2]Finding the optimal communication structure to implement peak-picking in a distributed manner is the same as determining the optimal communication structure for networked sensing with data compression [15], [19], [20]. And the algorithms developed in these papers can be directly applied to peak-picking. Hence, to keep the exposition interesting, we do not study implementing peak-picking distributively here.

[3]Our algorithms and the corresponding approximation factors do not depend on the exact model used for energy consumption provided the energy consumed remains a function of the number of bits transmitted over a network. Thus, nothing changes if a more complex model for energy consumption which incorporates energy consumed in overhearing is used. We discuss this in more detail in Section IV-C

the SVD of the FFT from $|N_s|$ sensors, $C(|N_s|) \leq C, \forall s \in S$. This constraint is equivalent to $|N_s| \leq d$, $\forall s \in S$ where $d := \max\ \{|N_s| \mid C(|N_s|) \leq C\}$. Also, recall that another constraint is that the computation on each pair $s_1, s_2 \in S$ is combinable.

Definition 1: **P1.** Find the set S and their corresponding $N_s, \forall s \in S$, and the routing structure to minimize the total energy consumed such that $|N_s| \leq d$, $\forall s \in S$ and the computations on all pairs $s_1, s_2 \in S$ are combinable.

Note that it is easy to modify our algorithms to minimize the maximum computational delay with a constraint on the energy consumption. Indeed, the dual of the linear programs we propose will optimally solve this alternative formulation. However, due to space limitations, we do not explore this dual formulation in this paper.

E. NP-completeness

A decision version of P1 can be shown to be NP-hard through a reduction from set cover.

Theorem 1: There is no polynomial time algorithm that solves P1, unless $P = NP$.

Proof: See Appendix B. ■

III. A Lower Bound

We first derive a lower bound on the optimal value. This lower bound is obtained by studying P1 without the computational delay constraint. This study also provides valuable intuition into the development of an approximation algorithm for P1.

To simplify the presentation, in this section, we assume that the weight of all edges is equal. Note that this not a stringent assumption as all the bounds derived in this section can be easily modified to incorporate different weights for each edge. With this assumption, the energy consumed to send data from node i to node j will merely depend on the number of hops on the shortest path between these two nodes.

Definition 2: **[Data Collection Tree]** A data collection tree for $G(V,E)$ is the spanning tree such that the path from each node $v \in V$ to the base station has the minimum weight.

Note that since all edges have the same weight, a path of minimum weight is equivalent to the path with the minimum hop count. Let T denote the data collection tree for $G(V,E)$, and let $d_T(v)$ denote the hop count of node $v \in V$ in T.

The following lemma derives a lower bound on the optimal energy consumption.

Lemma 1: $\left((|V|-1)R + \sum_{v \in V}(d_T(v)-1)r\right)(E_{Tx}+E_{Rx})$ is a lower bound on the optimal energy consumption.

Proof: For all the nodes $v \in V \backslash S$, a message exchange of size R from v to one of the nodes in S will occur. This message will go over at least one hop. If the hop count of node v from the base station is equal to $d_T(v)$, and if the message of size R goes over one hop, the message of size r will go over at least $d_T(v)-1$ hops. This will require at least $|V|-|S|$ message exchanges of size R and $\sum_{v \in V\backslash S}(d_T(v)-1)$ message exchanges of size r.

Now, each of the S computations should be combinable, that is, $\forall s_1, s_2 \in S$, either $N_{s_1} \cap N_{s_2} \neq \phi$ or there exists another node $s_3 \in S$ such that the computation at nodes s_1 and s_3 as well as on nodes s_2 and s_3 are combinable. To understand how many extra messages are needed to satisfy this constraint, consider the following graph $G^S(S,E^S)$. If for nodes $s_1, s_2 \in S$, $N_{s_1} \cap N_{s_2} \neq \phi$, we introduce an edge between s_1 and s_2 in E^S. Each edge in this new graph implies at least one extra message exchange of size R in addition to the $|V|-|S|$ message exchanges of size R. Two nodes in $s_1, s_2 \in S$ are combinable if and only if there exists a path between s_1 and s_2 in $G^S(S,E^S)$. For a path to exist between every pair of nodes, $G^S(S,E^S)$ should have at least $|S|-1$ edges. This implies that at least $|S|-1$ extra message exchanges of size R are required for all pairs $i, j \in S$ to be combinable.

Thus, at least $|V|-1$ message exchanges of size R will occur. Also, the computed eigenvectors will go through $d_T(v)$ hops for all nodes $v \in S$. Thus, messages of size r will go through at least $\sum_{v \in S} d_T(v)$. Thus, the optimal energy consumption is $\geq \left((|V|-1)R + \sum_{v \in V\backslash S}(d_T(v)-1)r + \sum_{v \in S} d_T(v)r\right)(E_{Tx}+E_{Rx}) \geq \left((|V|-1)R + \sum_{v \in V}(d_T(v)-1)r\right)(E_{Tx}+E_{Rx})$. ■

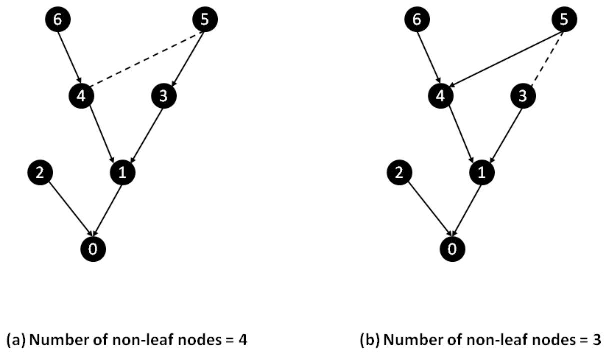

(a) Number of non-leaf nodes = 4
(b) Number of non-leaf nodes = 3

Fig. 3. Two data collection trees for the same network. The solid lines represent the edges of the tree. T_M is the tree in (b).

We now construct the optimal solution to P1 without the computational delay constraint under the assumption that $R > 2r$. (Note that the condition that $R > 2r$ is satisfied by the SVD computation for structural health monitoring.) Consider a data collection tree with the following property. All the children of a non-leaf node $v \in V$ in the tree cannot be moved to other nodes of height $\leq d_T(v)$. Thus, this tree has the minimum number of non-leaf nodes. Label this tree T_M. (Figure 3 gives an example to clarify the difference between T_M and another data collection tree.)

Theorem 2: The following solution to P1 is optimal without the computational delay constraint. S consists of all non-leaf nodes in the data collection tree T_M, $N_s, s \in S$ consists of all the immediate children of s and the data collection tree T_M is the routing structure.

Proof: Each sensor node sends its FFT to its parent (which incurs an energy cost of $(E_{Tx}+E_{Rx})R$). Since, the base station has no parent, this step incurs an energy cost of $(E_{Tx}+E_{Rx})(|V|-1)R$. Each non-leaf node computes the SVD from the FFT of its children's stream and its own stream, and then sends the eigenvectors to the base station. It incurs an energy cost of $\sum_{v \in V}(d_T(v)-1)r + |S|r$. The extra energy cost over the lower bound of Lemma 1 is equal to $|S|r$.

Thus, a larger S will only increase the energy consumption. We next show that a smaller S does not decrease the energy consumption either. We prove this by contradiction. Let S' denote a set which solves P1 without the computational delay constraint, results in a smaller energy consumption than S and $|S'| < |S|$. Using S' instead of S reduces the energy consumption by no more than $(|S| - |S'|)r$. Consider node $s \in S$ but not in S'. By definition of T_M and since $R > 2r$, none of its children are being evaluated at a node $s' \in S'$ such that $d_T(s') < d_T(s)$ and at least one of its children is being evaluated at a node $s'' \in S'$ which is at height greater than $d_T(s)$. Thus, the increase in energy consumption for any $s \in S$ but not in S' is equal to r. Since there are at least $|S| - |S'|$ such nodes, the increase in energy consumption is at least $(|S| - |S'|)r$. Hence, the energy consumption does not reduce which is a contradiction.

Finally, we prove that each pair of nodes $s_1, s_2 \in S$ are combinable. Note that removing the leaf nodes and the edges connecting the leaf nodes to the non-leaf nodes yields $G^S(S, E^S)$. Since, there exists a path between each pair of non-leaf nodes in T_M (and this path obviously does not go through a leaf node), every pair of nodes $s_1, s_2 \in S$ are combinable. ■

To summarize, constructing a data collection tree with the minimum number of non-leaf nodes yields the optimal solution for P1 without the computational delay constraint. Also, note that any other data collection tree T_D will yield a solution which has an additive extra energy cost of $(NL(T_D) - NL(T_M))r$ where $NL(T)$ represents the number of non-leaf nodes in the data collection tree T.

IV. Algorithms

In this section, we propose an exact ILP as well as a $\Theta(log(|V|))$ approximation algorithm for P1.

A. ILP

In this section, we propose an ILP to solve P1. We first define some extra variables for notational convenience. x_{ij} will be set to 1 if the FFT of sensor node i is sent to node j (i.e. $i \in N_j$), otherwise it will be set to 0. x_{ii} will be set to 1 only if $i \in S$. Thus, x_{ij} is the variable which defines both the set S as well as N_s. p_{ijk} is a variable which will be set to 1 if the FFT of the sensor node k is evaluated at both nodes i and j. Finally, we define c_{ijn} as,

$$c_{ijn} = \begin{cases} 1 & n = 0 \text{ and } \sum_{k \in V} p_{ijk} \geq 1, \\ 1 & 0 < n < |V| \text{ and } \sum_{k \in V} c_{ik(n-1)} . c_{jk(n-1)} + c_{ij(n-1)} \geq 1, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Thus, $c_{ij(|V|-1)}$ will be equal to 1 if pairs $i, j \in S$ are combinable.

Let $H_{i \to j}$ denote the sum of the weights of the edges lying on the shortest path from node i to node j. Following is the ILP to solve P1 exactly.

$$\min \sum_{i \in V, j \in V} x_{ij} (E_{Tx} + E_{Rx}) \left(RH_{i \to j} + rH_{j \to 0} \right) \tag{2}$$

$$\text{s.t.}$$

$$\sum_{j \in V, j \neq i} \frac{x_{ji}}{V} \leq x_{ii} \leq \sum_{j \in V, j \neq i} x_{ji}, \forall i \in V \tag{3}$$

$$\sum_{j \in V} x_{ij} \geq 1, \forall i \in V \tag{4}$$

$$p_{ijk} \leq \frac{x_{ki} + x_{kj}}{2}, \forall i, j, k \in V \tag{5}$$

$$c_{ij0} \leq \sum_{k \in V} p_{ijk}, \forall i, j \in V \tag{6}$$

$$c_{ij(|V|-1))} \geq \frac{x_{ii} + x_{jj}}{2}, \forall i, j \in V \tag{7}$$

$$t_{ijkn} \leq \frac{c_{ik(n-1)} + c_{jk(n-1)}}{2}, \forall i, j, k \in V, 0 < n < |V| \tag{8}$$

$$c_{ijn} \leq c_{ij(n-1)} + \sum_{k \in V} t_{ijk(n-1)}, \forall i, j \in V, 0 < n < |V| \tag{9}$$

$$c_{iin} = 0, \forall i \in V, 0 \leq n < |V| \tag{10}$$

$$\sum_{i \in V} x_{ij} \leq d, \forall j \in V \tag{11}$$

$$x_{ij}, c_{ijk}, p_{ijk}, t_{ijkn} \in \{0, 1\} \forall i, j, k \in V, 0 \leq n < |V| \tag{12}$$

We now explain in detail how the objective is set up as well as the implication of each constraint. We first look at the objective (Equation (2)). If the FFT from sensor node i is sent to node j, it consumes $RH_{i \to j}(E_{Tx} + E_{Rx})$. Node j evaluates the SVD and sends the eigenvector to the base station for putting all the eigenvectors together. Since, the FFT from sensor i will generate a unique eigenvector of size r, an additional $rH_{j \to 0}(E_{Tx} + E_{Rx})$ amount of energy is consumed in sending the eigenvector to the base station.

The first constraint (Equation (3)) sets the value of x_{ii} to be equal to 1 if $N_i \neq \phi$, else it is set to 0. (Note that if $N_i \neq \phi$, $1 \leq \sum_{j \in V, j \neq i} x_{ji} \leq |V|$). The second constraint (Equation (4)) ensures that the FFT of every sensor node is sent to at least one node. The third constraint (Equation (5)) ensures that p_{ijk} is equal to 1 if the FFT from node k is sent to both nodes i and j.

The next five constraints set the value of c_{ijn}. Recall that the purpose of introducing c_{ijn} is to ensure that all pairs $i, j \in S$ are combinable. The fourth constraint (Equation (6)) ensures that the value of c_{ij0} is 1 if there is at least one node whose FFT is being sent to both i and j. The fifth constraint (Equation (7)) states that if both $i, j \in S$, the computations at i and j should be combinable. The next two constraints (Equation (8) and (9)) populate the value of c_{ijn}. Note that t_{ijkn} is a temporary variable introduced to express the quadratic condition in Equation (1) as a linear function. Equation (10) sets the value of c_{iin} to zero for every $i \in V, 0 \leq n < |V|$. This prohibits a corner case where c_{ijn} is set to 1 by setting $c_{ii(n-1)}$ to 1 without ensuring that the computation at i and j are combinable.

Finally, Equation (11) imposes the computational delay constraint at each sensor node.

B. An Approximation Algorithm

In this section, we propose a $\Theta(log(|V|))$ approximation algorithm. To simplify the presentation, we again assume that the weight of all edges is equal. Note that all the algorithms proposed in this section can be easily modified without changing their approximation factors to incorporate different weights for each edge.

1) Degree-Constrained Data Collection Tree: Using a data collection tree to build the solution to P1 will violate the computation delay constraint if the number of immediate children a node has is greater than $d-1$. (Note that a node in S will include the FFT of its own data stream in its computation, hence, having more than $d-1$ immediate children will violate the computational delay constraint.) A data collection tree with the additional constraint that no sensor node in the tree has more than $d-1$ immediate children will satisfy the computational delay constraint, but may no longer be optimal even if it has the fewest non-leaf nodes.

Definition 3: **P2.** Find the data collection tree for $G(V,E)$ such that no sensor node has more than $d-1$ immediate children.

P2 is also NP-hard. Its APX-hard even when weights on edges satisfy the triangle inequality [22]. Results for P2 are known only for complete graphs whose weights satisfy the triangle inequality [22], [23], and our work is the first to propose approximation algorithms and analytically derive their approximation factors for P2 in graphs induced by a communication network.

We will first propose an ILP to solve P2. The advantage of this new ILP over the ILP presented in Section IV-A for P1 is that it has much fewer variables and constraints, and hence takes less time to solve. We then propose a new approximation algorithm for P2 based on relaxing the ILP and then appropriately rounding the fractional values. This algorithm is also an approximation algorithm for P1. We derive the approximation factor for this algorithm (with respect to the original problem) in Theorem 3. Finally, based on the intuition derived while analyzing the approximation algorithm, we present a simpler, distributed approximation algorithm with the same asymptotic approximation factor.

2) The ILP: We first present an ILP to solve P2. We define the following variables for notational convenience. For a given $G(V,E)$, define a graph $\overline{G}(V,\overline{E})$ with directed edges. Each undirected edge in E is replaced by two directed edges, one in each direction to construct $\overline{E}$. For each edge $e \in \overline{E}$ between nodes $i \in V$ and $j \in V$, the complementary edge $\hat{e} \in \overline{E}$ is defined to be the edge between nodes j and i. Let $O_v, v \in V$ denote the set of outgoing edges from node v in $\overline{E}$. Similarly, let $I_v, v \in V$ denote the set of incoming edges into node v in $\overline{E}$.

We next define the variables used in the ILP. $x_e, e \in \overline{E}$ is set to 1 if the edge e is part of the data collection tree, else it is set to 0. $f_e, e \in \overline{E}$ denotes the flow value traveling through the edge e. If an edge is not a part of the data collection tree, the flow through it should be constrained to be equal to 0.

The following set of equations define the ILP which will

```
$NV=\{0\}$, $NE=\phi$, $h=0$, assign $h_v=-1,\ \forall v \in V\backslash\{0\}$ and
$h_0=0$
while $(NV!=V)$ do
    $h=h+1$
    Solve the ILP for P2 with fractional variables
    and the additional constraint that $x_e=1, \forall e \in NE$
    For $\forall v \in NV$ and $h_v=h-1$
        If the value of $x_e$ for more than $(d-1)$
        incoming edges at $v$ is greater than $0$
            Set the largest $(d-1)$ $x_e$ values
            amongst the incoming edges at $v$ to $1$
            (ties are broken arbitrarily)
        Otherwise
            Set the $x_e$ value of all incoming
            edges at $v$ to $1$
        Add the edges for which $x_e$ was set to $1$
        in the previous step to $NE$
        For all edges added to $NE$ in the
        previous step, add the node $v$ from
        which the edge emanates to $NV$ and
        assign $h_v=h$
```

Fig. 4. Algorithm A1: The LP rounding approximation algorithm for P2.

solve P2 exactly.

$$\min \textstyle\sum_{e\in\overline{E}} f_e \tag{13}$$

$$\textstyle\sum_{e\in I_0} f_e - \sum_{e\in O_0} f_e = |V|-1 \tag{14}$$

$$\textstyle\sum_{e\in I_v} f_e - \sum_{e\in O_v} f_e = -1, \forall v \in V\backslash\{0\} \tag{15}$$

$$f_e \leq (|V|-1)\left(x_e + x_{\hat{e}}\right), \forall e \in \overline{E} \tag{16}$$

$$\textstyle\sum_{e\in\overline{E}} x_e = |V|-1 \tag{17}$$

$$\textstyle\sum_{e\in O_v} x_e = 1, \forall v \in V\backslash\{0\} \tag{18}$$

$$\textstyle\sum_{e\in I_v} x_e \leq d-1, \forall v \in V \tag{19}$$

$$x_e \in \{0,1\}, \forall e \in \overline{E} \tag{20}$$

$$f_e \in \{0,1,\ldots,|V|-1\}, \forall e \in \overline{E} \tag{21}$$

We now explain in detail how the objective is set up as well as the implication of each constraint. Minimizing the total flow forces each node to send its data to the base station through the shortest path. The first two constraints (Equations (14) and (15)) ensure that each node sends a unit flow towards the base station. The third constraint (Equation (16)) forces f_e to be 0 if x_e is 0, otherwise, it is redundant. The fourth constraint (Equation (17)) ensures that the output is a tree with exactly $|V|-1$ edges. The fifth and the sixth constraint (Equations (18) and (19)) ensure that there is no more than one outgoing edge per vertex (other than the base station) and no more than $d-1$ incoming edges per vertex. This ensures that no sensor node has more than $d-1$ immediate children.

3) Algorithm A1: The LP Rounding Approximation Algorithm: We next present a polynomial-time approximation algorithm which relaxes the ILP presented in Section IV-B2 and then appropriately rounds the fractional values. The ILP is relaxed by allowing x_e and f_e to be fractional, and adding the constraints $0 \leq x_e \leq 1$ and $f_e \geq 0$ $, \forall e \in \overline{E}$. The fractional values obtained by solving the linear program are rounded through the algorithm presented in Figure 4.

4) The Approximation Factor: Even though the approximation algorithm is general and makes no assumption on the network, the derivation of the approximation factor makes the following assumptions. (i) $d \geq 3$, (ii) The height of the unconstrained data collection tree derived from the algorithm presented in Theorem 2 is $O(log(|V|))$, and (iii) We assume that nodes can transmit to each other if the distance between them is less than the transmission range. Let R_{tx} denote the transmission range.

Before presenting the analysis, we first discuss the implications of these assumptions briefly. (i) $d = 2$ implies that only one other sensor's data stream can be combined at a node. Then, building an optimal degree-constrained data collection tree is equivalent to the traveling salesman problem and the corresponding approximation algorithms [24] with a better approximation factor can be directly applied here. So, we exclude this special case from our analysis. (ii) Since the sensor networks are assumed to be not very sparse, the assumption on the height of the data collection tree will be satisfied by most networks. Thus, this assumption is not restrictive. (iii) For analytical tractability, this is the most common assumption made to define when two nodes can transmit to each other. However, our analysis is not heavily dependent on this assumption as we discuss in the footnote in the proof of Lemma 3 and the derived approximation factor will hold for more realistic physical layer assumptions also.

We next show that the approximation factor of the proposed algorithm with respect to P1 is $\Theta(log(|V|))$. The derivation of the approximation factor is based on the following observation. The approximation factor is equal to the ratio of the height of the data collection tree constructed using algorithm A1 and the height of the data collection tree constructed in Theorem 2. (Note that we have assumed that the height of the data collection tree constructed in Theorem 2 is $O(log(|V|))$.)

We first prove a lemma which will be later used in the derivation of the approximation factor. We define the following variables for notational convenience. Let there be m nodes, amongst which the maximum number of nodes which cannot transmit to each other be p. We run algorithm A1 on this set of m nodes with a randomly selected base station. Let the height of the corresponding data collection tree be h_T. Define a non-full node to be a node at height $h < h_T$ which has less than $d-1$ children. A height $1 \leq h < h_T$ is defined to be a non-full height if there exists at least one non-full node at height h.

Lemma 2: The data collection tree cannot have more than p non-full heights.

Proof: We prove by contradiction. Let there be $p+1$ non-full heights: $h_1 < \ldots < h_{p+1}$. Let v_i be a non-full node at height $h_i, 1 \leq i \leq p+1$. Then, v_i, v_j, $1 \leq i < j \leq h_{p+1}$ cannot transmit to each other, otherwise the approximation algorithm would put v_j as the child of v_i. Thus none of the nodes $v_1, \ldots, v_{p+1}$ can transmit to each other. However, by assumption we cannot have more than p nodes which cannot transmit to each other. Hence, a contradiction. ■

We next derive the height of the tree constructed algorithm A1. We define the following variables for convenience. Let the height of the data collection tree constructed using the algorithm of Theorem 2 be h_{orig} ($= O(log(|V|))$ by assumption).

Lemma 3: The height of the tree constructed by algorithm A1 is $\Theta(log(|V|))$.

Proof: We will first show that the maximum number of nodes none of which can transmit to each other is no more than $2\pi clog(|V|)$ where c is a constant. Recall that the transmission range of sensor nodes is denoted by R_{tx}. Then, the maximum distance of a node from the base station is $h_{orig}.R_{tx}$[4]. Using the geometric arguments similar to the ones used in [27], its easy to show that the maximum number of nodes none of which can transmit to each other is equal to $\frac{2\pi}{\cos^{-1}\left(1-\frac{1}{2h_{orig}^2}\right)} \leq \frac{2\pi}{\cos^{-1}\left(1-\frac{1}{2c^2\log^2(|V|)}\right)} = 2\pi c\log(|V|)$ where c is a constant. (The final equality follows from the small angle approximation $\cos x \approx 1 - \frac{x^2}{2}$.)

We are now ready to prove the lemma. By Lemma 2, there are no more than $2\pi clog(|V|)$ non-full heights. And the number of full heights is $\Theta(log_{d-1}(|V|))$ by definition. Hence, the height of the tree constructed by algorithm A1 is $\Theta(log(|V|))$. ■

Finally, we derive the approximation factor for algorithm A1.

Theorem 3: The approximation factor for algorithm A1 is $\Theta(log(|V|))$.

Proof: The height of the data collection tree constructed using the algorithm of Theorem 2 is $O(log(|V|))$; thus its height can be a constant. However, the height of the tree constructed by algorithm A1 is $\Theta(log(|V|))$. Hence, the approximation factor is equal to $\Theta(log(|V|))$. ■

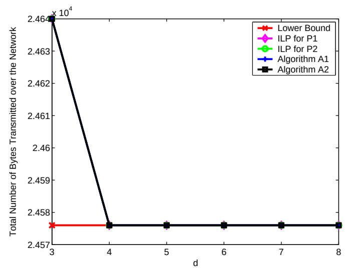
5) Algorithm A2: A Distributed Approximation Algorithm: The approximation algorithm presented in the previous section is a centralized algorithm as it requires solving a global linear program. We now present a simpler, distributed algorithm which has the same asymptotic approximation factor.

The proof of Theorem 3 uses the following observation. At a height h, if there exists a node with more than $d-1$ neighbors which are not yet a part of the tree, the algorithm will add $d-1$ children to it. Otherwise, all its neighbors not yet a part of the tree will be added as its children.

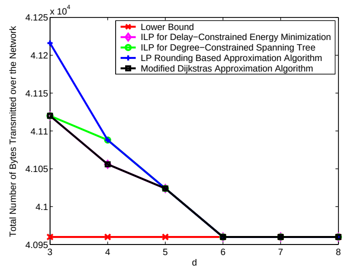
Using this intuition, we propose a modified version of Dijkstra's shortest path algorithm in Figure 5. This algorithm satisfies the observation made in the previous paragraph, and, hence has the same approximation factor of $\Theta(log(|V|))$ as the algorithm proposed in Figure 4. This algorithm can be easily distributed in a manner similar to any shortest path routing algorithm [28]. The tree is build top down from the root with each node choosing its $d-1$ children arbitrarily. Hence, like any shortest path algorithm, it can be built by message exchanges only between neighboring nodes.

We compare the modified Dijkstra's algorithm with the LP rounding approximation algorithm through simulations in Section V and find that the modified Dijkstra's algorithm always outperforms LP rounding and is always within 3% of the optimal.

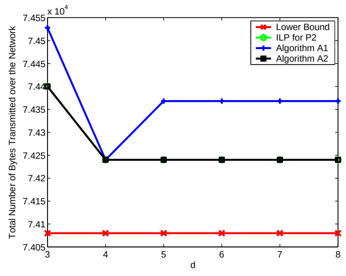
[4]Note that due to fading effects, the transmission range may not be a constant; it may be even time-varying. However, there will always exist distances R_0 and R_1 such that if two nodes are within a distance R_0 of each other, they can transmit to each other with negligible loss, and if they are at a distance more than R_1, they cannot exchange packets with each other at all [25], [26]. R_0 and R_1 may be much smaller and larger respectively than the actual transmission range, but they will still be a given constant. Replacing R_{tx} by these constants appropriately allows the argument to go through for a more general physical layer model.



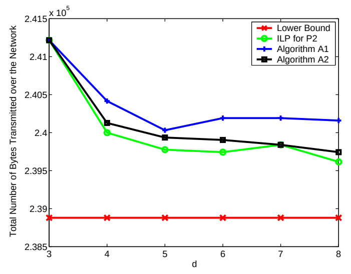
(a)



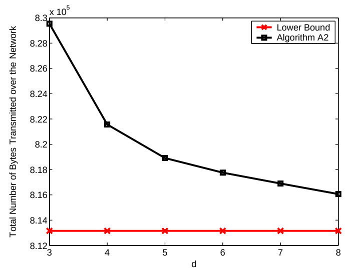
(b)



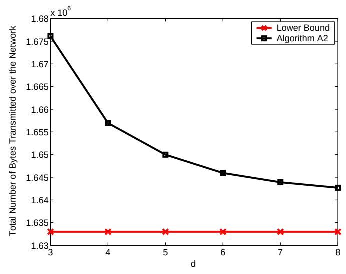
(c)



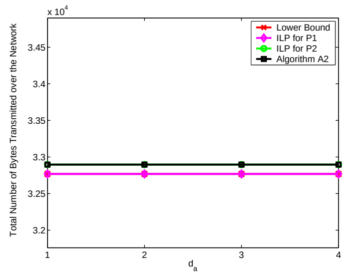
(d)



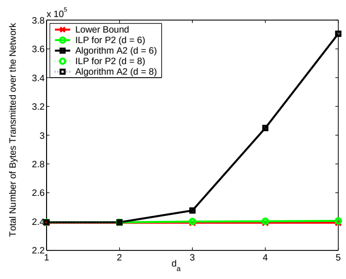
(e)



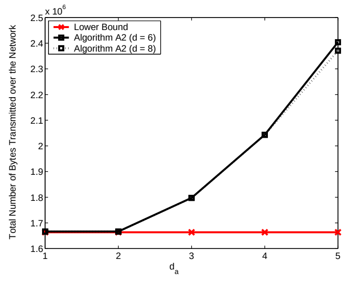
(f)



(g)



(h)



(i)

Fig. 6. Simulation Results. (a) $|V| = 4$ (24576). (b) $|V| = 6$ (40960). (c) $|V| = 10$ (163840). (d) $|V| = 30$ (573440). (e) $|V| = 100$ (1359872). (f) $|V| = 200$ (2342912). The number in brackets denotes the number of bytes transmitted in the network without in-network computation. Simulation Results with an accuracy constraint. (g) $|V| = 5, d = 5$. (h) $|V| = 30$. (i) $|V| = 200$.

```
$NV = \{0\}$, $h_v = \infty$, $\forall v \in V \backslash \{0\}$, $h_0 = 0$, $C_v = 0$, $\forall v \in V$.
($C_v$ denotes the number of children of node $v$.)
while $(NV != V)$ do
    For each edge $e \in E$ such that $e$ connects
    nodes $v \in NV$ and $v' \in V \backslash NV$ and $C_v < d-1$
        $h'_v = \min(h'_v, h_v + 1)$
    $v_{min} = \mathrm{argmin}_v\{h_v \mid \forall v \in V \backslash NV\}$
    Add $v_{min}$ to $NV$.
    Let the parent of $v_{min}$ be $v_{parent}$. Update
    $C_{v_{parent}} = C_{v_{parent}} + 1$
    Set $h_v = \infty$, $\forall v \in V \backslash NV$
```

Fig. 5. Algorithm A2: Modified Dijkstra's approximation algorithm for P2.

C. Discussion: Additional Constraints / Alternative Formulations

Accuracy Constraint: Global eigenvectors are determined by linearly combining the eigenvectors computed locally at different sensor nodes (in S). Now, if there is no noise in the system, then the global eigenvectors computed using this decomposition will exactly match the actual eigenvectors. However, the presence of noise in the sensed values can lead to errors in the computation [29]. And these errors will accumulate and propagate if the decentralized (or decomposed) method is used to compute the SVD. This is due to the fact that in a centralized implementation, a least-squares effect minimizes the error due to noise across all the eigenvectors, whereas the decentralized implementation allows this noise error to accumulate through each combination of locally computed eigenvectors.

Now, more the number of FFT's being combined at each sensor node (that is bigger the value of $|N_s|, s \in S$), smaller will be this error. Hence, a constraint on the desired accuracy will impose a constraint on the minimum number of FFT's being combined at each sensor node, that is, a constraint on the minimum value of $|N_s|, \forall s \in S$. We refer to this constraint as the accuracy constraint.

Incorporating this constraint in our algorithms is straightforward. Let the minimum value of $|N_s|, \forall s \in S$ imposed by the accuracy constraint be equal to d_a. (The value of d_a depends on the noise floor in the sensors as well as the accuracy desired by the application.) In the ILP presented in Section IV-A for P1, the following additional constraint is introduced: $\sum_{i \in V} x_{ij} \geq d_a x_{jj}, \forall j \in V$. Similarly, in the ILP presented in Section IV-B2 for P2, the following constraint is introduced to incorporate the accuracy constraint: $\sum_{e \in I_v} x_e \geq (d_a - 1) l_v, \forall v \in V$, where $l_v \in \{0, 1\}$ is an integer variable which is set to 1 if v is a non-leaf node. The following additional constraint ensures that l_v is set 1 only if v is non-leaf node: $\sum_{e \in I_v} x_e/|V| \leq l_v \leq \sum_{e \in I_v} x_e, \forall v \in V$. Finally, the two approximation algorithms proposed can be easily modified to maintain the number of children of each node in the data collection to be greater than $d_a - 1$. This extra constraint has no impact on the approximation factor of these algorithms as the fundamental intuition summarized in Section IV-B5 does not change.

Storage Constraint: As the number of FFT's being computed at a sensor node increases, not only the computational delay but also the storage required at that sensor increases [29]. Since, the available memory on each sensor is also limited, this storage constraint also bounds the maximum number of FFT's which can be combined at a sensor, that is, the maximum value of $|N_s|, \forall s \in S$ is bounded by the storage constraint. Since, the computational delay constraint also results in a similar constraint, the storage constraint can be incorporated in a manner similar to the computational delay constraint. Let d_s be the maximum value of $|N_s|$ being imposed by the storage constraint. Now, the value of d (defined before Definition 1) is defined to be $d := \min \{d_s, \max \{|N_s| \mid C(|N_s|) \leq C\}\}$. With this new definition for d, no other change is required in the proposed ILP's as well as the proposed approximation algorithms to incorporate the storage constraint.

Alternative Energy Models: The model presented in Section II-C to compute the total communication energy does not incorporate phenomenon like the energy expended in overhearing packets destined to other nodes etc. However, as long as the energy model is a linear function of the number of packet transmissions / bit transmissions occurring per node (which yields an accurate representation for most energy consumption models), the proposed algorithms can be directly applied without any change in their optimality / approximation factors.

V. Simulations

In this section, we evaluate the performance of the proposed approximation algorithms using simulations, and compare them to the performance of the optimal communication structure. We use CPLEX [30] to solve the ILPs. All our simulations are done on topologies generated by randomly distributing nodes in an area of $50 \times 50m^2$ and assuming the transmission range to be $30m$.

We will compare the energy consumed by the communication structures derived using the algorithms proposed in Section IV for different values of d. For the SVD computation, $R = 8192$ bytes and $r = 32$ bytes [4]. Figures 6(a) and 6(b) compare the lower bound on the number of bytes transmitted on the network (Lemma 1) to the number of bytes transmitted in the optimal communication structure derived by solving the ILP for P1 (Section IV-A), and the number of bytes transmitted in the communication structure derived using the ILP for P2 (Section IV-B2), algorithm A1 (Figure 4) and algorithm A2 (Figure 5) for different values of d, with $|V| = 4$ and $|V| = 6$ respectively. We observe that the approximation algorithms perform very close to the optimal.

It takes more than one hour of computation to solve the ILP for P1 for $|V| > 6$ on a 2.99 GHz machine with 4 GB of RAM. Hence, for larger values of $|V|$, we only compare the three approximation algorithms against the lower bound in Figures 6(c) and 6(d). We make the following two observations, (i) all approximation algorithms are within 3% of the optimal, and (ii) the algorithm A2 outperforms the algorithm A1. This simulation also demonstrates the advantage of using the ILP for P2 over the ILP for P1. Since the former has fewer variables and constraints, it runs much faster, and on the same machine, converges within an hour till $|V| \leq 40$.

For even larger values of $|V|$, we compare the performance of the algorithm A2 (as it consistently outperforms algorithm A1) against the lower bound in Figures 6(e) and 6(f). And we observe that it is always within 3% of the optimal. These results also demonstrate the advantage of in-network computation as the number of bytes transmitted over the network are reduced by more than half. Finally, note that Figures 6(e) and 6(f) demonstrate the trade-off between communication energy and computation delay. The more the computation delay allowed per node (larger the value of d), the smaller the energy consumed in the network.

In figures 6(g)-6(i), we compare the performance of the different approximation schemes after incorporating an accuracy constraint in the formulation for different values of $|V|, d$ and d_a. In this scenario, we observe that the ILP for P2 yields results within 5% of the optimal while algorithm A2 yields values within 40% of the optimal. And the advantage of using a better centralized algorithm becomes more pronounced as the value of d_a increases.

VI. Parallel Simulated Annealing

In a structural health monitoring system, a common technique to translate raw sensor data into an estimate of damage involves comparing system properties in an unknown state of health to those in a known, undamaged state [31], [32]. This technique is referred to as model updating and involves adjusting the system parameters iteratively in an analytical model such that the analytical system produces response data that matches results obtained experimentally. Using this method, damage can be detected in a system by periodically searching for changes in model parameters that can be linked directly to suboptimal system performance.

A wide variety of model updating techniques have been developed over the years [33]. One common approach is to define an objective function, E, which relates the difference between analytical and experimental data. This function can be repeatedly evaluated with varying values of the analytical model parameters until the difference between the analytical and experimental response is minimized.

Simulated annealing (SA) is one of the most common algorithms for stochastically searching for the global minimum of such an objective function. This method has been used frequently in model-based damage detection techniques [34]. Metropolis *et al.* [35] developed this algorithm to determine the global minimum energy state amidst a nearly infinite number of possible configurations. The Metropolis criterion expresses the probability of a new system state being accepted at a given system temperature, and can be stated as: accept the new state if and only if $E_{new} \leq E_{old} - T ln(U)$, where E is the value of the objective function for a given energy state, U is a uniformly distributed random variable between 0 and 1, and T is the temperature of the system. The addition of the $T ln(U)$ term allows the system to accept an invalid state in the hope of avoiding premature convergence to a local minima.

A standard SA algorithm begins the optimization process by assigning an initial temperature T_1, and letting the Metropolis algorithm run for N_1 iterations. During each iteration, certain analytical model parameters are reassigned in a pseudo-random fashion, and the objective difference between the experimental and analytical output is determined. This newly created state is either accepted or rejected based on the Metropolis criterion. After N_1 iterations, the temperature of the system is reduced to T_2 and the process runs for N_2 iterations. This process continues till the temperature drops to a really low temperature, T_M, where very few new states are accepted, and the system has, in essence, frozen. To summarize, the process runs for M temperature steps, and for $N_j, 1 \leq j \leq M$ iterations for each temperature step T_j.

Over the years, many parallel SA techniques have been developed and successfully implemented [36]. Zimmerman *et al.* [37] proposed a new parallel SA technique more suited to be implemented over a wireless sensing system for structural health monitoring as it reduces the communication required between processing nodes. This technique breaks up the traditionally serial SA tree (which is continuous across all temperature steps) into a set of smaller search trees, each of which corresponds to a given temperature step and begins with the global minimum values for the preceding temperature step. Each of these smaller trees can be assigned to a cluster of available nodes in the network, and thus can run concurrently.

As the parallelized search progresses, updated global state information has to be disseminated downwards (to the nodes doing the computations at lower temperatures) through the network. Specifically, when a node detects a new global minimum energy state at a given temperature, it communicates this information to the cluster-head of its cluster, which then propagates this information to all nodes doing the computation at lower temperatures. These nodes (computing at lower temperatures) will re-start their search based on this new state. This may seem wasteful at high temperatures, however, as the search algorithm converges on a solution, it becomes decreasingly likely that a new global minimum will be found at a given temperature step which reduces the total number of transmissions.

[37] explores the advantages of this approach in a wireless sensing system. However, it does not explore how to construct the communication structure so as to minimize the energy consumption which will be the focus of this section.

We now precisely state the problem. The designer will set the values of N_j and $k_j, 1 \leq j \leq M$, which denote the number of computations to be performed at temperature T_j and the number of sensor nodes performing the computation at T_j respectively. (Note that $\sum_{j=1}^{M} k_j \leq |V|$.) The values of N_j and k_j will be determined based on the accuracy and the computational constraint per node.

Given the values of N_j and k_j, determining the communication structure involves dividing the V nodes into M clusters each of size $k_j, 1 \leq j \leq M$ and choosing a cluster-head for each cluster. Let the cluster of nodes corresponding to temperature T_j be denoted by K_j. (Note that $|K_j| = k_j$.) Finally, let $b_j \in K_j$ denote the cluster-head for the cluster K_j. Any computation which results in a new minimum energy state at a temperature T_j requires exchanging this information between all nodes belonging to the cluster K_j, between the cluster-heads b_j and $b_l, l > j$, and all nodes belonging to clusters $K_l, l > j$. Thus, the total number of transmissions for each new minimum energy state found at temperature T_j is equal to $\sum_{v \in K_j} H_{b_j \to v} + \sum_{l=j+1}^{M} H_{b_j \to b_l} + \sum_{l=j+1}^{M} \sum_{v \in K_l} H_{b_l \to v}$, where recall that $H_{i \to j}, i, j \in V$ denotes the average number of transmissions required to exchange information between nodes i and j along the shortest path between the two nodes.

We first describe an ILP to determine the optimal communication structure for parallel simulated annealing. Let $x_{ij}, i \in V, 1 \leq j \leq M$ be an indicator variable which is set to 1 only if node $i \in K_j$. Let $y_{ij}, i \in V, 1 \leq j \leq M$ be another indicator variable which is set to 1 only if node $i = b_j$, that is, i is the cluster-head for K_j. Note that here we have a separate variable to denote the cluster-head whereas for the SVD computation, we merely set x_{ii} to 1 if node i was a cluster-head. The extra variable is needed for parallel simulated annealing to convert the quadratic objective into a linear equation. Let $t_{ikj}, i, k \in V, 1 \leq j \leq M$ denote an indicator variable which is set to 1 only if node i is the cluster-head for temperature T_j and node $k \in K_j$ (that is $t_{ikj} = y_{ij} x_{kj}$) and let $p_{ikj}, i, k \in V, 1 \leq j \leq M-1$ denote an indicator variable which is set to 1 only if node i is the cluster-head at temperature T_j and node k is the cluster-head at temperature T_{j+1} (that is $p_{ikj} = y_{ij} y_{k(j+1)}$). Finally, let $a_j, 1 \leq j \leq M$ denote the probability of generating a new minimum energy state per computation at temperature T_j. Then, for N_j computations at that temperature, the number of new minimum energy states generated are $a_j N_j$. Note that generating a new minimum energy state triggers new transmissions.

Following is the ILP to determine the optimal communication structure for parallel simulated annealing.

$$\sum_{j=1}^{M} a_j N_j \left(\sum_{i \in V} \sum_{k \in V} H_{i \to k} \left(t_{ikj} + \sum_{l=j+1}^{M} p_{ikl} + \sum_{l=j+1}^{M} t_{ikl} \right) \right) \tag{22}$$

$$\sum_{i \in V} x_{ij} = k_j, 1 \leq j \leq M \tag{23}$$

$$t_{ikj} \geq \frac{y_{ij} + x_{kj} - 1}{2}, i, k \in V, 1 \leq j \leq M \tag{24}$$

$$p_{ikj} \geq \frac{y_{ij} + y_{k(j+1)} - 1}{2}, i, k \in V, 1 \leq j \leq M-1 \tag{25}$$

$$x_{ij}, y_{ij}, t_{ikj}, p_{ikj} \in \{0,1\}, i, k \in V, 1 \leq j \leq M. \tag{26}$$

The first constraint (Equation (23)) ensures that the cluster performing computations at temperature T_j has k_j nodes while the next two constraints populate the values of $t_{ikj} = y_{ij} x_{kj}$ and $p_{ikj} = y_{ij} y_{k(j+1)}$.

We finally describe a greedy approximation algorithm to determine the communication structure for parallel simulated annealing. Recall that we need to determine the set of nodes which form a cluster as well the corresponding cluster-head for each temperature $T_j, 1 \leq j \leq M$. Figure 7 describes the greedy algorithm. We first start from the smallest temperature T_M because finding a new energy state at any temperature

```
$K = V$, $j = M$
while $(j > 0)$ do
    $minE = \infty$
    $\forall v \in K$
        $\left(e_v^j, k_v^j\right)$ = findMin $(K, v, j)$
        If $\left(MinE > e_v^j\right)$
            $MinE = e_v^j$, $K_j = k_v^j$, $b_j = v$
    $j = j - 1$, $K = K \backslash K_j$


findMin $(K, b, j)$
    $T = \phi$
    $\forall v \in S$
        $d_v = H_{v \to b}$
    Sort the nodes in $K$ in ascending order of $d_v$'s
    Add the first $k_j - 1$ nodes from this sorted list
    to $T$
    $E = \sum_{v \in T} H_{v \to b} + I_{j<M} H_{b_{j+1} \to b}$
    ($I_{j<M}$ is an indicator variable which is equal to
    1 if $j < M$, else it is equal to 0.)
    return $(E, T)$
```

Fig. 7. A greedy approximation algorithm to determine the communication structure for parallel simulated annealing.

will trigger a transmission between the cluster-head b_M and the nodes belonging to the cluster K_M. Amongst all the nodes $v \in V$, determine the cluster-head to be the node which has the smallest sum of the average number of transmissions required to get to k_M nodes. This yields both b_M and K_M. From amongst the remaining nodes, in a similar manner, greedily select b_{M-1} and K_{M-1} and continue. Assuming the maximum height of the unconstrained data collection tree (defined in Definition 2) is $O(log(|V|))$, using arguments similar to ones made in Section IV-B4, the approximation factor of the greedy approximation algorithm is also $O(log(|V|))$.

VII. Conclusions

This paper presents centralized, optimal ILPs and polynomial, distributed approximation algorithms to derive the communication structure with networked computing for any given computation. The approximation factor for each approximation algorithm we propose is derived analytically, and simulations are used to evaluate their performance for real engineering applications. For functions with only single stream input operators, the proposed approximation algorithm is always within 35% of the optimal while for functions with a multiple stream input operator, the proposed approximation algorithms are always within 3% of the optimal. Our results also demonstrate the advantage of in-network computation as it can reduce the number of bytes transmitted over the network by more than half.

Appendix A
A Brief Overview of Singular Value Decomposition

Let A be a real $m \times n$ matrix with $m \geq n$. Then, the singular value decomposition (SVD) factors A as follows: $A = U\Sigma V^T$ where $U^TU = V^TV = VV^T = I_n$ and $\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)$. The matrix U consists of n orthonormalized eigenvectors associated with the n largest eigenvalues of AA^T, and the matrix V consists of the orthonormalized eigenvectors of A^TA. The diagonal elements of Σ are the non-negative square roots of the eigenvalues of A^TA. We shall assume that $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n$.

SVD comprises of two steps [38]. The first step converts the matrix A into a bi-diagonal form, and then the second step uses a variant of the QR algorithm to iteratively diagonalize this bi-diagonal matrix.

A. Reduction to the bi-diagonal form

This step decomposes A as $A = PJ^{(0)}Q^T$, where P and Q are unitary matrices and $J^{(0)}$ is an $m \times n$ bi-diagonal matrix of the form

$$J^{(0)} = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & . & . & . & 0 \\ 0 & \alpha_2 & \beta_2 & 0 & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & . & . & . & 0 & \alpha_{n-1} & \beta_{n-1} \\ 0 & . & . & . & 0 & 0 & \alpha_n \end{bmatrix}.$$

Let $A = A^{(1)}$, and let $A^{(3/2)}, A^{(2)}, \ldots, A^{(n)}, A^{(n+1/2)}$ be defined as follows:

$$A^{(k+1/2)} = P^{(k)}A^k, \qquad k = 1, 2, \ldots, n,$$
$$A^{k+1} = A^{(k+1/2)}Q^{(k)}, \quad k = 1, 2, \ldots, n-1.$$

$P^{(k)}$ and $Q^{(k)}$ are hermitian, unitary matrices of the form

$$P^{(k)} = I - 2x^{(k)}x^{(k)T}, \quad x^{(k)T}x^{(k)} = 1,$$
$$Q^{(k)} = I - 2y^{(k)}y^{(k)T}, \quad x^{(k)T}x^{(k)} = 1,$$

The unitary transformation $P^{(k)}$ is determined so that $a_{i,k}^{(k+1/2)} = 0$, $i = k+1, \ldots, m$, and $Q^{(k)}$ is determined so that $a_{k,j}^{(k+1)} = 0$, $i = k+2, \ldots, n$. Solving these set of linear equations sequentially yields $P, J^{(0)}$ and Q.

B. SVD of the bi-diagonal matrix

The matrix $J^{(0)}$ is iteratively diagonalized so that $J^{(0)} \to J^{(1)} \to \ldots \to \Sigma$, where $J^{(i+1)} = S^{(i)T}J^{(i)}T^{(i)}$, and $S^{(i)}, T^{(i)}$ are orthogonal. The matrices $T^{(i)}$ are chosen such that the sequence $M^{(i)} = J^{(i)T}J^{(i)}$ converges to a diagonal matrix while the matrices $S^{(i)}$ are chosen such that all $J^{(i)}$ are of the bi-diagonal form.

We now describe how to derive $\{S^{(i)}\}$ and $\{T^{(i)}\}$. For notational convenience, we drop the suffix and use the notation: $J \equiv J^{(i)}, \overline{J} \equiv J^{(i+1)}, S \equiv S^{(i)}, T \equiv T^{(i)}, M \equiv J^TJ, \overline{M} \equiv \overline{J}^T\overline{J}$.

The transition $J \to \overline{J}$ is achieved by the application of Givens rotations to J alternately from the right and the left. Thus, $\overline{J} = S_n^TS_{n-1}^T \ldots S_2^TJT_2T_3 \ldots T_n$, where $S^T = S_n^TS_{n-1}^T \ldots S_2^T$, $T =$

$T_2 T_3 \dots T_n$,

$$S_k = \begin{bmatrix} 1 & 0 & & & & & & & 0 \\ 0 & \cdot & & & & & & & \\ & & \cdot & & & & & & \\ & & & \cdot & & & & & \\ & & & cos\theta_k & -sin\theta_k & & & & \\ & & & sin\theta_k & cos\theta_k & & & & \\ & & & & & 1 & & & 0 \\ & & & & & & \cdot & & \\ & & & & & & & \cdot & \\ & & & & & & & & \cdot \\ 0 & & & & & & & 1 & 0 \\ 0 & & & & & & & & 1 \end{bmatrix},$$

the $(k-1)\times(k-1)^{th}$, $(k-1)\times k^{th}$, $k\times(k-1)^{th}$ and $k\times k^{th}$ elements of S_k are $cos\theta_k$, $-sin\theta_k$, $sin\theta_k$ and $cos\theta_k$ respectively, and T_k is defined analogously to S_k with ψ_k instead of θ_k.

Let the first angle ψ_2 be chosen arbitrarily while all the other angles are chosen so that $\overline{J}$ has the same form as J. Thus,

$$\begin{gathered} T_2 \text{ annihilates nothing, generates as entry } \{J\}_{21}, \\ S_2^T \text{ annihilates } \{J\}_{21}, \text{ generates as entry } \{J\}_{13}, \\ T_3 \text{ annihilates } \{J\}_{13}, \text{ generates as entry } \{J\}_{32}, \\ \cdot \\ \cdot \\ \cdot \\ S_n^T \text{ annihilates } \{J\}_{n,n-1}, \text{ generates nothing.} \end{gathered}$$

What now remains is to define how to choose the first angle ψ_2. It is chosen such that the transition $M \rightarrow \overline{M}$ is a QR transformation with a given shift s. The usual QR algorithm with shifts [39] is described as: $M - sI = T_s R_s$, $R_s T_s + sI = \overline{M}_s$. This shift parameter s is determined by an eigenvalue of the lower 2×2 minor of M, and T_2 is chosen such that its first column is proportional to that of $M - sI$ which yields the value of ψ_2.

APPENDIX B
PROOF OF THEOREM 1

First, the decision version of our problem is in NP. Given a communication structure, computing the energy consumed at each node and checking if the constraints specified in Equations (3)-(11) are satisfied can be done in polynomial time. Hence, testing feasibility as well as testing if the total cost is less than a given value M is accomplished in polynomial time.

Next, to prove NP-hardness, we perform a reduction from the set cover problem [40], whose decision version is defined as follows.

Definition 4 (Set Cover): Given a collection C of subsets of a finite set P and an integer $0 < K \leq |C|$, with $|C|$ the cardinality of C, does C contain a subset of $C' \subset C$ with $|C'| \leq K$, such that every element of P belongs to at least one of the subsets in C' (this is called a set cover from P)?

For any instance of the set cover problem, we build an instance of the decision version of the problem P1. Figure 8 illustrates the construction of the graph instance of the problem. The resulting graph is formed of three layers: the base station V_0, a layer corresponding to the subsets $C_k \in C$, and a layer corresponding to the elements $\{p_j\} \in P$. For each element $C_k \in C$, we build a structure formed by $|C_k|+3$ nodes as in Figure 8(b) (each subset C_k has its own such graph, and nodes, but we drop the subscript k to simplify presentation). The node x_3 is connected to the base station V_0, nodes x_1 and x_2 are connected to x_3 and the weight of the corresponding edges is 1 and $1 < a < d$ respectively. The rest of the $|C_k|$ nodes are connected to both x_1 and x_2, and the weight of each of these edges is $d > 0$. Finally, x_1 and x_2 are connected with an edge of weight d also.

Furthermore, we connect each structure $C_k \in C$ (namely the node x_1 from that structure) to only those nodes in the P layer that correspond to elements contained in C_i (example: in the instance in Figure 8(a), subset $C_1 = \{p_1, p_2, p_3\}$ etc). All the edges connecting the P layer to the C layer also have a weight equal to d. Finally, all nodes x_3 are inter-connected with an edge of weight 0 as well as connected to the base station V_0 with an edge of weight 0. Nodes which do not have an edge between them are not connected. Finally, recall that R and r denote the number of bits required to represent the FFT from a single sensor and a single eigenvector respectively.

Next, we define the computational delay constraint on each node. No more than $|C_k|+1$ FFT's can be combined on nodes x_1 and x_2 for a given C_k. At nodes x_3 of all C_k's, no more than 4 FFT's can be combined. There is no computational delay constraint on the base station as well as on the nodes in the P layer.

The goal is to build a communication structure for which the energy cost is at most M while respecting the computational delay constraints at each node and the combinability constraint for each computation. We will next show that if $M = dR(|P| + \sum_k |C_k|) + R|C| + aR|C| + ar(|P|+K) + r\sum_k(|C_k|+1)$, for the positive integer $K \leq |C|$, then solving the problem P1 is equivalent to finding a *Set Cover* of cardinality K or less for the set P. Notice that the construction of our graph instance from the set cover can be performed in polynomial time.

For $d > (R|C| + aR|C| + ar(|P|+K) + r\sum_k(|C_k|+1))/R$, the communication structure for the problem P1 will have transmissions on exactly $|P|$ edges between the layers P and C, and on exactly $|C_k|$ edges in the structure shown in Figure 8(b) for every $C_k \in C$. That means no other node than x_1,x_2 and x_3 will be used as a relay. That is, only x_1,x_2 and x_3 can belong to S. If some other node belongs to S, then the cost of the communication structure would contain R bits passing through more than $|P| + \sum_k(|C_k|)$ edges which would result in a cost larger than M. This also implies that x_1 and x_3 for all $C_k \in C$ belong to S. The only degree of freedom is whether x_2 lies in S or not. (Recall that $x_2 \in S$ only if a SVD computation takes place on x_2 also.)

The key idea of our proof is that for $1 < a < d$, finding a communication structure with cost at most M means connecting the nodes in layer P to at most K nodes of layer C. If the structure needs to connect the layer P to more than K nodes in C, the cost of the communication structure will necessarily be higher than M. The intuition is that node $x_2 \in S$ if and only if the corresponding C_k is connected to the P layer. Then, if

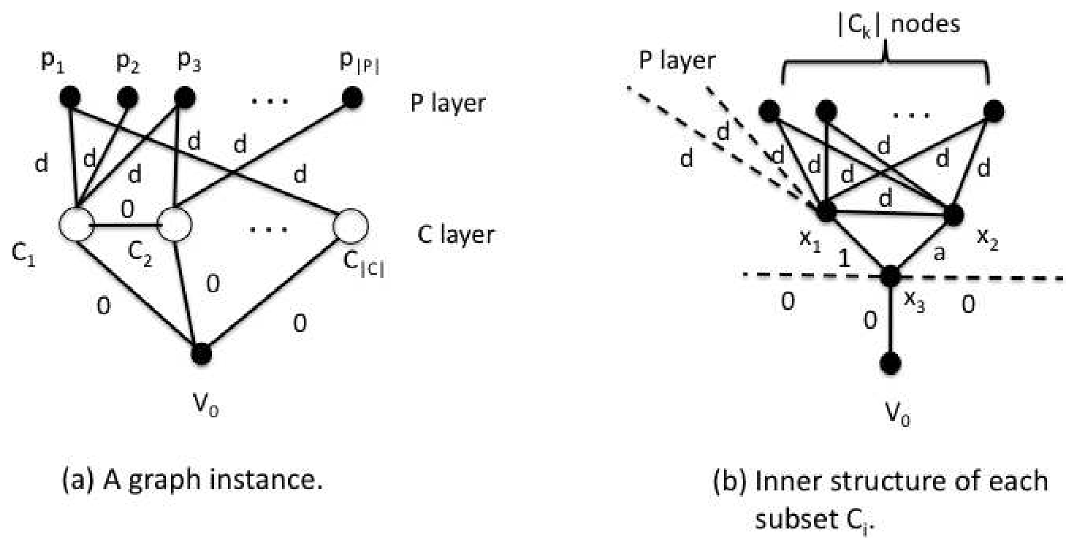


Fig. 8. Instance of the problem P1 for any given instance of the set cover problem.

the number of x_2 nodes in S is more than K, the cost of the communication structure will be more than M.

We first show that if a corresponding C_k is not connected to the P layer, then its corresponding x_2 will not belong to S. In such a scenario, the optimal communication structure is to have all the other $|C_k|$ nodes (other than x_1, x_2 and x_3) send their data to x_1 (since $a > 1$, transmitting everything to x_1 instead of x_2 will consume less energy) who will then compute the SVD and send the corresponding eigenvectors as well as its own FFT to x_3. x_3 will receive the FFT from x_1, x_2 and the x_3 node of $C_{(k+1)mod|C|}$ (the final communication ensures combinability and has an energy cost of 0). It will forward all the computed eigenvectors to the base station. The total energy consumed in this operation is $d|C_k|R + aR + R + r(|C_k|+1)$.

We next show that if a corresponding C_k is connected to the P layer, then its corresponding x_2 will always belong to S. Since no more than $|C_k|+1$ FFT's can be combined on x_1, if t_k of the p_j nodes in the P layer send their FFT to x_1, then x_1 can combine FFT's from no more than $|C_k| - t_k$ nodes belonging to the structure of C_k. The remaining t_k nodes will have to send their FFT to x_2 as it has the next smallest distance (after x_1) to these nodes. Thus, x_2 will combine data from $t_k + 1$ nodes. The energy consumed in this scenario is $d|C_k|R + dt_kR + R + aR + ar(t_k+1) + r(|C_k|+1)$. (Note that $\sum_{C_k} t_k = P$.)

Thus, if the number of structures in the P layer connected to the C layer is equal to K', then the total energy consumed is equal to $dR(|P| + \sum_k |C_k|) + R|C| + aR|C| + ar(|P| + K') + r\sum_k(|C_k|+1)$ which will be larger than M if $K' > K$. This means that finding a communication structure with a cost at most M implies finding a set of K elements or less from the C layer to which all the nodes in set P connect. In other words, a communication structure with a cost of at most M yields a set cover of size at most K.

Now, we need to ensure that the set cover of size at most K also yields a communication structure of cost at most M. From the previous discussion, it is obvious that by connecting the nodes in P to those nodes in the C layer which belong to the set cover such that all nodes in set P are connected yields an energy cost of no more than M. The computational delay constraint is also obviously satisfied at all nodes. We merely need to ensure that all computations are combinable. Since each node x_3 belonging to the structure of C_k send its FFT to the node x_3 belonging to the structure of node $C_{(k+1)\text{mod}|C|}$, all computations are combinable.

Thus our decision problem is NP-complete and our optimization problem is NP-hard.

References

[1] S. Fang and R. Perera, "Power mode shapes for early damage detection in linear structures," *Journal of Sound and Vibration*, vol. 324, no. 1-2, pp. 40–56, 2009.

[2] Z. Ismail, H. Razak, and A. Rahman, "Determination of damage location in RC beams using mode shape derivatives," *Engineering Structures*, vol. 28, no. 11, pp. 1566–1573, 2006.

[3] E. Clayton, B. Koh, G. Xing, C. Fok, S. Dyke, and C. Lu, "Damage detection and correlation-based localization using wireless mote sensors," in *IEEE International Symposium on Intelligent Control*, 2005.

[4] A. Zimmerman, M.Shiraishi, R. Swartz, and J. Lynch, "Automated Modal Parameter Estimation by Parallel Processing within Wireless Monitoring Systems," *Journal of Infrastructure Systems*, vol. 14, no. 1, pp. 102–113, 2008.

[5] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "TAG: a Tiny AGgregation service for ad-hoc sensor networks," in *Proc. of OSDI*, 2002.

[6] B. Patt-Shamir, "A note on efficient aggregate queries in sensor networks," in *Proc. of ACM PODC*, 2004.

[7] S. Kashyap, S. Deb, K. Naidu, and R. Rastogi, "Efficient gossip-based aggregate computation," in *Proc. of ACM PODS*, 2006.

[8] X. Li, Y. Wang, and Y. Wang, "Complexity of convergecast and data selection for wireless sensor networks," Illinois Institute of Technology, Tech. Rep., 2009.

[9] A. Giridhar and P. R. Kumar, "Towards a Theory of In-Network Computation in Wireless Sensor Networks," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 98–107, 2006.

[10] D. Friedlander, C. Griffin, N. Jacobson, S. Phoha, and R. Brooks, "Dynamic agent classification and tracking using an ad hoc mobile acoustic sensor network," *EURASIP Journal on Applied Signal Processing*, vol. 2003, no. 4, pp. 371–377, 2003.

[11] J. Gupchup, R. Burns, A. Terzis, and A. Szalay, "Model-Based Event Detection in Wireless Sensor Networks," in *Proc. of Workshop on Data Sharing and Interoperability on the World-Wide Sensor Web*, 2007.

[12] L. Balzano and R. Nowak, “Blind Calibration of Sensor Networks,” in *Proc. of IPSN*, 2007.

[13] G. Derveaux, G. Papanicolaou, and C. Tsogka, “Time reversal imaging for sensor networks with optimal compensation in time,” *The Journal of the Acoustical Society of America*, vol. 121, no. 4, pp. 2071–2085, 2007.

[14] V. Raykar, I. Kozintsev, and R. Lienhart, “Position Calibration of Microphones and Loudspeakers in Distributed Computing Platforms,” *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 1, 2005.

[15] R. Cristescu, B. Beferull-Lozano, M. Vetterli, and R. Wattenhofer, “Network correlated data gathering with explicit communication: NP-completeness and algorithms,” *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 41–54, 2006.

[16] D. Ewins, *Modal testing: Theory and practice*, 2nd ed. Research Studies Press Ltd.

[17] O. Salawu, “Detection of structural damage through changes in frequency: A review,” *Engineering Structures*, vol. 19, no. 9, pp. 718–723, 1997.

[18] R. Brincker, L. Zhang, and P. Andersen, “Modal identification of output-only systems using frequence domain decomposition,” *Smart Materials and Structures*, vol. 10, no. 3, pp. 441–445, 2001.

[19] Y. Zhu, K. Sundaresan, and R. Sivakumar, “Practical Limits on Achievable Energy Improvements and Useable Delay Tolerance in Correlation Aware Data Gathering in Wireless Sensor Networks,” in *Proc. of IEEE SECON*, 2005.

[20] S. Pattem, B. Krishnmachari, and R. Govindan, “The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks,” in *Proc. of IPSN*, 2004.

[21] R. Newton, S. Toledo, L. Girod, H. Balakrishnan, and S. Madden, “Wishbone: Profile-based Partitioning for Sensornet Applications,” in *Proc. of NSDI*, 2009.

[22] B. Brinkman and M. Helmick, “Degree-constrained minimum latency trees are apx-hard,” Miami University, Tech. Rep. 2008-03-25, 2008.

[23] M. Helmick and F. Annexstein, “Depth-Latency Tradeoffs in Multicast Tree Algorithms,” in *Proc. of The IEEE 21st International Conference on Advanced Information Networking and Applications*, 2007.

[24] V. Vazirani, *Approximation Algorithms*, 1st ed. Springer.

[25] D. Aguuayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, “Link-Level Measurements from an 802.11b Mesh Network,” in *Proc. of ACM SIGCOMM*, 2004.

[26] J. Zhao and R. Govindan, “Understanding packet delivery performance in dense wireless sensor networks,” in *Proc. of ACM SenSys*, 2003.

[27] W. Wu, H. Du, X. Jia, Y. Li, and S.-H. Huang, “Minimum Connected Dominating Sets and Maximal Independent Sets in Unit Disk Graphs,” *Theoretical Computer Science*, vol. 352, no. 1, pp. 1–7, 2006.

[28] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, “Collection Tree Protocol,” in *Proc. of ACM SenSys*, 2009.

[29] A. Zimmerman and J. Lynch, “Market-based frequency domain decomposition for automated mode shape estimation in wireless sensor networks,” *Journal of Structural Control and Health Monitoring*, 2010.

[30] “ILOG CPLEX 11.0,” http://www.ilog.com/products/cplex.

[31] S. Doebling, C. Farrar, and M. Prime, “A Summary Review of Vibration-Based Damage Identification Methods,” *The Shock and Vibration Digest*, vol. 30, no. 2, pp. 91–105, 1998.

[32] A. Teughels, J. Maeck, and G. Roeck, “Damage assessment by FE model updating using damage functions,” *Computers and Structures*, vol. 80, no. 25, pp. 1869–1879, 2002.

[33] J. Mottershead and M. Friswell, “Model Updating In Structural Dynamics: A Survey,” *Journal of Sound and Vibration*, vol. 167, no. 2, pp. 347–375, 1993.

[34] R. Levin and N. Lieven, “Dynamic finite element model updating using simulated annealing and genetic algorithms,” *Mechanical Systems and Signal Processing*, vol. 12, no. 1, pp. 91–120, 1998.

[35] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, “Equation of State Calculations by Fast Computing Machines,” *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[36] D. Greening, “Parallel simulated annealing techniques,” *Physica D*, vol. 42, pp. 293–306, 1990.

[37] A. Zimmerman and J. Lynch, “A Parallel Simulated Annealing Architecture for Model Updating in Wireless Sensor Networks,” *EEE Sensors Journal*, vol. 9, no. 11, pp. 1503–1510, 2009.

[38] G.H.Golub and C.Reinsch, “Singular Value Decomposition and Least Squares Solution,” *Numerical Mathematics*, vol. 14, pp. 403–420, 1970.

[39] J. Francis, “The QR transformation: A unitary analogue to the LR transformation,” *The Computer Journal*, vol. 4, pp. 265–271, 1961.

[40] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co.