Efficient Gathering of Correlated Data in Sensor Networks

Himanshu Gupta, Vishnu Navda, Samir Das, and Vishal Chowdhary
State University of New York, Stony Brook

In this paper, we design techniques that exploit data correlations in sensor data to minimize communication costs (and hence, energy costs) incurred during data gathering in a sensor network. Our proposed approach is to select a small subset of sensor nodes that may be sufficient to reconstruct data for the entire sensor network. Then, during data gathering only the selected sensors need to be involved in communication. The selected set of sensors must also be connected, since they need to relay data to the data-gathering node. We define the problem of selecting such a set of sensors as the *connected correlation-dominating set* problem, and formulate it in terms of an appropriately defined correlation structure that captures general data correlations in a sensor network.

We develop a set of energy-efficient distributed algorithms and competitive centralized heuristics to select a connected correlation-dominating set of small size. The designed distributed algorithms can be implemented in an asynchronous communication model, and can tolerate message losses. We also design an exponential (but non-exhaustive) centralized approximation algorithm that returns a solution within $O(\log n)$ of the optimal size. Based on the approximation algorithm, we design a class of centralized heuristics that are empirically shown to return near-optimal solutions. Simulation results over randomly generated sensor networks with both artificially and naturally generated data sets demonstrate the efficiency of the designed algorithms and the viability of our technique – even in dynamic conditions.

Categories and Subject Descriptors: C.2.4 [**DistributedSystems**]: —*Distributed Applications*

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Correlated Data, Topology Control, Energy Efficiency

1. INTRODUCTION

Advances in embedded processing and wireless networking have made possible creation of sensor networks [Estrin et al. 2000; Badrinath et al. 2000]. A sensor network consists of sensor nodes with short-range radios and on-board processing capability, forming a multi-hop network of irregular topology. Sensor nodes must be powered by small batteries, making energy efficiency a critical design goal. There has been a significant interest in designing algorithms, applications, and network protocols to reduce energy usage of sensors. Examples include energy-aware routing [Intanagonwiwat et al. 2000], energy-efficient information processing [Estrin et al.

Contact Address: Himanshu Gupta, Computer Science Department, SUNY at Stony Brook, Stony Brook, NY-11794. Email: hgupta@cs.sunysb.edu
Note: Preliminary version of the paper appeared in ACM MobiHoc 2005.

2000; Chu et al. 2002], and energy-optimal topology construction [Wattenhofer et al. 2001]. In this article, we focus on designing techniques to conserve energy by exploiting existing data correlations in the sensor network. The techniques developed in this paper are orthogonal to some of the other mentioned approaches, and can be used in combination with them to conserve energy.

The sensors are usually deployed in a redundant fashion as the deployment cost is high relative to the hardware cost of individual sensor nodes. Redundant deployment also allows sampling of a signal at very fine spatial resolution and/or to improve fault tolerance and noise immunity. But for ordinary usage, a dense sensor network simply presents data rich in correlation which we exploit in our techniques.

Our targeted applications are those that need to sense or sample a signal over the geographic region represented by the sensor network. Such sensor network applications have two types of nodes: *sensing* nodes and *data-gathering* nodes [Chou et al. 2003]. The data-gathering node gathers periodic snapshots of signal data values measured at the individual sensors and use interpolation to derive the signal value at all points in the geographic region. We exploit the correlations in the sensor data by selecting a small subset of sensor nodes called *correlation-dominating set* whose signal data values will be sufficient to derive the signal value at all points with sufficient accuracy. The selected correlation-dominating set of sensors should also form a connected communication graph, since they need to relay data to the data-gathering node using a spanning communication tree. In this article, we design a correlation structure that captures general data correlation relationships among sensor nodes in a sensor network, and formulate the connected correlation-dominating set problem in terms of a hypergraph describing the correlation structure. We design various distributed and centralized algorithms for computing a small connected correlation-dominating set. Using extensive simulations on artificially and naturally generated data, we show that the energy savings achieved by the above described approach is substantial.

Our work is complementary to the approaches that use compression techniques to exploit correlation in order to reduce the total amount of data transmitted [Chou et al. 2003; Marco et al. 2003; Cristescu and Vetterli 2003; Cristescu et al. 2004]. These techniques still require *all* sensor nodes to transmit their data. Interestingly, Marco et al [2003] show that compression schemes are of limited use for very dense sensor networks. In particular, Marco et al [2003] show that any compression scheme is *insufficient* to transport required amount of data for a given accuracy, when the density of the sensor network increases to infinity. A necessary fallout of this thesis is that "oversampling" beyond network capacity is possible for a sufficiently dense sensor network, and the only way to prevent this would be to suppress data transmission by some nodes. This provides credence to approaches such as ours that select only a subset of sensors for data transmission.

The rest of the paper is organized as follows. In the next section, we motivate and formally define the problem of connected correlation-dominating set. In the following two sections, we present the designed distributed and centralized algorithms respectively. Section 5 presents our simulation results. We end with sections on related work and conclusions.

2. MOTIVATION AND PROBLEM FORMULATION

In this section, we motivate the problem addressed in the article through an application and an example, and give a formal definition of the problem. We start with presenting our sensor network model.

A sensor network consists of a large number of sensors distributed randomly in a geographical region. Each sensor has a unique ID, and a radio interface, which is used to communicate directly with some of the sensors around it. A sensor s is said to be *correlated* to a set of sensors S if the data measured by s can be inferred/computed from the data measured by the sensors of S within an acceptable error bound as defined by the application. Such correlations can be discovered by prior data analysis (as described later).

2.1 Motivating Application and Example

In this article, we focus on data-gathering applications [Chou et al. 2003], where data-gathering nodes are responsible for gathering periodic snapshots of sensor data of interest. All sensor nodes transmit their measured data of interest to the data-gathering node upon being queried. The focus of this article is to exploit inherent data correlations and reduce the number of sensors that need to transmit data. For example, if a sensor s is correlated to a set of sensors S and each sensor in S is transmitting its data to the data-gathering node, then s need not transmit its data to the data-gathering node. Such suppression of transmissions enables gathering of snapshots with lower communication without compromising much on the data quality.

Our article addresses the following optimization problem (formally defined later) that arises in sensor networks with data correlations. Given a sensor network, select a minimum set of sensors M, called *connected correlation-dominating set*, such that (a) each sensor that is not in M is correlated to a subset of sensors in M, and (b) the selected set of sensors M forms a connected communication graph. The requirement for connectivity in the communication graph is due to the fact that the selected sensor set needs to collectively relay data to the data-gathering node. We choose the set M of smallest set since we assume the communication cost (number of messages) incurred in gathering data from M is equal to $|M|$ (see Section 2.2).

It is conceivable that if the sensor data values are rich in correlations, then $|M|$ could be very small compared to n, the total number of nodes in the sensor network. To develop a complete technique based on the above idea, we need to first discover correlations in the sensor data, and then, exploit the data correlations effectively to select a small set of sensors M that forms a connected communication graph and is sufficient to infer data of all the n sensors in the sensor network. The data is relayed to the data-gathering node over a communication tree spanning over M using one message per node per snapshot. Now, if the application is required to gather q snapshots, and D is the total communication cost incurred by a distributed algorithm for computing the set M, then the condition $(D + q|M|) < qn$ would ensure overall energy cost savings. Moreover, a small D (low communication-cost of the distributed algorithm), large q (long running data-gathering queries), and $n >> |M|$ (high degree of data correlation and good solution quality of the algorithm)

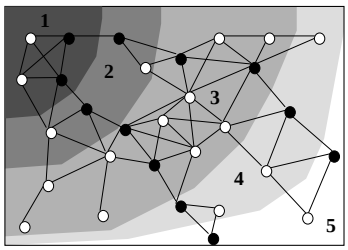


Fig. 1. Connected Correlation-Dominating Set Problem. Here, the set of black nodes form the connected correlation-dominating set.

will ensure $D << q(n - |M|)$ and hence, significant overall energy savings.

EXAMPLE 1. Consider the sensor network in Figure 1. The sensor network region has been divided into five regions numbered 1 to 5. Each region is shaded differently and has highly correlated signal values. Thus, sensors in the same region are measuring very similar signal data values. Let us assume that the data correlations between the sensor nodes can be represented by a simple rule: for a given region, any two sensor data values are sufficient to infer the data values of all other sensors in the region. This correlation structure could be represented by a correlation hypergraph where every sensor node s has a hyperedge $((s_i, s_j), s)$ incident on it for every pair of sensors nodes s_i and s_j that belong to the same region as s.

Let M be the set of dark nodes in the figure. Following the above mentioned data correlation rule, it is easy to see that the sensors in M are sufficient to infer the signal data of all the sensors in the sensor network, as M contains at least two nodes from every region. Since, the set M also forms a connected communication graph, it is a connected correlation-dominating set. Note that M contains 4 nodes from region 3 to ensure connectivity even though only 2 are sufficient to infer all others in the region. In this example, the total number of sensors is 30, while the size of the connected correlation-dominating set is only 12. □

2.2 Formal Problem Definition

We now formally define the connected correlation-dominating set problem addressed in this article. We start with a few definitions.

DEFINITION 1. (Communication Graph; Communication Distance) Given a sensor network consisting of a set of sensors I, the *communication graph* for the sensor network is the undirected graph CG with I as the set of vertices and an edge between any two sensors if they can communicate directly with each other. The *communication subgraph induced* by a set of sensors M is the subgraph of CG involving only the vertices/sensors in M.

The *communication distance* between two sensors I_1 and I_2 is the length of the shortest path between I_1 and I_2 in the communication graph. □

DEFINITION 2. (Correlation Graph; Correlation Neighbors) Given a sensor network consisting of a set of sensors I, the *correlation graph* over the sensor nodes is

a directed hypergraph with I as the set of vertices, and a subset of $(P(I) \times I)$ as the set of directed hyperedges, where $P(I)$ is the power set of I. In other words, the correlation graph is a hypergraph $G(V = I, E \subseteq (P(I) \times I))$. An edge (S, s) (where $s \notin S$) in the correlation graph signifies that the sensor data of the node s is correlated to the data of the set of sensors S and hence, the data of s can be computed (within some[1] error bound) from the data of the sensors in S. The correlation graph for the sensor network in Example 1 will have a hyperedge $((s_i, s_j), s_k)$ for any three sensors s_i, s_j, and s_k that belong to the same region (1 to 5).

A hyperedge in the correlation graph is also referred to as *correlation edge*. In a correlation edge (S, s), s is the *sink node*, S is the *source set*, and for any $x \in S$, s and x are called *correlation neighbors*. We assume that in a hyperedge (S, s), $s \notin S$. □

Definition 3. (Connected Correlation-Dominating Set) Consider a sensor network consisting of n sensors. Let C be the correlation graph over the sensor nodes in the network. A set of sensors M is called a *connected correlation-dominating set* if the following two conditions hold:

(1) For each sensor node $s \notin M$, there is a set of sensors $S \subseteq M$ such that (S, s) is a correlation edge in C.
(2) The communication subgraph induced by M is connected, and M contains the data-gathering node.

A set of sensors that satisfies only the first condition is called a *correlation-dominating set* in the network. □

Cost of Data Gathering. Let M' be a correlation-dominating set (not necessarily connected) in the given network graph G. Gathering of data from M' at a data-gathering node I requires first constructing a data-gathering tree T that spans M' and is rooted at I. Once such a tree T is available, each node in T collects data from all its descendants in T, and then, transmits the collected data to its parent. The communication cost (number of messages) incurred in the above data gathering (per snapshot) is equal to the number of nodes in T, since each tree node transmits exactly once. Here, we assume that the collective size of the data of descendants at any node is bounded, and can be packaged in one (or a constant number of) packet(s). We discuss relaxation of the above assumption at the end of Section 3. Thus, the cost of gathering data from a set of nodes M' is equal to the size of a tree T that spans M' and is rooted at the data-gathering node. Note that the set of nodes of the tree T forms a connected correlation-dominating set. Moreover, any connected correlation-dominating set M can be used to gather data from its contained correlation-dominating set at a cost of $|M|$. Thus, in order to minimize the cost of data gathering from some correlation-dominating set, we should select a *connected* correlation-dominating set of minimum *size*.

Connected Correlation-Dominating Set Problem. Given a sensor network and a correlation graph over the sensors, the connected correlation-dominating set

[1]The choice of error bound depends on the accuracy and energy-efficiency requirements of the application. If the error bound allowed is higher, the correlation graph will have more number of correlation edges, which will result in a more energy-efficient solution.

problem is to find the smallest connected correlation-dominating set.

The connected correlation-dominating set problem is NP-hard as the less general minimum dominating set problem is well known to be NP-hard [Guha and Khuller 1998]. Constructing a minimum connected correlation-dominating set enables an energy- efficient gathering of sensor data of interest in a sensor network with data correlations.

2.3 Distributed Computation of Correlation Graph

In this subsection, we briefly describe how the correlation graph of a sensor network is constructed by piggybacking additional messages over the normal data-gathering messages, and adding correlation hyperedges. We start with describing when a correlation edge is added, and how are the associated correlation parameters computed.

Computing Correlation Hyperedge Parameters. We use the least squares (LS) approach to determine existence of a hyperedge. In particular, we draw a hyperedge (S, s) from a set of nodes S to a node s, if the data readings of sensor s can be inferred from the readings of sensors in S within a certain bound. Let $x[k]$ and $x'[k]$ denote the actual and predicted values of a sensor x at k^{th} time instant. Let $S = \{s_1, s_2, \ldots, s_L\}$. We choose to use a linear predictive model to model the correlation, i.e., we predict $s'[k]$ to be a fixed linear combination of $s_1[k], s_2[k], \ldots, s_L[k]$ for all k. More formally,

$$s'[k] = \sum_{l=1}^{L} \alpha_l s_l[k],$$

where α_l are weighting coefficients. The above equation can be easily generalized to handle temporal correlations as well. A similar model has been used to model correlation in prior work [Chou et al. 2003]. We use the LS approach to minimize the error between the predicted and actual readings, and draw an hyperedge (S, s) if the minimum error is within a certain application-dependent bound. In particular, the weighted coefficients are chosen to minimize the least square error

$$E(\alpha) \; = \; \sum_{k=1}^{K} (s[k] - s'[k])^2, \tag{1}$$

where K is the number of samples, and the weighted coefficients are given [Kay 1998; Chou et al. 2003] by

$$[\alpha_1, \alpha_2, \ldots, \alpha_L]^T \; = \; (\mathbf{S}^{\mathrm{T}}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}}\mathbf{s}. \tag{2}$$

Here, $\mathbf{s} = [s[1], s[2], \ldots, s[K]]^T$ are the actual readings of the node s, and $\mathbf{S}$ is the $K \times L$ $(K > L)$ matrix of full rank L representing the actual readings of sensors in S, over time instants 1 to K. Equation (2) can be executed on an individual sensor node within affordable energy cost for reasonable values of K and L, and the energy cost expended in computing Equation (2) is proportional to K^2L^6. For instance, for $L = 3$ and $K = 5$, some profiling shows that the above matrix equation uses around 100,000 CPU instructions on Atmel 128L micro-controller used in Berkeley motes [Hill et al. 2000]. In addition, from power profiling data in [Shnayder et al.

2004], we can estimate that energy used is transmitting a single default-sized (30 bytes) message on TinyOS [D. Culler et al. 2004] at the maximum power is the same as that used in computation the above Equation (2) 6 times.

Computing Correlation Graph. Let $N(s, d)$ denote the set of d-hop neighbors of s in the communication graph of the sensor network. To compute *all* the correlation edges a node s is involved in, each node s in the sensor network should collect sufficient (say, K) *samples* from each node in $N(s, d)$, where d is sufficiently large to capture all data correlations. Initially, when the correlation structure is unknown, *all* the network nodes are periodically involved in transmitting data to the data-gathering node using a communication tree. Thus, we can collect samples (K from each node) from d-hop neighbors at each node by piggybacking over data-gathering messages for d snapshots as follows. Let us assume that each node has collected samples from all i-hop neighbors after i snapshots. The inductive step is as follows. During the $(i + 1)^{st}$ snapshot, when a node v is transmitting the data-gathering message to its parent, it (a) piggybacks the i-hop neighbors' samples it has already collected, and (b) instead of unicast transmission uses a broadcast transmission. Thus, *all* of v's 1-hop neighbors receive v's i-hop neighbors' samples. Since, the above piggybacking is also done by all the 1-hop neighbors of v during the i^{th} snapshot, the node v would have collected i-hop neighbors' samples of all its 1-hop neighbors which is equivalent to samples from all its $(i + 1)$-hop neighbors.

In the above process, we have implicitly assumed that the collective size of the i-hop neighbors' samples is bounded (since, d is expected to be small due to locally spatial data correlations), and hence, can be packaged/piggybacked in one transmission. In general, if the collective samples (K from each node) from $(d - 1)$-hop neighborhood need x packet messages, then we need to use the above piggybacking strategy for xd snapshots or use additional (at most ndx) messages. The number of piggybacked snapshots required can be reduced by using the strategy while the original K samples are being gathered.

3. ENERGY-EFFICIENT DISTRIBUTED ALGORITHM

In this section, we present a set of energy-efficient distributed algorithms to select a connected correlation-dominating set in a sensor network. We start with a description of the basic distributed algorithm. In the later paragraphs, we will optimize the basic algorithm further to develop two distributed algorithms viz. 2-Rounds and Handshake algorithms.

Basic Distributed Algorithm. The basic distributed algorithm works as follows. Initially, each node assigns itself a priority, which could be its own ID or an appropriately chosen number (as described later). Next, each node collects *k-hop neighborhood information*, i.e., information about communication neighbors of all nodes that are within a communication distance of $k - 1$. Here, k is a small constant; we chose $k = 3$ for our simulations. The neighborhood information can be gathered during the data-gathering process using the piggyback strategy described in the previous section. In the remaining part of the algorithm, each node periodically tests for a set of conditions to be satisfied. If the conditions are satisfied, the node marks itself `deleted` and instructs some of its correlation neighbors to mark themselves `selected`. The `selected` marking on a node signifies that it is being

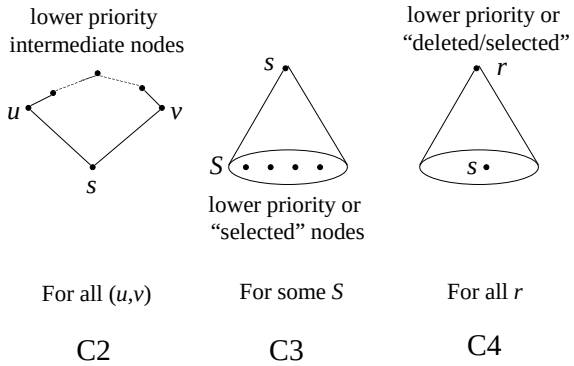


Fig. 2. Condition for marking `deleted` of a node s. The condition C2 involves communication edges, while C3 and C4 involve correlation edges.

used to infer another node and hence, should not be marked `deleted` in future. The `deleted` or `selected` marking of a node is permanent and in the end, some of the nodes may be left unmarked. If the communication graph of the initial sensor network is connected, the set of nodes that have not been marked `deleted` at any stage form a connected correlation-dominating set. The data-gathering node of the sensor network marks itself as `selected` initially, since it must be part of the connected correlation-dominating set.

Conditions for marking deleted. Informally, a node is marked `deleted` if: (i) it can be inferred (using a correlation edge) from a set of non-`deleted` nodes, and (ii) its deletion preserves the connectivity of the communication subgraph induced over the non-`deleted` nodes. Priorities are used to avoid cyclic dependency of conditions. More formally, a node s with priority $p(s)$ is marked `deleted` if the following conditions are satisfied (see Figure 2):

C1:. The node s has not been marked `selected`.

C2:. In the communication subgraph induced over the set of non-`deleted` nodes and using only the k-hop neighborhood information, every pair of neighbors (u, v) of s are connected by a communication path wherein all the *intermediate* nodes have a priority less than $p(s)$. This condition ensures that deletion of s will preserve the connectivity of the communication subgraph induced by the set of nodes not marked `deleted`.[2]

C3:. There is a correlation edge (S, s) in the correlation graph, such that every node in the set S is either marked `selected` or has a priority less than $p(s)$.[3] This condition selects a set of nodes (S) that can be used to infer s through a correlation edge.

C4:. For every correlation edge (R, r) where $s \in R$, either r is marked `deleted` or is marked `selected` or has a priority less than $p(s)$. This condition is to ensure that the set of nodes in R are not being chosen for `selected` markings by the node r at the same time.

[2]Wu et al. [2003] use a similar condition for computing a connected dominating set.
[3]Note that by virtue of the next C4 condition, no node in S could be marked `deleted`.

When a node s is marked `deleted`, the nodes in a source set S that satisfy the C3 condition are instructed by s to mark themselves `selected`. A node already marked `selected` can be used to infer other nodes without comparison of priorities.

Termination. The `selected` or `deleted` markings of a node are permanent, and hence, the distributed algorithms discussed in this section are guaranteed to terminate. At any intermediate stage of the algorithm, the set of non-`deleted` nodes forms a connected correlation-dominating set. Thus, any intermediate solution is usable by an application, and it may not be critical to explicitly *detect* termination of the algorithm. If required, the algorithm can be considered terminated after sufficient time (message latency times the upper bound on the number of messages) has elapsed. The upper bound on the number of messages is derived later.

Communication Messages. Let d be the maximum communication distance between two correlation neighbors. Initially, each node needs to gather k-hop neighborhood information and priorities of its correlation neighbors, where k is the constant in condition C2. After the initial accumulation of information from close neighbors, additional communication is incurred by the algorithm whenever a node marks itself `deleted` or `selected`. In particular, when a node s is marked `deleted` or `selected`, the following communication steps are executed.

(1) Node s informs its *correlation* neighbors of its `deleted` status, so that they could retest their C4 condition. In the *same*[4] message, node s also instructs nodes of a source set S that satisfies C3 condition to mark themselves `selected`.

(2) Node s informs its *communication* neighbors of its `deleted` status, so that they could retest their C2 condition.[5]

(3) Node s informs its *correlation* neighbors of its `selected` status, so that they could retest their C3 and C4 conditions.

In the following theorem, we prove the correctness of the basic distributed algorithm, which is unaffected by message losses. Also, the distributed algorithms designed in this section do not require any synchrony in the underlying communication model and can be easily implemented in an asynchronous system.

THEOREM 1. *The above described basic distributed algorithm correctly computes a connected correlation-dominating set, even if there are message losses.*

Proof: We start with observing that a node s marks itself `deleted` based upon its condition C1-C4 being true at *some* point of time. Below, we will show that irrespective of any additional messages being received, the set of such `deleted` nodes do not disconnect the sensor network, and that each such `deleted` node s can *always* be inferred by a set of non-`deleted` nodes S. Thus, proving the correctness of the algorithm even in the presence of message losses.

[4]If we use different messages, then we need to worry about the possibility of *only* the `deleted` message reaching a node in S. See proof of Theorem 1.

[5]Note that an unsatisfied C2 condition of a node can become `true` only by `deleted` marking of one of its neighbors. In addition, it can be shown (Wu et al. [2003]) that once the C2 condition is satisfied for a node s, the deletion of s will always preserve connectivity in the communication subgraph induced by the non-`deleted` nodes, even if other nodes get marked as `deleted`.

First, we show that the removal of nodes that are marked `deleted` does not disconnect the original communication graph of the sensor network. It is easy to see that removal of any single node s that satisfies the C2 condition will definitely maintain the connectivity of the sensor network's communication graph. Now, note that the satisfaction of C2 condition of s is based upon existence of alternate paths involving nodes with *lower* priority than that of s. `Deleted` marking of a node u in any of the alternate paths of s will result in another alternate path consisting of even lower priority nodes. This is because the node u will also satisfy the C2 condition due to nodes having lower priority than that of u and hence, that of s. Thus, if s satisfies the C2 condition, then there will always exist an alternate path connecting every pair of its neighbors. Thus, removal of s maintains the connectivity of the original communication graph of the sensor network.

Now, we show that the set of nodes that are not marked `deleted` form a correlation dominating set. It is sufficient to show that if a node s is marked `deleted`, then there will always exist a correlation hyperedge (S, s) such that no node in S has been marked `deleted`. Since, the node s satisfied condition C3, there is a hyperedge (S, s) such that each node $x \in S$ is either marked `selected` (and hence, would never be marked `deleted` in future due to condition C1) or has a priority lower than that of s. In the latter case, the node x could mark itself `deleted` *only* when it receives a message about s's `deleted` marking (see condition C4). However, as mentioned before, in the *same* message the node x is also instructed by s to mark itself `selected`. Hence, the node x will never mark itself `deleted`. Thus, no node in S is ever marked `deleted` and the node s can always be inferred using the nodes in S. Since the above is true for any node s that has been marked `deleted`, the set of non-`deleted` nodes form a correlation dominating set. ∎

We now present the 2-Rounds, Handshake and Delayed-Connection distributed algorithms, which optimize the basic distributed algorithm further.

2-Rounds Distributed Algorithm. The above described basic algorithm could be improved (in terms of increasing the number of `deleted` nodes) by comparing priorities only between nodes that are contending with each other for marking themselves `deleted`. The 2-Rounds algorithm consists of an initial round before executing the basic algorithm. In the initial round, the priority comparisons in conditions C3 and C4 are made only for nodes that satisfy the C2 condition. More precisely, for the initial round, the conditions C3 and C4 are replaced by the following modified C33 and C34 conditions.

C33:. There is a correlation edge (S, s) in the correlation graph, such that no node in the set S is marked `deleted`. In addition, each node in S is either marked `selected` or doesn't satisfy the C2 condition or has a priority less than $p(s)$.

C44:. If there is a correlation edge (R, r) where $s \in R$, then either r is marked `deleted` or marked `selected` or doesn't satisfy the C2 condition or has a priority less than $p(s)$.

To test the above C33 and C44 conditions, each node should be able to evaluate the C2 condition of its correlation neighbors. Thus, the 2-Rounds algorithms begins with gathering $(d + k)$-hop neighborhood information, where d is the maximum communication distance between correlation neighbors. After the initial round,

the 2-Rounds algorithms behaves exactly as the basic distributed algorithm. In our experiment results, we observed that 2-Rounds algorithm yielded significant improvement in the solution size over the basic distributed algorithm.

Handshake Algorithm. Handshake algorithm is essentially the basic distributed algorithm with the conditions C3 and C4 replaced by the modified conditions C33 and C44, where we compare priorities between only those nodes that satisfy the C2 condition. Thus, we require each node to transmit a "C2-satisfied" message when its C2 condition is satisfied. However, loss of "C2-satisfied" messages may result in neighboring nodes s and $\hat{s}$ both marking themselves `deleted` with s depending on $\hat{s}$ for inference. Thus, to ensure correctness in event of message losses, we need to incorporate additional "handshake" messages. The addition communication steps required in the Handshake Algorithm are:

(1) Whenever a node's C2 condition is satisfied, it transmits a 'C2-satisfied' message to all its correlation neighbors, so that they have complete information to test their C33 and C44 conditions.
(2) Before node s marks itself `deleted` (due to satisfaction of conditions C1, C2, C33, and C44), it makes a "handshake" with the nodes in S of condition C33. The handshake involves the node s sending a handshake message to the nodes in S and the nodes in S responding positively or negatively. The node s marks itself `deleted` only if it receives a positive response from all the nodes in S. A node in S sends a positive response only if it is not marked `deleted` and if it either doesn't satisfy C1 or doesn't satisfy C2 condition or its priority is less than that of s.
(3) A node in S of condition C33 marks itself `selected` only after the corresponding node s has conclusively (after positive acknowledgment from all nodes in S) marked itself `deleted`.

Handshake algorithm is expected to select a smaller connected correlation-dominating set compared to the 2-Rounds algorithm. However, the better performance comes at the cost of additional messages for 'C2-satisfied' and handshake messages. We note here that the distributed algorithms presented in this section do not have a provable bound on the quality of the solution delivered. However, in Section 5, we conduct extensive simulations to compare the performance of these algorithms with the centralized algorithms that has a provable bound on the solution quality.

Delayed-Connection Versions of Algorithms. Each of the above two algorithms viz. 2-Rounds, and Handshake algorithms can be modified to yield algorithms that first select a correlation-dominating set and then, connect the selected nodes using a communication Steiner tree. Selection of correlation-dominating set can be done by executing the original algorithms with the assumption that each node satisfies the C2 condition. After sufficient time has elapsed to guarantee termination of the first phase, a communication Steiner tree connecting the correlation-dominating set can be computed using a simple breadth-first search. In our simulations, we observed that the delayed-connection versions always performed worse than their counterparts, in terms of both the solution size returned as well as the communication costs incurred.

Number of Communication Messages. Let d be the maximum communication

distance between correlation neighbors. As mentioned before, the 2-Rounds algorithm needs to gather $(d+k)$-hop neighborhood information, while the Handshake algorithm collects k-hop neighborhood information. If n is the total number of sensor nodes in the network, l-hop neighborhood information can be gathered using ln messages. However, since the neighborhood information can also be gathered during the normal data-gathering process (using piggybacking strategy as described in Section 2.3), we ignore the communication cost incurred.

Let δ be the average[6] size of a connected dominating set of the communication subgraph induced by a sensor node and its correlation neighbors. Also, let $|D|$ be the number of nodes that get marked `deleted` and $|S|$ be the number of nodes that get marked `selected` during the entire course of the algorithm. Note that $(|D| + |S|)$ is bounded by n, the size of the sensor network. The total communication cost incurred in each of the distributed algorithms is as follows.

(1) 2-Rounds Algorithm: $\delta(|D| + |S|) \leq n\delta$.
(2) Handshake Algorithm: $\delta(|D| + |S|)$ + 'C2-satisfied' messages + handshake messages $\leq n(\delta + \delta + 2\delta) \leq 4n\delta$.

For the Delayed-Connection version of the algorithm, the first term reduces to $n\delta$. However, they incur additional (n + size of the Steiner tree) messages to construct the communication Steiner tree that connects the correlation-dominating set. Note that in the above formulae, we have assumed that the initial gathering of $d + k$ or d neighborhood information and data samples from d-hop neighborhood (for computation of correlation edges) is achieved using the piggybacking strategy, and hence, do not incur any additional messages. See Section 2.3.

Assignment of Priorities. Note that we can use different priorities for condition C2 and condition C3-C4. For the C2 condition, we use the inverse of the node's degree as the priority [Wu and Dai 2003]. For the C3-C4 conditions, we assign priority $p(s)$ for a node s as follows. Let S be a source set containing s. If node s satisfies C2 condition initially, we use $p(s) = 1/(\sum_S 1/|S|)$ (since our algorithms favors deletion of higher priority nodes), else we use $p(s) = 1/(30 * NodeDegree(s) \sum_S 1/|S|)$ so that comparison of priorities (in C3, C4) is relevant only for nodes that satisfy the C2 condition.

Handling Dynamic Changes to Correlation Graph. Change in data correlations may result in changes to the correlation graph. Thus, each `deleted` node s periodically checks for validity of the hyperedge (S, s) used in its C3 condition by gathering data from the nodes in S (using the piggyback strategy over normal data-gathering process as described in Section 2.3), recomputing the α parameters (Equation (2)), and checking if the least square error (Equation (1)) is within the given bound. If the least square error is more than the acceptable bound, the hyperedge (S, s) is invalidated, and the node s searches for a still valid hyperedge (S', s) that satisfies its $C3$ condition and consists of only non-`deleted` nodes in S'. If such a hyperedge exists, the nodes in S' are instructed to mark themselves `selected`. If a node r in S' had to be un-`deleted`, it informs its communication neighbors. If no such hyperedge exists, then the node s marks itself un-`deleted`[7] itself, and

[6]Average is across all instances when a node has to send a message.
[7]By "un-`delete`," we mean that it removes its `deleted` marking.

informs its communication neighbors. If a `deleted` neighbor v of either s or r fails to satisfy the C2 condition now due to un-`deletion` of s or r respectively, the node v un-`deletes` itself and informs all its neighbors. The above process continues, and ensures that the connectivity of the non-`deleted` nodes is maintained. To conserve energy, we do not un-`select` a node as it does not affect correctness.

Generalizations. We now discuss certain generalizations of our techniques.

Partial-Inference Hyperedges. Throughout this article, we have modeled correlation as complete inference of a node from a set of other nodes. However, our distributed and centralized algorithm (next section) can be easily generalized to model the case when a hyperedge (S, s) with a weight b signifies that the node s can save on transmission of b bits, if the data from nodes in S is available. Such a model has been used in [Chou et al. 2003]. In the above model, higher-weighted hyperedges are preferred for use in C3 condition for distributed algorithms. Moreover, unless the transmission of s is being completely suppressed, there is no need to check for C2 condition.

Generalizing Data Gathering Cost. In our problem formulation, we assumed the communication cost of data-gathering from a set of nodes M to be equal to $|M|$, based on the premise that the collective size of data from descendants at any node in the data-gathering tree is bounded and can be packaged in a constant number of message packets. If transmission of one data unit (data from one sensor node) requires one message, then the minimum number of messages required to gather data from a set of nodes M (not necessarily connected) is the sum of the shortest paths from each node in M to the data-gathering node. Here, the shortest paths may involve any node in the network. Thus, we can associate with each node a weight equal to the length of the shortest path to the data-gathering node, and formulate the problem as selecting the correlation-dominating set with minimum weight. Note that we don't need the selected set to be connected since other nodes in the network can be used as relay nodes (without incurring additional cost beyond the sum of shortest paths). Our distributed algorithms can be easily generalized to solve the above node-weighted version of the problem by ignoring condition C2, and preferring hyperedges with higher-weighted sinks and lower-weighted source nodes in condition C3. In a similar way, non-uniform battery energies can also be handled by preferring hyperedges with lower-energy sinks and higher-energy source nodes in condition C3.

Centralized algorithms developed in the next section use a notion of benefit of a set of selected nodes – which can also be easily adapted to handle the above generalizations.

4. CENTRALIZED APPROXIMATION ALGORITHM

In this section, we present a centralized approximation algorithm that returns a connected correlation-dominating set that is within an $O(\log n)$ factor of the optimal size. Based on the approximation algorithm, we design a class of polynomial-time heuristics that perform well empirically. In Section 5 (Figure 4), we will show that even the lower order polynomial-time heuristics deliver near-optimal solutions for spatial sensor networks. The centralized heuristics developed in this section also allow us to ascertain the quality of the *solution sizes* of the energy-efficient

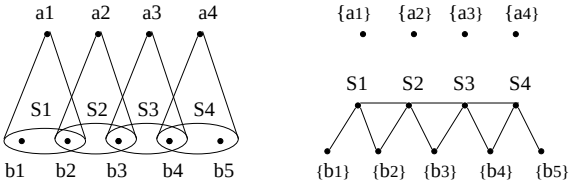


Fig. 3. Correlation graph and its intersection graph of source sets.

distributed algorithms of previous section by comparing them with the near-optimal centralized heuristics. The centralized heuristics can be implemented in a sensor network by executing the heuristic on one of the special-purpose computationally powerful sensor nodes in the network, after gathering required information from all the nodes in the sensor network. We start with some definitions for a sensor network with a correlation graph.

DEFINITION 4. (Intersection Graph of Source Sets) Let I be the set of nodes in the network, and $\mathcal{I} = \{\{s\}|s \in I\}$. Let $\mathcal{S}$ be the set of source sets in the correlation graph of the network. The *intersection graph of source sets* is the simple graph $G(V = \mathcal{S} \cup \mathcal{I}, E = \{(v_1, v_2)|(v_1 \cap v_2) \neq \phi\})$.[8]
See Figure 3. □

DEFINITION 5. (Connected Subgraph of Sources; Connected Source Set) A connected subgraph in the intersection graph of source sets is called a *connected subgraph of sources*.[9] A *connected source set* is a set of nodes corresponding to some connected subgraph of sources, i.e., the union of the sets corresponding to the vertices of a connected subgraph of sources. For example, in Figure 3, $S1, S2$, and $S3$ form a connected subgraph of sources and the corresponding connected source set is $\{b1, b2, b3, b4\}$. □

DEFINITION 6. (Inferred Nodes) Given a set of nodes S, the set of inferred nodes for S is denoted by $I(S)$ and is defined as

$$I(S) = S \cup \{x|(Y, x) \text{ is a correlation edge and } Y \subseteq S\}.$$

□

DEFINITION 7. (Benefit of a Set of Nodes) Benefit of a set S with respect to a set M of nodes M is denoted by $B(S, M)$ and is defined as $B(S, M) = |(I(S) - I(M)|/|S - M|$, where $I(S)$ and $I(M)$ are the set of inferred nodes for S and M respectively. □

[8]We include the elements of $\mathcal{I}$ in the vertices of graph G, since a node trivially infers itself. Hence, the correlation graph can be thought of to have trivial hyperedges $(\{s\}, s)$ for each node s in the network, but we have precluded presence of these hyperedges in a correlation graph for sake of clarity.
[9]The notion of "connected" here is in context of the intersection graph of source sets and has nothing to do with the communication graph.

Approximation Algorithm. Our proposed approximation algorithm consists of two phases. The first phase is a greedy phase that constructs a near-optimal correlation dominating set. The second phase runs a Steiner tree approximation algorithm [Berman and Ramaiyer 1994] to connect the correlation-dominating set constructed in the first phase. The first greedy phase of the algorithm works as follows. Let M denote the set of already selected nodes at any stage. Initially M contains the data-gathering node. At each stage, the algorithm adds to M the connected source set that has the maximum benefit with respect to M at that stage. The phase of the algorithm terminates when M becomes a correlation-dominating set. The second phase of the algorithm builds a communication Steiner tree over the set of nodes in M by adding additional nodes. We build the communication Steiner tree by iteratively connecting the closest pair of connected components, i.e., the pair of components that can be connected by addition of minimum number of nodes.

THEOREM 2. *The size of the connected correlation-dominating set returned by the above described centralized algorithm is at most* $2d(1+\log n)|OPT|$, *where* OPT *is the optimal correlation-dominating set,* d *is the maximum communication distance between two correlation neighbors, and* n *is the number of nodes in the sensor network.*

Proof: Whenever a connected source set S is added to M by the greedy first phase of the algorithm, we charge the newly inferred nodes $I(S)-I(M)$ with the number of unselected (not in M) nodes in S, i.e. $|S-M|$. The charge $|S-M|$ is evenly distributed on each of the newly inferred $I(S)-I(M)$ nodes giving each newly inferred node a charge of $|S-M|/|I(S)-I(M)|$.

Let OPT be an optimal correlation-dominating set (not necessarily connected). Consider the subgraph ζ of the intersection graph of source sets induced by the source sets and nodes contained in OPT. Let $\zeta_1, \zeta_2, \ldots, \zeta_l$ be the connected components in the induced subgraph ζ. Let S_{ζ_i} be the connected source set corresponding to the component ζ_i. Also, let T_i be the charge accumulated by $I(S_{\zeta_i})$, the set of inferred nodes for S_{ζ_i}, during the entire course of the algorithm. Since, S_{ζ_i} is also considered for selection by the greedy algorithm at each stage, it can be shown ([Cormen et al. 2001]) that T_i is at most $|S_{\zeta_i}|(1+\log|I(S_{\zeta_i})|)$. Thus, the total charge T accumulated by $I(OPT)$, the set of all nodes inferred by optimal solution OPT, during the course of the algorithm can be bounded as follows.

$$
\begin{aligned}
T &\leq \sum_{i=1}^{l}(1+\log|I(S_{\zeta_i})|)|S_{\zeta_i}| \\
T &\leq (1+\log n)\sum_{i=1}^{l}(|S_{\zeta_i}|) \\
T &\leq (1+\log n)|OPT|.
\end{aligned}
$$

The last equation follows from the fact that the sets of nodes $S_{\zeta_1}, S_{\zeta_2}, \ldots, S_{\zeta_l}$ are connected components of the graph ζ, and hence, form a mutually-disjoint *partition*

of the optimal set of nodes OPT.[10] Now, since $I(OPT)$ is the set of all nodes in the network, the total charge T accumulated by $I(OPT)$ is the size of the correlation dominating set M returned by the first phase of the algorithm. Thus, we get $|M| \leq (1 + \log n)|OPT|$.

We now show that the second phase of the algorithm adds at most $(2d-1)|M|$ additional nodes. Consider the connected components $M_1, M_2, \ldots, M_p$ in the communication subgraph induced by M. It can be shown by contradiction that there exist two nodes in different connected components M_i and M_j such that the nodes are connected by a communication path of length at most $2d-1$. Since the above is true for any correlation dominating set, and as $p \leq |M|$, all the connected components of M can be connected together using at most $(2d-1)|M|$ additional nodes. Thus, the size of the connected correlation-dominating set returned by our approximation algorithm is at most $2d(1 + \log n)|OPT|$. ∎

Gupta [1997; 1999] has used similar techniques to construct approximation algorithms for a related dual problem of selection of views to materialize in a general data warehouse. The time complexity of the above proposed approximation algorithm is exponential in n, the total number of nodes in the network, since the number of connected source sets considered at each stage of the first phase can be exponential. However, the approximation algorithm is non-exhaustive and may perform better for sensor networks whose intersection graph of source sets has few edges. More importantly, the approximation algorithm gives us an insight and a basis to design polynomial-time greedy heuristics that restrict the search space by considering only a polynomial number of connected sources sets at each stage. Note that locality of data correlations (within a certain d-hop neighborhood) does not help improve the approximation ratio or the complexity of the above described approximation algorithm.

Polynomial-time Heuristics. Based on the above approximation algorithm, we design the following class of l-hop polynomial-time heuristics (without any provable performance guarantee). For a given l, the l-hop heuristic works as follows. At each stage, the l-hop heuristic constructs, for *each* source set S, the connected source set $f_l(S)$ (defined in the next paragraph). Then, the algorithm picks the $f_l(S)$ that has the maximum benefit and adds it to the already selected set of nodes M. After M has become a correlation-dominating set, additional nodes are added to construct a communication Steiner tree spanning over M.

The connected source set $f_l(S)$ for a given S is constructed in a greedy manner by merging with S the best source set that is within a distance of at most l from S in the intersection graph of source sets. The greedy construction of $f_l(S)$ stops when its benefit cannot be improved further.

Example. For the correlation graph in Figure 3, in the first stage (when $M = \phi$) of the 1-hop heuristic, $f_1(S_1) = S_1 \cup S_2$ and $f_1(S_2) = S_1 \cup S_2 \cup S_3$. Since, the benefit of $f_1(S_2)$, $B(f_1(S_2), M)$, is the maximum for all $f_1(S_i)$, the 1-hop heuristic sets $M = S_2 \cup S_1 \cup S_3$ in the first stage.

[10] It is for this reason that the approximation algorithm considers all possible connected source sets at each stage.

ALGORITHM 1. l-hop Centralized Heuristic

Input: A sensor network with a correlation graph.
Output: A connected correlation-dominating set M.
BEGIN
Let ζ denote the intersection graph of the source sets in the correlation graph.
$M = \{v\}$, where v is the data-gathering node;
while (1)
$MaxB = 0$;
$Best\hat{S} = \phi$;
for all source sets S such that $B(S, M) > 0$
/* Construct $\hat{S} = f_l(S)$ */
$\hat{S} = S$;
Let $S_1, S_2, \ldots, S_p$ be the source sets that are at a distance of at most l from S in the graph ζ.
while $(\exists S_i \mid B(\hat{S} \cup S_i, M) > B(\hat{S}, M))$
Pick S_i for which $B(\hat{S} \cup S_i, M)$ is maximum
$\hat{S} = \hat{S} \cup S_i$
end while;
if $(B(\hat{S}, M) > MaxB)$ $Best\hat{S} = \hat{S}$;
end for;
$M = M \cup Best\hat{S}$;
if $(Best\hat{S} = \phi)$ break;
end while;
Add any remaining un-inferred nodes to M.
Connect M using a 2-approximation Steiner tree algorithm over the communication graph of the network.
RETURN M.
END. ◇

In the above described l-hop heuristic, the number of iterations of the outer *while* loop as well as the *for* loops is bounded by the total number of source nodes (the number of vertices in the intersection graph of source sets). Also, the number of iterations of the inner *while* loop is bounded by g^l, where g is the maximum degree of a vertex in the intersection graph of source nodes. Thus, the overall time complexity of the above described l-hop heuristic is $O(nm^2g^l)$, since computation of $B(S, M)$ may incur $O(n)$ cost where n is the total number of nodes in the network. If the size of source sets is bounded by a constant L, then the worst-case time complexity of the above l-hop heuristic is $O(n^{3L+1})$ since number of iterations of each of the three loops is bounded by n^L (the maximum number of source nodes). We expect 1-hop or 2-hop heuristics to perform very well for ad-hoc wireless networks with spatial data correlations where the size of source sets is expected to be small. In fact, using extensive simulations on random sensor networks, we demonstrate that 1-hop heuristic yields a near-optimal connected correlation-dominating set. Thus, the near-optimal 1-hop heuristic helps us ascertain the quality of the solution sizes

of the distributed algorithms developed in the previous section by comparison of solution sizes as shown in the next section.

5. PERFORMANCE RESULTS

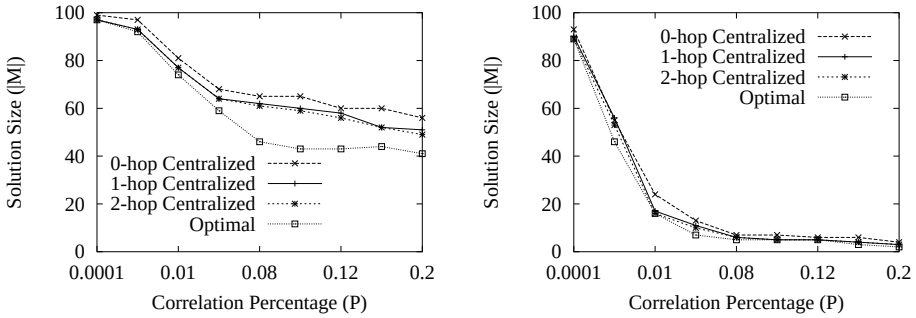
In this section, we present our simulation results that demonstrate the performance and effectiveness of our proposed approach and algorithms. In particular, we compare the size of the connected correlation-dominating set returned by our proposed distributed and centralized algorithms viz. 2-Rounds, Handshake, 0-hop, 1-hop, and 2-hop centralized heuristics. Note that 0-hop is essentially a naive greedy centralized approach. In addition, we compare the communication costs incurred by the two distributed algorithms. We ignore the communication cost incurred in gathering neighborhood information and data samples (for computation of correlation edges), since they can be gathered using the piggybacking strategy or will incur almost same cost for all distributed approaches. Note that due to the small values of d, k, and K (2, 2, and 3 respectively) used in our simulations, the piggybacking strategy is needed over only a small number of snapshots.

We present results from four sets of experiments that elicit various interesting properties of our approaches. The four cases differ in how the sensor network and sensor data are generated. In the first case, the sensor network as well as the correlation graph are generated randomly. In the second set, records of temperature data from 100 US cities is used as the sensor data set, and the selected cities form the sensor network. In the third set, the sensor network is generated randomly, while the data is generated by simulating the behavior of hypothetical signal sources. Finally, in the fourth set, we use data from a real sensor network deployed at [James Reserve Data Management Systems].

Computation Cost Model. In all our experiments, we estimate energy consumed in computing hyperedge parameters as follows. Using the calculations in Section 2.3, the computation of Equation 2 (for K=3 and L=5) consumes roughly about 1/25 fraction of energy required to transmit one message on Berkeley motes. Here, we assume message size of 1K bits. Initial computation of the correlation graph requires each node computing the Equation 2 about 1000 times (number of possible hyperedges in a 2-hop neighborhood size of around 20) for the first (synthetic data) and third (time-varying signals) sets of experiments, and around 100 times (2-hop neighborhood size of around 10) for the temperature data. We have assumed computation of hyperedge parameters at individual nodes, but depending on the resources available the computation could also be done at a powerful central node.

5.1 Random Sensor Networks with Synthetic Correlations

We generated data for random sensor networks as follows. First, we randomly place 1000 sensors in an area of 40×40 units. Two sensors can communicate with each other if they are located within each other's transmission radius. Since varying the number of sensors is tantamount to varying the transmission radius for the purposes of evaluating our algorithms, we fix the number of sensors to be 1000 and vary the transmission radius. We generate the correlation graph over the sensor network as follows. For each node s and a set of nodes S consisting of 1 to 3 sensor nodes that are within a communication distance of at most $d = 2$



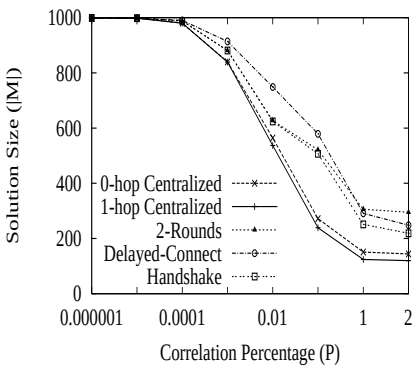
(a) Transmission Radius = 2 (b) Transmission Radius = 4

Fig. 4. Comparison of centralized heuristics with optimal algorithm for small sensor networks (100 sensors in a 7×7 area with synthetically generated correlations).

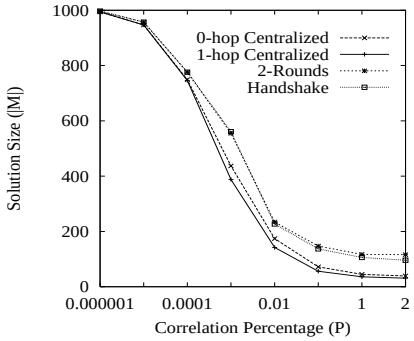
from s, we add the hyperedge (S, s) with a probability of $P/100$, where P is a constant less than 100 and is referred to as the *correlation percentage*. Based on the above parameters, the expected number of 2-hop neighbors of a node is $1000\pi(2r)^2/1600$, where r is the transmission radius. Thus, for a given correlation percentage P, the expected number of hyperedges (S, s) for a node s for $r = 2$ is about $(\binom{32}{1} + \binom{32}{2} + \binom{32}{3})P/100 = 55P$, and about $3500P$ for $r = 4$. Below, we analyze the performance of our algorithms for a sufficiently wide range of P values and r equal to 2 and 4.

Centralized Heuristics' Solution Sizes. In Figure 4, we compare the quality of some of the centralized heuristics with the *optimal* (exhaustive search) algorithm for small size sensor networks (100 sensors placed randomly in a 7×7 area) for transmission radius 2 and 4. We notice that the 0-hop, 1-hop, as well as 2-hop centralized heuristic perform quite close to the optimal solution. We observe that the 1-hop heuristic performs quite better than the 0-hop heuristic, while there is no noticeable difference in the solution sizes of 1-hop and 2-hop centralized heuristics. Since, the l-hop heuristics are based on the approximation algorithm that provably returns a near-optimal solution, and increase in l results in increasingly better performance, we conjecture that 1-hop heuristic returns a near-optimal solution for dense sensor networks with rich spatial data correlations involving small source sets. For larger sensor networks (see Figures 5 (a) and (b)), we report the solution sizes returned by 0-hop and 1-hop heuristics, and compare them with the solution sizes of the distributed algorithms. In each of the above experiments, we increased the correlation percentage P until the solution size reaches a saturation point (i.e., stops decreasing). Further increase in P does not result in much decrease in solution size, and hence, not shown.

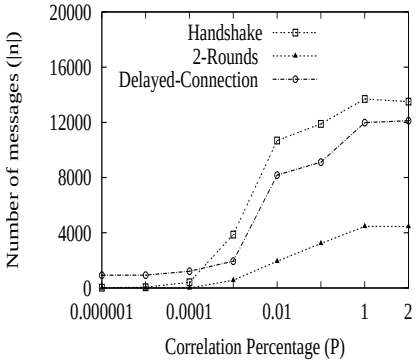
Distributed Algorithms' Solution Sizes. We observe that the Handshake algorithm returns a smaller solution size than the 2-Rounds algorithm (see Figures 5 (a) and (b)). However, the solution size returned by the Handshake algorithm is only marginally better than the 2-Rounds algorithm, which demonstrates the effective-



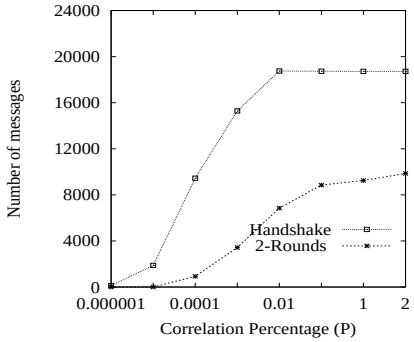
(a) Solution sizes with r=3



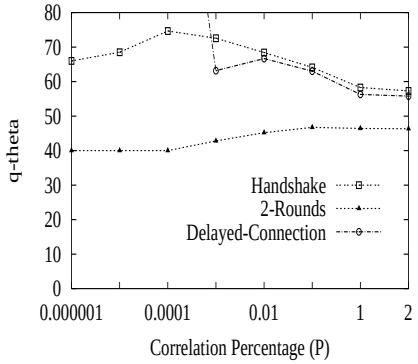
(b) Solution sizes with r=5



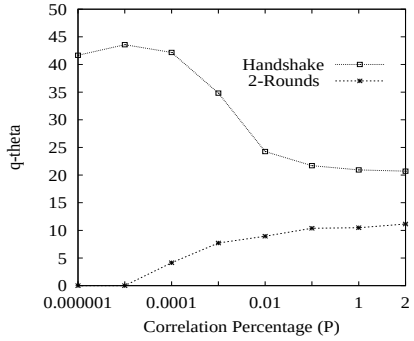
(c) Number of messages with r=3



(d) Number of messages with r=5



(e) q_θ with r=3



(f) q_θ with r=5

Fig. 5. Comparison of different centralized and distributed algorithms for large and dense (1000 nodes in 40×40 units square area) randomly generated sensor networks with synthetically generated correlations. Transmission radius is 3 or 5 units.

ness of the initial round and the strategy of subsequent assignment of lower priorities to nodes that don't satisfy the C2 condition in the 2-Rounds algorithm. We also evaluated the delayed-connection versions of the Handshake and 2-Round algorithms wherein the algorithms first select a correlation-dominating set and then, connect the selected nodes using a communication Steiner tree. We observed that the delayed-connection versions always performed worse than their counterparts, in terms of both the solution size returned as well as the communication costs incurred.

Distributed vs. Centralized Solution Sizes. As expected, the solution sizes returned by the centralized heuristics is always better than the solution sizes returned by the distributed algorithms (see Figures 5 (a) and (b)). This is hardly surprising, since, the centralized heuristics have global information of the sensor network and have time complexities in high order polynomials. In contrast, the distributed algorithms are localized and incur only a linear number of messages. However, we should note that the distributed algorithms still perform impressively close to the near-optimal centralized heuristics.

Number of Messages. The communication cost incurred by the 2-Rounds algorithm is much less than that incurred by the Handshake algorithm (see Figures 5 (c) and (d)).[11] Since, the solution size returned by the Handshake algorithm is only marginally better than the 2-Rounds algorithm, we can conclusively say that the 2-Rounds algorithm is the best performing distributed algorithm.

Let D be the the total communication cost incurred by a given distributed algorithm and n be the total number of sensors in the network. As discussed in Section 2.1, as long as the data-gathering query requires more than $q_\theta = \frac{D}{(n-|M|)}$ snapshots the overall communication cost for the data-gathering query using the given distributed algorithm is lower than that incurred using the naive approach, wherein all sensor nodes are involved in transmitting data to the data-gathering node. We plot the q_θ value in Figure 5 (e) and (f). As estimated before, we have added 40 message transmissions per node for about 1000 computations of Equation 2. We can see that the value of q_θ for the best performing 2-Rounds distributed algorithm is around 50, which is encouraging since data-gathering queries typically gather a much larger number of snapshots.

5.2 Sensor Networks with Time Varying Data

We ran three sets of experiments to evaluate the performance of our distributed algorithms under dynamic conditions, i.e., when the sensor data and hence, correlation graph changes, and sensor nodes may die after battery depletion. For the first set of experiments, we used real temperature data of US cities; for the second set, we generated synthetic time varying data; for the third set, we used data from a real sensor network. We start with describing inference of correlation hyperedges, modeling of node battery power, a new 2-Rounds-Multiple algorithm (based on using multiple correlation dominating sets), and quantifying error in gathered data.

[11] Note that for very low correlation percentages, the 2-Rounds algorithm incurs zero number of messages, because the nodes marked `deleted/selected` in the initial round do not need to transmit as their correlation neighbors can deduce their markings using the $(d + k)$-hop neighborhood information.

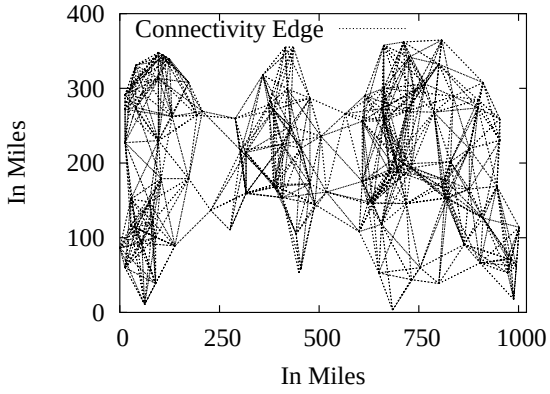
Correlation Graph. A correlation hyperedge (S, s) is created for a node s and a set of nodes S, where S consists of 1 to 3 nodes in the 2-hop neighborhood of s, if the least square error (Equation 1) calculated for optimal α parameters (Equation 2) is within a certain error threshold. We run experiments for various error threshold values. Initially, the correlation graph is constructed, and the distributed algorithm executed to compute the connected correlation-dominating set. Over time, invalidation of *used* hyperedges is detected, and the solution incrementally maintained as described in Section 3. Also, just before a sensor node dies, it informs its correlation neighbors who rerun the distributed algorithm.

Battery Power. The sensor network is tasked to gather snapshots of the sensor data values in the entire region. Each sensor is initialized with a battery power chosen randomly between 750-1250 units, where 1 unit of battery power is consumed by every message transmission. Messages are transmitted to compute the hyperedge parameters, to execute the distributed algorithm, and to gather snapshots from the connected correlation-dominating set. As estimated before, each node consumes about 40 units of battery energy for about 1000 computations of Equation 2. Note that, in absolute terms, the energy consumption at each node for initial computation of correlation edges is of the order of a few millijoules, which is negligible compared to the total energy (a few Joules) in a normal battery.

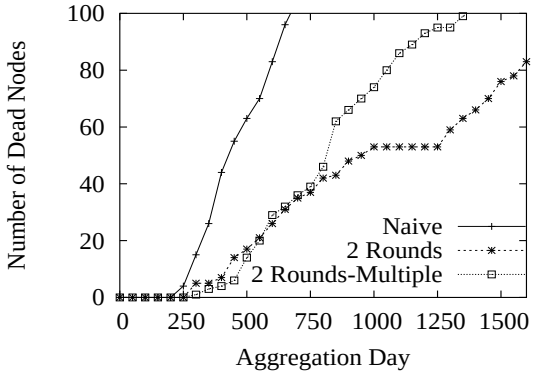
2-Rounds-Multiple Algorithm. We also evaluate a load balancing scheme (2-Rounds-Multiple) where multiple disjoint dominating sets share the task of data gathering in a round robin schedule. The goal is to balance the energy usage among all nodes in the network. In the simulations, the 2-Rounds distributed algorithm is run twice at the start to construct two correlation dominating sets. For the second run, node priorities are changed in order for the algorithm to pick a different selected set. In particular, the priority of nodes that were marked `selected` in the first run is increased, while the priority of nodes that were marked `deleted` is decreased. As a result, the nodes not selected in the first run have a higher chances of marking themselves `selected` in the second run. Next, the data gathering task is periodically rotated between the two selected correlation dominating sets thus selected. For our experiments, we rotate after every data gathering.

Quantifying Error in Gathered Data. We compare the performance of our distributed algorithms with a Naive approach, wherein all sensors in the network report data for each data gathering snapshot. We use the root mean square (RMS) approach to quantify the error in the gathered data. Note that the gathered data may have inaccuracies because of the following two reasons. In our approach of gathering data only from a connected correlation-dominating set, bounded errors are introduced when data for other sensors is inferred. Also, for both approaches (Naive and ours), data for dead sensors is "recovered" at the data-gathering node by using an available hyperedge involving alive source nodes. If no such hyperedge exists, then we take an average of the data from 2 of its nearest alive neighbors.

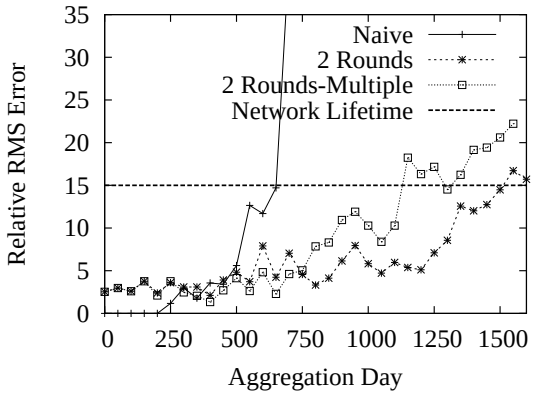
Simulations on Temperature Data. We gathered daily average temperature data of 100 US cities [National Climatic Data Center] for the last decade. We consider a sensor network formed by placing a sensor node at each of the 100 city locations. We chose a transmission radius of 150 miles for each sensor node, so that each sensor has a good number of communication neighbors. We choose an error



(a) Communication Graph



(b) Number of Dead Sensors



(c) RMS Error in the Gathered Data

Fig. 6. Simulation experiments with real temperature data. Sensor nodes with transmission radius of 150 miles are located at 100 US cities in a 400×1000 square miles area.

threshold of 5% for creating correlation hyperedges, since lower error thresholds resulted in a large correlation dominating set, and higher error thresholds had too many hyperedges resulting in a large computation overhead. We will present a performance comparison due to various choices of error thresholds in the next set of experiments (on light signal data).

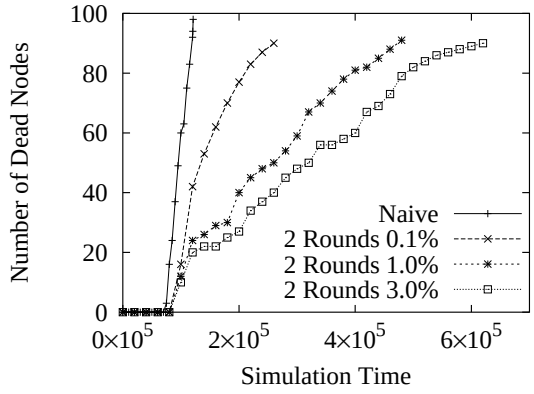
Figure 6(a) depicts the connectivity graph of the sensor network created, while Figure 6(b) and (c) plot the number of dead sensors and the RMS error respectively in the gathered data for the 2-Rounds algorithm, 2-Rounds-Multiple algorithm, and Naive approach. We have only shown 2-Rounds algorithm in the figure, as 2-Rounds consistently outperformed Handshake for the used parameter values. Note that there is no error in the beginning for the Naive approach. However, there is a very sharp rise in the error when the nodes start dying, and the nodes are all dead soon after 600 days. In contrast, both 2-Rounds and 2-Rounds-Multiple algorithms incur some error throughout and the error gradually increases when the nodes start dying. The 2-Rounds and 2-Rounds-Multiple algorithms have similar performance until 750 days, after which the performance of 2-Rounds-Multiple algorithm deteriorates in comparison.

If we define *network lifetime* as the time span uptil which the RMS error remains below 15%, then the lifetime of Naive approach is only 700 days while that of 2-Rounds-Multiple and 2-Rounds is 1200 and 1500 days respectively. Thus, these simulations demonstrate the effectiveness of our approach in increasing the useful lifetime of a sensor network.

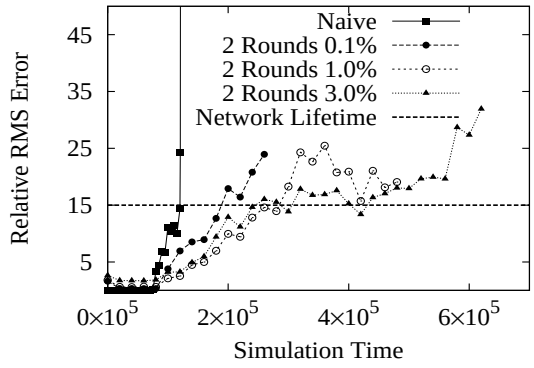
Simulations on Synthetic Time Varying Data. For this experiment, we randomly place 100 sensor nodes, each having a transmission radius of 3 units, in an area of 10×15 square units. We also place a number of signal sources[12] at random locations in this region. The signal intensity at a distance d from the source is assumed to be inversely proportional to d^2, and the intensity at any point is the sum of the intensities from individual light sources. The sources generate a signal with an intensity chosen randomly between 100 and 200 units. At random intervals (with a mean period of 2000 time units), the signal sources either increase or decrease their intensity by 5% or remain unchanged with equal probability. We ran simulations for error threshold values of 0.1, 1, and 3 percentages. The sensor network is tasked to gather snapshots of the signal intensity levels in the entire region at regular intervals of 100 time units.

Figure 7(a) and (b) plot the number of dead sensors and the RMS error in the gathered data for the Naive and 2-Rounds algorithms, for various choices of error threshold values. Here, we do not show results for Handshake and 2-Rounds-Multiple, since they performed consistently worse than the 2-Rounds algorithm (as in the previous sets of experiments). We notice better performance for higher error thresholds. In particular, we observe that the RMS error remains at a reasonable level for upto 600,000 time units for 3% error threshold. Figure 7(c) plots the effect of the error threshold (used in creating correlation hyperedges) used by 2-Rounds algorithm on energy consumption and network lifetime. In particular, for a particular time instant (at 60,000 simulation time), Figure 7(c) plots the total

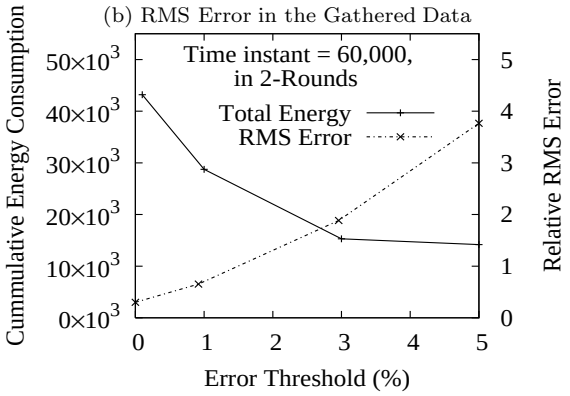
[12]The exact nature of the signal is not important for our purposes.



(a) Number of Dead Sensors



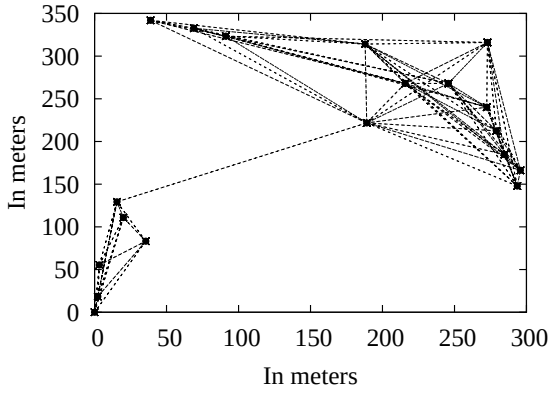
(b) RMS Error in the Gathered Data



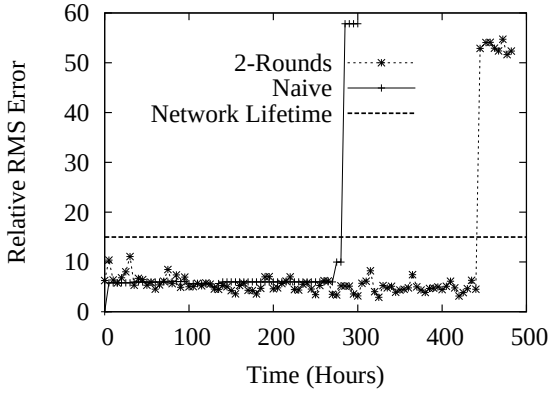
(c) Total Energy Consumed Vs. Relative RMS Error at 60,000 simulation time of 2-Rounds Algorithm for Various Error Thresh olds (used for created the correlation hyperedges)

Fig. 7. Simulation experiments for random sensor networks (100 nodes with transmission radius of 3 units in a $10\times$ 15 area) with time-varying signal data for various error thresholds.

energy consumed in the entire network as well as RMS error, for different error thresholds of 2-Rounds algorithm. We can see that use of a higher error threshold results in a smaller solution (connected correlation-dominating set) and thus lower total energy cost, but yields a higher RMS error.



(a) Communication Graph



(b) RMS Error in the Gathered Data

Fig. 8. Simulation experiments with real sensor data from James Reserve, CA. Network is formed of 19 sensor nodes with transmission radius of 200 meters each.

Simulation on Real Sensor Data. Finally, we ran simulations using real temperature data collected from 19 sensor motes that are currently deployed in James Reserve National Forest in California. We used hourly temperature readings for these sensors available at [James Reserve Data Management Systems]. To derive the network topology, we used the actual latitude and longitude information for these sensor nodes and a transmission radius of 200 meters. See Figure 8(a). We initialized the battery power of each node to a random value between 0 to 350

units, and evaluated the performance of Naive and 2-Rounds algorithms. For the 2-Rounds algorithm, we used an error threshold of 5% to compute the hyperedges. Figure 8(b) shows the performance of the algorithms. We see that 2-Rounds algorithm extends the lifetime of the network by 60% as compared to Naive algorithm. Moreover, the RMS error remains below 10% until about 450 hours, when majority of the sensor nodes die.

5.3 Summary of Simulation Results

Based on the above described simulation results, we can conclude that the 2-Rounds algorithm is the best performing algorithm among all the designed distributed algorithms, in terms of the size of the connected correlation-dominating set selected, the number of messages incurred, and RMS error introduced in the gathering data. Also, 2-Rounds algorithms returns a solution whose size is quite close to that returned by the 1-hop heuristic, which seems to return a near-optimal solution for spatially correlated dense sensor networks. Based on the total number of messages incurred and the savings achieved, we conclude that the approach of constructing a connected correlation-dominating set using the 2-Rounds algorithm will result in substantial energy savings.

6. RELATED WORK

The problem of efficient gathering of correlated data in a sensor network has been recently addressed by Rickenbach et al. [2004], Chou et al. [2003], and Cristescu et al. [2003; 2004], where the focus is on reducing the total number of bits transmitted to the data-gathering node using coding techniques. In particular, Rickenbach et al. [2004] focus on single-input coding strategies (i.e., simple correlation edges). They consider two coding schemes viz. foreign-coding and self-coding, and present algorithms to construct optimal (minimum weighted number of bit transmissions) and near-optimal data-gathering trees for foreign-coding and self-coding respectively. In [Chou et al. 2003], the savings are achieved by having the data-gathering node track the correlation structure among nodes and then, use this information to inform the sensor nodes the number of bits they should use for encoding their measurements. However, they assume a fixed correlation structure and a "star" topology, and do not address the optimization problem of minimizing the number of bit transmissions. Lastly, Cristescu et al. [2004] consider a coding strategy based upon Slepian-Wolf model and design efficient distributed approximation algorithms optimizing the transmission structure and the rate allocation at the nodes. In all of the above methods, *all* sensors are engaged in data transmission albeit with reduced number of bit transmissions. As noted before, the model and techniques developed in this article can also be extended to optimize total number of bit transmissions.

Similar to the idea proposed in our paper, Yoon and Shahabi [2005] propose a mechanism (called CAG) that reduces the number of transmissions and provides approximate results to aggregate queries by utilizing the spatial correlation of sensor data. Like our approach, they also select only a select of nodes for transmission of data to the sink node. However, their formulation of the problem is a simpler version of our problem wherein correlation graph consists of only simple edges (connecting a node to its ancestor in the data gathering tree). They select a set of

clusterheads (which actually form a correlation dominating set in our terminology) using a simple localized scheme during the query propagation phase. Due to connectivity requirement, each node that has at least one descendant as clusterhead is involved in transmission. The main shortcomings of the above approach is that it uses a very simple notion of correlation, and uses *only* the edges of the forwarding tree (typically the shortest path tree) for selection of clusterheads and connecting nodes. In other closely related works, [Vuran and Akyildiz 2006] exploits spatial correlation at the MAC level, and [Doherty and Pister 2004] evaluates various simple localized selection strategies in terms of various cost metrics and error in the reconstructed field.

The problem of constructing an efficient *aggregation* tree to reduce the total bits of transmitted in the network have also been addressed recently [Goel and Estrin 2003; Enachescu et al. 2004]. In particular, Goel et al. [2003] look at the problem of finding efficient trees to send aggregated information to a sink, where information can be aggregated at intermediate nodes. They present a randomized tree construction algorithm, which is a good approximation simultaneously for all concave non-decreasing aggregate functions. Authors Enachescu et al. [2004] analyze a simple randomized tree construction algorithm that achieves a constant factor approximation of the optimal tree for grid network topologies. Both of the above works assume data compression models specific to aggregation, wherein *any* k data values can be compressed into a data value of appropriately defined size. In contrast, the correlation model considered in our article is more general, wherein only the given set (which can be arbitrary) of data values can be compressed depending on the correlation structure present in the network.

Recently, Marco et al. [2003] analyzed the data transport capacity of a dense sensor network in data-gathering applications. In their model, all sensors in the network encode/compress their measured samples and transmit them to a single data-gathering node. They show that as the density of a sensor network increases to infinity, then the total number of bits required to attain a given quality of reconstructed field also increases to infinity under *any* compressing scheme. Thus, the only way to limit the total amount of data transmitted below the network's transport capacity would be to suppress transmission from some sensors to prevent "oversampling." Authors Scaglione and Servetto [2002] show that if data can be encoded *en route* – for example, at the tree nodes – then the capability of the dense sensor network and the correlation structure of a typical random field are sufficient to permit data gathering by any nodes within any given distortion value. However, an appropriate routing technique must still be devised, which remains an open question. Our work proposes an alternate efficient approach to reduce data communication in data-gathering applications by minimizing the number of sensors that are involved transmitting their data to the data-gathering node. In other related work, Pattem et al. [2004] analyze the performance of various routing with lossless compression schemes, and show that near-optimal performance can be achieved using a static clustering scheme for the case of 1D and 2D array of sensor nodes for a wide range of spatial correlations.

There has been a significant amount of work on the related problem of computing a minimum connected dominating set [Guha and Khuller 1998; Dubhashi et al. 2003]

in a distributed manner. A connected dominating set (CDS) is used for efficient broadcasting of a message in a mobile ad hoc network, since only the nodes in CDS need to forward the message to its neighbors. The work in wireless network research community ([Das et al. 1997; Laouiti et al. 2002; Wu and Li 2001; Alzoubi et al. 2002; Chen and Liestman 2002; Wu and Dai 2003; Deb et al. 2003]) has primarily focussed on developing energy-efficient distributed algorithms to construct a good connected dominating set. The connected dominating set problem is a special case of the connected correlation-dominating set problem wherein the correlation graph consists of undirected simple edges and is the same as the communication graph.

In other related works, GAF [Xu et al. 2001], SPAN [Chen et al. 2001], PEAS [Ye et al. 2003], and ASCENT [Cerpa and Estrin 2002] develop distributed algorithms to identify nodes that are similar in routing perspectives so that other nodes can be turned off to conserve energy. None of these works use any notion of a data correlation structure.

7. CONCLUSIONS

In this article, we have considered the connected correlation-dominating set problem that helps in minimizing communication costs in data-gathering sensor network applications. Taking advantage of the existing data correlations in the sensor network, our proposed approach is to select a small set of sensors called the connected correlation-dominating set that form a connected communication graph and are sufficient to infer data of the remaining unselected sensors. The problem is defined in terms of a correlation structure (hypergraph) that captures general data correlations. To select a connected correlation-dominating set of small size, we have designed a set of energy-efficient distributed algorithms, and a class of centralized heuristics that are based on a provably competitive approximation algorithm. Our simulation results show that the designed distributed algorithms and the centralized heuristics return small size solutions and the communication cost incurred by the best performing distributed algorithm (2-Rounds) is small enough to considerably increase the useful lifetime of a sensor network.

Acknowledgments

This work was partially supported by an NSF grant (award number ANI-0308631).

REFERENCES

Alzoubi, K. M., Wan, P.-J., and Frieder, O. 2002. Message-optimal connected dominating sets in mobile ad hoc networks. In *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*.

Badrinath, B., Srivastava, M., Mills, K., Scholtz, J., and Sollins, K., Eds. 2000. *Special Issue on Smart Spaces and Environments,* IEEE Personal Communications.

Berman, P. and Ramaiyer, V. 1994. Improved approximation algorithms for the steiner tree problem. *J. Algorithms 17*.

Cerpa, A. and Estrin, D. 2002. Ascent: Adaptive self-configuring sensor networks topologies. In *Proceedings of the IEEE INFOCOM*.

Chen, B., Jamieson, K., Balakrishnan, H., and Morris, R. 2001. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*.

Chen, Y. and Liestman, A. 2002. Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks. In *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*.

Chou, J., Petrovic, D., and Ramchandran, K. 2003. A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks. In *Proceedings of the IEEE INFOCOM*.

Chu, M., Haussecker, H., and Zhao, F. 2002. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *IEEE Journal of High Performance Computing Applications*.

Cormen, T., Lieserson, C., Rivest, R., and Stein, C. 2001. *Introduction to Algorithms*. McGraw Hill.

Cristescu, R., Beferull-Lozano, B., and Vetterli, M. 2004. On network correlated data gathering. In *Proceedings of the IEEE INFOCOM*.

Cristescu, R. and Vetterli, M. 2003. Power efficient gathering of correlated data: optimization, NP-completeness and heuristics. *SIGMOBILE Mob. Comput. Commun. Rev. 7,* 3, 31–32.

D. Culler et al. 2004. TinyOS. http://www.tinyos.net.

Das, B., Sivakumar, R., and Bhargavan, V. 1997. Routing in ad hoc networks using a spine. In *Proceedings of the Intl. Conf. on Computer Communications and Networks (IC3N)*.

Deb, B., Bhatnagar, S., and Nath, B. 2003. Multi-resolution state retrieval in sensor networks. In *Proceedings of Intl. Workshop on Sensor Network Protocols and Applications*.

Doherty, L. and Pister, K. 2004. Scattered data selection for dense sensor networks. In *Proceedings of the International Workshop on Information Processing in Sensor Networks (IPSN)*.

Dubhashi, D., Mei, A., Panconesi, A., Radhakrishnan, J., and Srinivasan, A. 2003. Fast distributed algorithms for (weakly) connected dominating sets and linear-size skeletons. In *Proceedings of the ACM Symposium on Discrete Algorithms (SODA)*.

Enachescu, M., Goel, A., Govindan, R., and Motwani, R. 2004. Scale free aggregation in sensor networks. In *Proceedings of the First International Workshop on Algorithmic Aspects of Wireless Sensor Networks (Algosensors)*.

Estrin, D., Govindan, R., and Heidemann, J. 2000. eds. *Special Issue on Embedding the Internet,* Communications of the ACM *43*.

Goel, A. and Estrin, D. 2003. Simultaneous optimization for concave costs: single sink aggregation or single source buy-at-bulk. In *Proceedings of the ACM Symposium on Discrete Algorithms (SODA)*.

Guha, S. and Khuller, S. 1998. Approximation algorithms for connected dominating sets. *Algorithmica 20,* 4.

Gupta, H. 1997. Selection of views to materialize in a data warehouse. In *Proceedings of the International Conference on Database Theory*.

Gupta, H. 1999. Selection and maintenance of materialized views in a data warehouse. Ph.D. thesis, Stanford University.

Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D. E., and Pister, K. S. J. 2000. System architecture directions for networked sensors. In *Architectural Support for Programming Languages and Operating Systems*. 93–104.

Intanagonwiwat, C., Govindan, R., and Estrin, D. 2000. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*.

James Reserve Data Management Systems. http://dms.jamesreserve.edu/.

Kay, S. M. 1998. *Fundamentals of Statistical Signal Processing: Detection Theory, Vol. II*. Prentice-Hall.

Laouiti, A., Qayyum, A., and Viennot, L. 2002. Multipoint relaying: An efficient technique for flooding in mobile wireless networks. In *Proc. of the Hawaii Intl. Conf. on System Sciences*.

Marco, D., Duarte-Melo, E., Liu, M., and Neuhoff, D. L. 2003. On the many-to-one transport capacity of a dense wireless sensor network and the compressibility of its data. In *Proceedings of the International Workshop on Information Processing in Sensor Networks (IPSN)*.

National Climatic Data Center. www.ncdc.noaa.gov/cgi-bin/res40.pl?page=gsod.html.

Pattem, S., Krishnamachari, B., and Govindan, R. 2004. The impact of spatial correlation on routing with compression in wireless sensor networks. In *Proceedings of the International Workshop on Information Processing in Sensor Networks (IPSN)*.

Scaglione, A. and Servetto, S. D. 2002. On the interdependence of routing and data compression in multi-hop sensor networks. In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*.

Shnayder, V., Hempstead, M., Chen, B., Allen, G. W., and Welsh, M. 2004. Simulating the power consumption of large-scale sensor network applications. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys)*.

von Rickenbach, P. and Wattenhofer, R. 2004. Gathering correlated data in sensor networks. In *Proceedings of the joint workshop on Foundations of mobile computing (DIALM-POMC)*.

Vuran, M. C. and Akyildiz, I. F. 2006. Spatial correlation-based collaborative medium access control in wireless sensor networks. *IEEE/ACM Transactions on Networking 14,* 2.

Wattenhofer, R., Li, L., Bahl, P., and Wang, Y.-M. 2001. Distributed topology control for wireless muitihop ad-hoc networks. In *Proceedings of the IEEE INFOCOM*.

Wu, J. and Dai, F. 2003. Broadcasting in ad hoc networks based on self-pruning. In *Proceedings of the IEEE INFOCOM*.

Wu, J. and Li, H. 2001. A dominating-set-based routing scheme in ad hoc wireless networks. *Telecommunication Systems Journal 3*.

Xu, Y., Heidemann, J. S., and Estrin, D. 2001. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*.

Ye, F., Zhong, G., Cheng, J., Lu, S., and Zhang, L. 2003. PEAS: A robust energy conserving protocol for long-lived sensor networks. In *Proceedings of the International Conference on Distributed Computing Systems*.

Yoon, S. and Shahabi, C. 2005. Exploiting spatial correlation towards an energy efficient clustered aggregation technique (cag). In *Proceedings of the International Conference on Communications (ICC)*.