Efficient Data Collection and Selective Queries in Sensor Networks

Lars Kulik, Egemen Tanin, and Muhammad Umer

National ICT Australia
Department of Computer Science and Software Engineering
University of Melbourne, Victoria, 3010, Australia
{lars,egemen,mumer}@csse.unimelb.edu.au

Abstract. Efficient data collection in wireless sensor networks (SNs) plays a key role in power conservation. It has spurred a number of research projects focusing on effective algorithms that reduce power consumption with effective in-network aggregation techniques. Up to now, most approaches are based on the assumption that data collection involves all nodes of a network. There is a large number of queries that in fact select only a subset of the nodes in a SN. Thus, we concentrate on selective queries, i.e., queries that request data from a subset of a SN. The task of optimal data collection in such queries is an instance of the NP-hard minimal Steiner tree problem. We argue that selective queries are an important class of queries that can benefit from algorithms that are tailored for partial node participation of a SN. We present an algorithm, called Pocket Driven Trajectories (PDT), that optimizes the data collection paths by approximating the global minimal Steiner tree using solely local spatial knowledge. We identify a number of spatial factors that play an important role for efficient data collection, such as the distribution of participating nodes over the network, the location and dispersion of the data clusters, the location of the sink issuing a query, as well as the location and size of communication holes. In a series of experiments, we compare performance of well-known algorithms for aggregate query processing against the PDT algorithm in partial node participation scenarios. To measure the efficiency of all algorithms, we also compute a near-optimal solution, the globally approximated minimal Steiner tree. We outline future research directions for selective queries with varying node participation levels, in particular scenarios in which node participation is the result of changing physical phenomena as well as reconfigurations of the SN itself.

1 Introduction

Efficient data collection and aggregation algorithms for sensor networks (SNs) exploit the fact that a sensor node consumes significantly less energy for information processing than for communication. Aggregating information at the node level such as computing the sum or the average of sensor readings reduces the need for communication: instead of transmitting the packets of each individual node separately, a node first aggregates the incoming packets of the nodes in communication range and then communicates the aggregated information to the next node in the collection path.

S. Nittel, A. Labrinidis, and A. Stefanidis (Eds.): GSN 2006, LNCS 4540, pp. 25–44, 2008.

Major in-network data processing techniques for SNs do not take an explicit position on the issue whether or not a query predicate selects all the nodes of a SN. Instead, most techniques implicitly assume that a query requires *all* nodes to respond. We refer to the sensor nodes that report their readings to a selective query as *participating nodes*. An example of a *selective query* is "SELECT the humidity readings FROM all sensors WHERE the temperature is above 40° for a DURATION of 2 hours EVERY 5 minutes". We call all nodes that fulfill the WHERE clause the participating node set of this query.

Current SN data management models such as Cougar and TinyDB view the SN as an ever growing relation(s) of tuples that are distributed across the sensor nodes [1,2]. They mimic classical relational database management systems. In classical systems query predicates limit the number of tuples that form the output relation. The query predicates in these models also serve to limit the set of sensor nodes that contribute to the answer of a query. Techniques developed for efficient data collection in SNs, in particular classical tree-based and multipath-based techniques [3,4], generate a nearly optimal number of messages for aggregation operations if all nodes need to report to a query. We show that these major in-network data processing schemes do not continue this optimal behavior for selective aggregate queries.

Most approaches for data collection, do not explicitly address query selectivity while computing an efficient data collection path. There are two main directions in SN query processing for optimizing the data collection process for selective queries: (1) preventing that a query is sent to nodes that do not fall into the scope of that query and, therefore, are not aware of the query and do not need to respond, and (2) minimizing the number of non-participating nodes in the collection path. An example for the first direction is the concept of semantic routing trees [5] where optimization between queries is the main focus. In this paper, we address the second direction. The main contribution of our research is a strategy that minimizes the number of nodes used in processing a query by discovering constrained regions to grow sub-trees for data collection and combining these trees in an efficient manner to transmit the final result to the sink. This strategy contrasts with earlier tree-based approaches, such as TAG, where the tree is created in a random manner using local greedy parent selection policies [1].

Our algorithm, called Pocket Driven Trajectories (PDT) is based on the insight that spatial correlation in sensor values coupled with query selectivity results in a set of participating nodes formed by one or more geographically clustered sets. We refer to these geographical clusters as *pockets*. The PDT algorithm first discovers the set of pockets for a given query and then aligns the aggregation tree to the spatially optimal path connecting these pockets. This path maximizes the use of participating nodes in the tree and conversely minimizes the number of non-participating nodes. The PDT algorithm is best suited to the selective queries that regularly collect data from a relatively consistent set of nodes over time. The initial set-up cost for the PDT algorithm can be amortized over the query lifetime.

The task to minimize the number of non-participating nodes in processing a query can be modeled as a minimal Steiner tree problem; which is known to be NP hard [6]. The number of nodes used in the globally approximated minimal Steiner tree for a given set of participating nodes can be seen as a near-optimal solution for the minimum number of nodes required for data collection. There are a number of algorithms to

compute an approximation of the minimal Steiner tree efficiently [7,6,8]. These approximation algorithms, however, require global knowledge of the communication graph. This knowledge is typically not available in a SN. For a SN we require techniques that do not rely on global information about the network. The PDT algorithm can be considered as an approximation to the minimal Steiner tree that is solely based on local information. Our experiments show that for selective, aggregate queries, PDT does not only give better performance results than major in-network aggregation schemes but is also close to a globally approximated minimal Steiner tree.

A chief contribution of our work is the performance comparison of PDT and other major aggregation algorithms using extensive simulations in a number of node participation scenarios. We identify important parameters to capture the spatial properties of a node participation scenario and use these as a basis for our analysis. We show in our experiments that selectivity-awareness can reduce the usage of a large number of non-selected nodes in the data collection path, in particular, if the query selectivity is high.

Finally, we present future research directions where node participation is affected by changing physical phenomena as well as the SN configuration.

2 Related Work

Three major techniques for efficient data collection and aggregation are *packet merging*, *partial aggregation*, and *suppression*. The communication cost in wireless networks is determined by the number of packets a node has to transmit. Each packet has a fixed size and the size of a sensor reading, record, is typically much smaller. Packet merging [2] combines several smaller records into a few larger ones and thus reduces the number of packets and thereby the communication cost. Partial aggregation computes intermediate results at the node level such as the sum or the average of sensor readings. Suppression-based techniques only transmit values if the sensed values have changed from the previous transmission or differ from the values of neighboring sensor nodes [9]. Orthogonal techniques minimizing communication are compression techniques [10], topology control, or approximation of sensor readings.

TinyDB [1] enables in-network aggregate query processing using a generic aggregation service called TAG (Tiny AGgregation) [3]. TAG is one of the pioneering tree-based in-network aggregation schemes. To gather data within the SN, the sink is appointed to be the root of a tree and broadcasts its identifier and its level. All nodes that receive this message without an assigned level, determine their own levels as the level in the received message incremented by one. The identifier in the message determines the parent for the nodes receiving the message. In a lossless network in which all nodes are selected by a query, the resulting collection tree is close to an optimal solution. Aggregation in TAG is implemented by a merging function, an initializer, and an evaluator, and the aggregation operator is applied at every internal node.

In order to further optimize the data aggregation process, TinyDB introduces the concept of a semantic routing tree (SRT) [1,5]. Although the concept of selectivity is not explicitly addressed in TAG [3], SRTs can support selective queries. The basic idea behind SRTs is to proactively prune the SN in the query dissemination stage by ensuring

that a selective query is sent only to those nodes that fall under its scope. SRT maintains meta-information at each internal node of the aggregation tree. More precisely, an SRT is an index over a fixed attribute A, for example the temperature sensed by the network, where each parent maintains the range of all values of its children for the attribute A. When a query is received by a parent, it forwards the query only when at least one child satisfies the predicate. An SRT optimizes the query forwarding phase of TAG and greatly reduces the number of broadcasts required to reach the nodes selected by the query. To avoid a high cost of maintenance, SRTs are designed for constant attributes such as the temperature or the location [1]. However, the maintenance cost of SRT can exceed its benefit if it has to be maintained for frequently varying attributes [1]. SRTs do not focus on the data collection optimization but on the broadcast of the query.

Tree-based aggregation schemes can be extended for changing network conditions [11]: aggregation operators are pushed down in an aggregation tree and adapt to changing conditions, such as a sub-tree that generates more readings than a sibling. This approach incrementally improves on existing schemes. In our work, we develop an aggregation scheme that, after retrieving the initial readings from a SN, specifically tailors the collection path to the sensor readings and the set of selected nodes in a SN.

Data collection paths are susceptible to link and node failures in a SN [4,12]. If a link or node fails that is close to the sink, the aggregated information of an entire sub-tree might be lost. Multi-path aggregation algorithms exploit the benefits of the wireless broadcast advantage that all nodes in communication range can hear a message and propagate the aggregates toward the sink using multiple routes. As a consequence data collection along multiple paths can be more robust for node failures or communication losses. In return, a multi-path aggregation algorithm has to cope with redundancy and deviations in data aggregation [13].

In [4] multi-path aggregation algorithms are seen as energy efficient as tree-based ones because each node only has to transmit a message once, in the same way as in any tree-based aggregation algorithm. However, a crucial assumption is that the receive time for each communicating sensor is not increased by multiple readings. Recent studies on the energy consumption of sensor nodes report that the energy requirement for the receive mode is only slightly lower compared to the energy cost for the transmission mode [14], i.e., a sensor node consumes almost the same amount of energy in receiving data as in transmitting data. In multi-path aggregation, each node expects to receive data from all nodes in communication range that have a higher level than the listening node. Therefore, the duration for receiving data can be considerably longer compared to a tree-based aggregation algorithm, where each node only has to listen to its direct children, a set that can be a significantly smaller. In addition, we show that with decreasing participation levels for a selective query, the energy cost for multi-path schemes can be quite high than a tree-based scheme or our PDT algorithm.

To overcome the higher energy costs, an approach that locally applies multi-path aggregation locally but not to the entire network is suggested in [15]. This approach is a hybrid aggregation scheme that combines the benefits of the two major aggregation schemes to form a more efficient option than pure multi-path aggregation schemes: a multi-path-based aggregation scheme is preferred to a simple tree-based aggregation scheme when the in-network aggregation operator is close to the sink; for deeper levels

of aggregation tree, the operators work as if they are on a TAG-like aggregation tree as the loss of a sensor at deeper levels only marginally effects the final result.

A key challenge for multi-path routing schemes is to develop duplicate-insensitive algorithms for each aggregation operator. A naive approach could include meta information in each aggregated message such as the node identifier that participated in the creation of an aggregate, which could be used by forwarding nodes to suppress duplication. Although this approach could achieve the same accuracy as a tree-based aggregation algorithm, the limited storage and processing capabilities of sensor nodes, however, render such a scheme impractical for larger SNs. Thus, all multi-path schemes integrate much cheaper probabilistic Order and Duplicate Insensitive (ODI) methods of the sketch theory [4,13]. Therefore, current research in multi-path aggregation focuses on the development of better ODI algorithms to reduce the approximation error.

Data collection in SNs can be optimized using spatial and temporal suppression-based techniques [9]. Temporal suppression is the most basic method: the transmission of a sensor reading from a node is only allowed if its value has changed from last transmission. Spatial suppression includes methods such as clustered aggregation and model-based suppression [16]. They aim to reduce redundant transmissions by exploiting the spatial correlation of sensor readings. If the sensor readings of neighboring sensor nodes are the same, the communication of those sensed values can be suppressed. With CONCH [9], a hybrid spatio-temporal suppression algorithm, is introduced, that considers the node readings and their differences along the communication edges to suppress reports from individual nodes. Suppression is an orthogonal data flow optimization method to our approach and can easily integrated into other approaches.

Clustered in-network aggregation is a spatial suppression technique exploiting the spatial correlation of sensor readings to preserve energy [17,18,19]. Spatial correlation in sensed data refers to the fact that sensor readings in close proximity are typically similar. Spatial correlation is a frequent phenomenon for physical phenomena such as temperature or humidity [20]. If a selective query has to retrieve an aggregate such as the average temperature in a certain area, then nearby nodes typically have similar readings and are geographically clustered. Hence, only one node needs to respond to an aggregate query from a cluster [18,19] as in the Clustered AGgregation (CAG) and the Geographic Adaptive Fidelity (GAF) approaches. In static clustering [17], the network is statically partitioned into grid cells. For each grid cell one node is appointed as a cluster head that acts as a gateway but every node that has to respond to a query still reports its readings. In each cell, data is routed via a local tree, aggregated at the local gateway and then communicated to the sink. Methods that rely on approaches such as clustered in-network aggregation (such as CAG) have the disadvantage that the reported results can deviate from the real sensor readings. However, static clustering does not have this issue and is comparable to our work. Unlike our work, static clustering does not tune the collection paths to the specific network conditions.

In [21], the optimal data collection problem in SNs has been identified as a Steiner tree problem. The authors propose a data collection scheme based on a global Steiner tree approximation [8]. The main disadvantage of their approach is the requirement that each sensor node must have global knowledge in terms of network connectivity and minimum-hop routes, a knowledge that is costly and difficult to maintain in a SN where

nodes have very limited capabilities. Each update of the global knowledge about such a graph at each node leads quickly to unnecessary storage and communication overheads. The PDT algorithm uses only local information that is easily available to any node in the network. Furthermore, the PDT collection path is query specific, i.e., no permanent information has to be kept or maintained.

3 A Location Based Aggregation Algorithm

Queries such as the example query in Section 1 that identify all locations with a temperature higher than 40° represent an important type of SN queries. In a fire monitoring system, this query could be used to collect data about the humidity levels in a large region in order to identify areas that have a severe fire risk. Queries of this type run for a period of time and periodically retrieve selected sensor readings from the SN. We refer to these queries as *selective recurrent queries*.

Selective queries, in contrast to an exhaustive or a snapshot query, can benefit from an optimized data collection strategy. In snapshot queries, where the sink cannot predict the number or the location of participating nodes in advance, it can be ineffective to spend resources on the discovery of the participating nodes. In exhaustive queries, where all nodes have to respond to a query, a random tree such as TAG often provides a near-optimal data collection strategy. However, in selective queries, we show that it is beneficial to invest resources for identifying nodes that do not need to be part of the data collection especially if the query involves many sampling periods, or epochs. The initial cost for an optimized aggregation path can later later be amortized during the life time of a query. Thus, selective recurrent queries are the primary motivation for the PDT algorithm.

In general, it is not possible to find a data collection and aggregation strategy that only employs those nodes that need to participate for a given query. Due to the constraints on the transmission range for sensor nodes, a data gathering algorithm usually has to include some nodes that are not selected by the query in order to reach the sink. The quality of a data aggregation scheme is determined by the total number of non-participating nodes used in a query. The smaller the number of non-participating nodes, the more energy efficient an algorithm will generally be. In the PDT algorithm, we minimize the number of non-selected nodes in the data collection structure by spatially restricting the aggregation path to a corridor that connects the pockets in an energy efficient manner.

The problem of reporting aggregates back to a sink by involving minimum number of non-participating nodes can be seen as an application of the minimum Steiner tree problem: given a graph $G = (V, E)$ and a set of terminal nodes $T \subset V$, we seek a minimum cost spanning tree that includes all terminal nodes [22]. In our case, the terminal nodes are the participating nodes. The minimum Steiner tree problem is known to be NP-hard and has been widely discussed in the literature [6].

In order to compare aggregation schemes with the optimal aggregation tree, the minimum Steiner tree, we outline a popular global Steiner tree approximation algorithm developed by Kou, Markowsky, and Berman [7] (henceforth referred to as KMB). The KMB algorithm allows us to compute a Steiner tree that has been shown to achieve a

mean efficiency that is not worse than 5% compared to the optimal Steiner tree [6,23]. For a graph G and a set of terminal nodes T, the KMB algorithm first computes a complete distance graph $G' = (V', E')$ for G such that $V' = T$ and the weight of each edge e' in E' is the cost of the minimum cost path between the nodes of e' in G. Then, the algorithm computes a minimum spanning tree MST' of G', translates MST' to G by substituting every edge of MST' with the corresponding path in G, and finally removes any possible cycles resulting from the translation.

The KMB algorithm is not directly applicable to SNs because, the algorithm requires global knowledge of the node connectivity for any node in the graph. Therefore, we develop a *localized* aggregation scheme, called PDT (pocket driven trajectories), that approximates the minimal Steiner tree for a selective query. The experiments in Section 4 show that the resulting aggregation tree is comparable to KMB's global approximation.

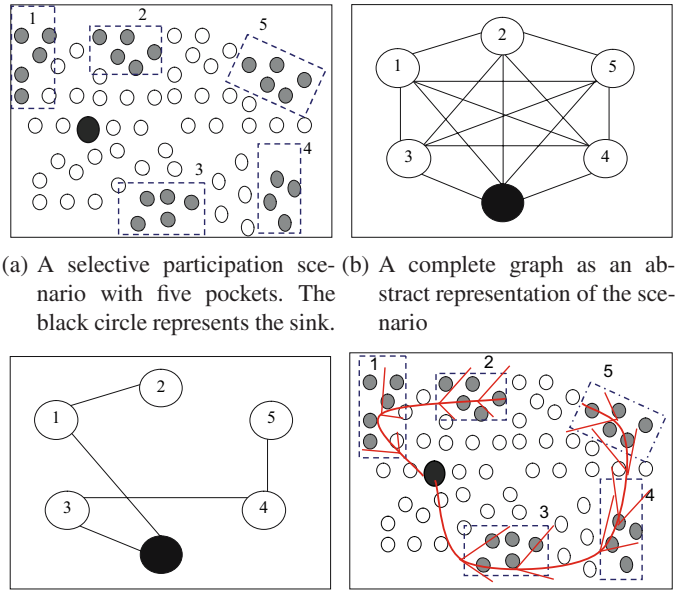
3.1 Algorithm Overview

We assume that a SN with n nodes is represented by a connected unit disk graph $G = (V, E)$, where V is the set of sensor nodes and E the set of communication links. Each query Q issued by a sink S selects a subset T of V. The PDT algorithm works as follows (Figure 1):

1. the sink broadcasts the query Q and establishes a tree as in TAG;
2. during the first epoch the sink discovers a set of pockets $P = \{p_1 \dots p_k\}$ that partitions the set T;
3. the sink computes a complete graph $G' = (V', E')$, where $V' = P \cup \{S\}$ and each edge weight is the Euclidean distance in the SN;
4. the sink computes the minimum spanning tree MST' of G';
5. the sink establishes an aggregation tree aligned to MST';

One of the key steps in the PDT algorithm is the localized discovery of pockets by the sink. A pocket is a cluster of nodes selected by a query that are proximal, i.e., within a certain distance. Due to spatial correlation, pockets are common while sensing physical phenomena. We refer to the time between the two sampling operations of a query as an epoch following the terminology used in data aggregation for SNs. In the following, we describe a novel pocket discovery method, *location aggregation*, that computes pockets with minimal overhead.

Figure 1 illustrates possible pockets that are selected by a query issued at a sink. The sink broadcasts the query and all sensor nodes build a random query tree. In the first epoch, participating leaf nodes start the aggregation phase by sending the requested sensor readings to their parent nodes and by piggybacking their location information as Euclidean coordinates. Parent nodes recursively apply the query and location aggregation operators and forward the partial aggregates to their parent nodes. The aggregation operation proceeds as in classical data collection schemes. The key step is the location aggregation, performed in parallel to data aggregation. The location aggregation operator merges locations to an enclosing rectangle if the children nodes are proximal leaf nodes, and forwards the non-proximal nodes as singletons. If a participating node receives a rectangle, it merges the rectangle with its own position into a larger rectangle,



(a) A selective participation scenario with five pockets. The black circle represents the sink.

(b) A complete graph as an abstract representation of the scenario

(c) MST'

(d) MST' as an aggregation path

Fig. 1. The computation of the PDT for a selective participation scenario

if its position is proximal, and otherwise simply forwards the existing rectangle and singleton locations with its own location. Since the location information is piggybacked with the desired data, we expect that the location aggregation incurs a small overhead in terms of communication costs. Moreover, the successive merging of close pockets keep the volume of information small.

At the end of first epoch, the sink receives the queried aggregate and after applying location aggregation operation discovers the pockets ($p_1 \ldots p_k$) selected by the query. The sink then computes a complete graph G' as explained in the overview of the PDT algorithm (see Figure 1(a), 1(b)). The algorithm then creates a minimum spanning tree MST' for G' at the sink. MST' is a pocket driven trajectory that optimizes aggregation for the specific pocket layout (Figure 1(c)). The PDT information can be encoded as a series of locations, each corresponding to either a sink location or the center point of one of the pocket rectangles. During the next epoch, the sink establishes the PDT by broadcasting the PDT information to all its direct children. All participating nodes that receive the PDT information packet join the trajectory by reassigning their parents to that node from which they receive the PDT information. Non-participating nodes decide to join the trajectory depending upon their distance to the trajectory and only nodes that join the trajectory forward the information packet. The successive forwarding and parent switching leads to a new aggregation tree aligned with the PDT (Figure 1(d)). This new tree is afterwards used for data forwarding and aggregation. The initial TAG tree is still maintained because in future epochs previously non-participating nodes may

participate. Future participating nodes might have never heard the PDT information and thus have to use the original tree for data aggregation.

3.2 Shortcomings and Overheads

A query in a SN can consist of a large number of epochs. Even if the change per epoch is small, the node participation can change significantly during the lifetime of a query. The PDT algorithm is ideal if the change per epoch is relatively small so that the pockets do not change significantly in every step. Under those conditions, the PDT may have to be realigned a few times during the lifetime of a query. However, currently the PDT algorithm does not adapt to change within a query. We leave the investigation of more dynamic scenarios to future work.

The increase in packet size due to the aggregation of location information increases the communication overhead. The location information consists of aggregated pocket information and a list of atomic locations. Due to the spatial correlation of physical phenomena, the number of atomic locations is typically small and singletons mostly occur at the leaf level of the aggregation tree. The singletons are almost completely merged at the lower levels of the tree into pockets that traverse for the remainder of the tree in a compact form as rectangles. Our experiments show that the location information aggregation only slightly increases the communication messages.

The announcement of the PDT is the other overhead of the PDT algorithm. The number of extra messages generated during PDT information broadcast phase is equal to the number of nodes that decide to join the PDT. The initial setup cost can be amortized over the query lifetime. However, the initial cost cannot be amortized for snapshot queries and hence we do not recommend the use of the PDT algorithm for such queries.

4 Experimental Evaluation

4.1 Evaluation Parameters

In this section we compare the performance of the PDT algorithm and other major in-network aggregation schemes in a variety of SN settings. We first lay out the evaluation parameters that we use to analyze the impact of spatial characteristics of selective aggregate queries.

Spatial Selectivity Index. The nature and extent of spatial clustering of participating nodes may depend upon a number of factors, such as the magnitude of spatial correlation, SN configuration, the query predicate, and so forth. We introduce an integrated measure, called spatial selectivity index, SSI, that describes the extent of spatial clustering in a SN. SSI is based on two key parameters, *node scattering* and *pocket scattering*.

The *node scattering*, NS, at time t (we will drop the argument t for the sake of simplicity) describes the distribution of pockets for a query as the ratio of the total number of pockets P relative to the number of participating nodes N:

$$\text{NS} = P/N$$

The *pocket scattering*, PS, characterizes the degree of dispersion for the pocket locations. We define the *pocket centroid*, PC, of a pocket P_i as the sensor node that is closest to the average location of all nodes belonging to P_i. We then define the *global pocket centroid*, GPC, as the node that is closest to the centroid of all pocket centroids $P_1, \ldots, P_l$ partitioning the nodes selected by the query:

$$\text{GPC} = \frac{1}{l} \cdot \sum_{i=1}^{l} \text{PC}(P_i)$$

Let $\text{HC}(v, v')$ denote the minimum number of hops for a path connecting two nodes v and v'. Then, the pocket scattering PS is defined as the average of hop counts connecting the global pocket centroid with the pocket centroids of the pockets $P_1, \ldots, P_l$:

$$\text{PS} = \frac{1}{l} \cdot \sum_{i=1}^{l} \text{HC}(\text{PC}(P_i), \text{GPC})$$

We use the hop count between two nodes as a distance measure instead of their Euclidean distance. Since deployments of SNs can exhibit holes and barriers, the hop count provides a realistic approximation of the actual communication cost. It should be noted that the use of hop count as a distance measure is purely for evaluation purposes. The PDT algorithm itself does not use the hop count measure due to the practical limitations in making such information available locally at each node.

The spatial selectivity index SSI is then defined as a measure that describes the impact of node scattering as well as of pocket scattering for a given scenario:

$$\text{SSI} = \text{NS} \cdot \text{PS}$$

Lower SSI values characterize scenarios that are well pocketed and have a small pocket scatter, for example see Figure 3(a) and 3(b) in Section 4.3 that show two network deployments with different SSI values.

Sink Centroid Distance. We formalize sink position in order to analyze the affect of sink location on the performance of an aggregation algorithm. The ideal position of a sink is the GPC. The *sink centroid distance*, SCD, measures the hop count between the sink and its ideal position. If S is the position of the sink, the sink centroid distance SCD is defined as

$$\text{SCD} = \text{HC}(S, \text{GPC})$$

Co-Connectivity of a Deployment. Although a rectangular deployment is common in simulations, e.g., without any holes that alter connectivity, in this paper, we also consider the impact of irregular deployments on aggregation algorithms. Particularly, we measure the impact of holes. To simplify our discussion, we only take the total number of holes and their normalized size into account. If $|\mathcal{R}|$ denotes the size of the deployment area $\mathcal{R}$, then the normalized size of a hole H_i is $|H_i|/|\mathcal{R}|$. The *co-connectivity measure* CC is then defined as the sum of the normalized sizes of all holes:

$$\text{CC} = \sum_{i} |H_i|/|\mathcal{R}|$$

4.2 Simulation Setup and Methodology

In addition to PDT, we implement comparable aggregation algorithms discussed in Section 2. We implement all algorithms in Network Simulator-2 (NS-2) [24]. To provide a lower bound, we compute for each experiment an approximation of the optimal aggregation tree using the KMB algorithm. In our simulations, we collect the AVERAGE on a deployment of 750 nodes, placed randomly in a 75m x 75m grid. Each query collects data from a SN for 100 epochs. We utilize the NS-2 wireless communication infrastructure that simulates 914 MHz Lucent Wave LAN DSSS radio interface using the two ray ground reflection propagation model and IEEE 802.11 MAC layer (Chapters 16 & 18 of the NS-2 Manual [24]). Communication is performed using omni-directional antennas centered at each node, while the communication radius is fixed at 5m. The message payload is fixed at 72 bytes and we assume that every algorithm has the same payload for data transfers. Furthermore, we assume a lossless network with synchronized many-to-one aggregation, i.e., during in-network aggregation each internal node is perfectly synchronized with its children and after aggregation it always emits just one packet.

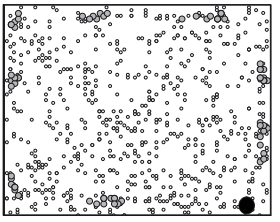
We use total data transmission (in MBs) as an indication of energy usage and hence as the basic metric of performance comparison. The amount of data transmission can be related to the energy expenditure by a simple function such as $\varepsilon = \sigma_s + \delta_s x$, where ε is the total amount of energy spent in sending a message with x bytes of content, and σ_s and δ_s represent the per-message and per-byte communication costs, respectively [9].

In order to systematically study the impact of varying participation levels and selective participation measures as defined in Section 4.1, we design our experiments according to the following questions:

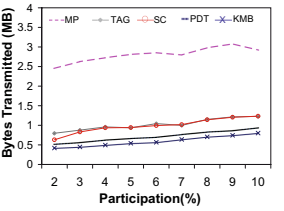
- What is the impact of query selectivity and level of spatially correlated node participation on the performance of aggregation algorithms?
- What is the impact of the position of pockets and their dispersion in selective participation scenarios?
- What is the impact of the location of the sink on the performance of an aggregation algorithm in low node participation levels?
- What is the impact of (communication) holes on data collection?

4.3 Results

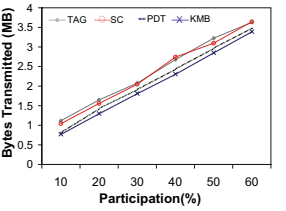
Impact of Query Selectivity. In two experiments, we investigate the impact of the query selectivity on four different aggregation schemes: PDT, multi-path (MP), static clustering (SC), and TAG. In the first experiment, Figure 2, we change the selectivity of an aggregate query and hence the number of nodes that participate in a query by 1% increments from 2% to 10% of the total nodes in the SN. The participating nodes are spatially clustered (see the deployment snapshot in Figure 2(a)). Figure 2(b) shows the mean value of the number of bytes transmitted by each algorithm at each participation level (the average of five runs is used to find the mean value). Figure 2(c) shows results from a similar experiment with participation levels ranging in discrete steps from 10% to 60%.



(a) A spatial configuration for partial node participation. Larger circles represent nodes that participate in a query. Black circle represents the sink.



(b) Detailed comparison of data transmissions for each algorithm in low participation scenarios (the participation level ranges between 2%–10%)



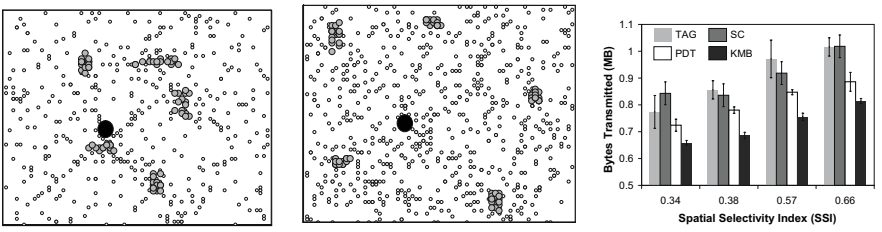
(c) The overall trend for each algorithm in terms of data transmissions (the participation level ranges between 10%–60%)

Fig. 2. The performance of aggregation techniques for varying levels of partial node participation

Figure 2(b) shows that for low node participation levels, PDT performs better than other aggregation schemes. For participation levels from 2% to 10% PDT is, on average, 41% more efficient than TAG and 37% more efficient than SC. In addition, PDT is just 21% less efficient than the approximated lower bound, where TAG and SC are 72% and 67% less efficient, respectively. This experiment also reveals that the energy consumption for MP in low participation scenarios is significantly higher than all other aggregation algorithms. MP requires at least 2.7 and 2.8 times as much data transmission as TAG and SC, respectively and 3.8 times more than PDT. Similarly, the trend in Figure 2(c) shows that PDT remains energy efficient even for high participation levels but its advantage decreases as the participation levels increase. At 10% participation, PDT requires 30% less data transmissions than TAG and 31% less than SC; however at the participation level of 60% this lead reduces to 4% and 5% for TAG and SC, respectively. The decrease in efficiency results from the fact that with the increase in node participation the benefit of spatial correlations diminishes. This effect can also be observed from the fact that at 60% participation level PDT is just 3% less efficient than the KMB lower bound. For high participation levels, MP is not shown on the figure to simplify the presentation.

Impact of Varying the Spatial Selectivity Index. This section describes a set of experiments that assess the performance of the PDT algorithm in various spatial layouts, characterized by different SSI values. A low SSI value represents a low dispersion scenario. We expect a better performance from PDT and other aggregation algorithms for selective queries with low SSI values. In this experiment, we achieve the effect of increasing SSI values by expanding the dispersion of pockets in the network, while the total number of pockets, the node participation level, and the sink position remains constant. Figure 3(a) and 3(b) show the deployment snapshots of two scenarios.

Figure 3(c) confirms the hypothesis that all algorithms perform better for lower SSI values (standard deviations are also shown in this chart). As the dispersion of the participating nodes increases, all algorithms have to spend more energy. For the analyzed



(a) A partial node participation scenario with an SSI of 0.34. Black circle represents the sink.

(b) A partial node participation scenario with an SSI of 0.57

(c) The impact of an increasing SSI on the performance of aggregation algorithms for a 10% node participation level. Standard deviations are shown.

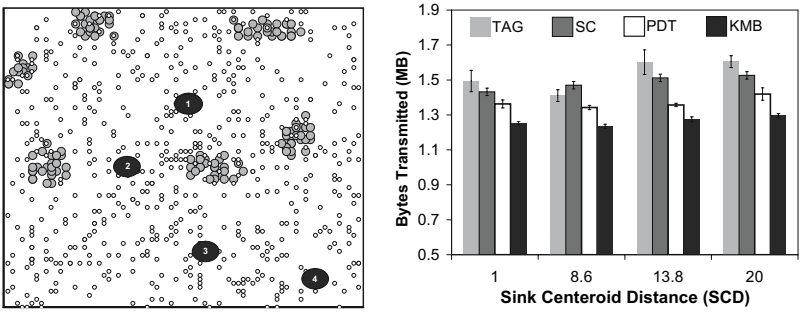
Fig. 3. Various network configurations simulating an increase in the spatial selectivity index by increasing the pocket dispersion for a 10% node participation level

scenarios, TAG and SC transmit up to 31% and 22% more data for the highest SSI value. PDT also generates more data and shows an increase of 22% for the highest SSI value, however it remains 15% energy efficient than both TAG and SC.

Impact of the Location of the Sink. In this experiment we analyze the effect of sink position on PDT and other aggregation schemes. Figure 4 shows the performance of each algorithm in a deployment where the same query is issued from different sink positions. The chart shows that, for the given deployment, different sink positions affect the overall cost only modestly: between initial and final sink position data transmission rises by just 7% for both SC and TAG, while it rises to only 4% for PDT. The average change in cost from one scenario to next is 1%, 2% and 3% for PDT, SC, and TAG respectively.

The result is not surprising for the SC and PDT algorithms. In PDT the aggregation tree is mostly determined by the pocket locations while the impact of sink location is limited to the distance between the sink and the pockets closest to it. Similarly, SC always uses fixed paths to aggregate data inside each cluster and the impact of sink location is limited to the final phase where cluster heads have to route the aggregated data to the sink. The bulk of data transmission in both cases occurs inside pockets (or clusters) and as a result the impact of the position of sink is reduced. However, it is important to note that the behavior of TAG fluctuates with the sink location. Among the simulated scenarios, the second sink position is the best for TAG. At this position, the sink is located in a way that paths to distant pockets naturally emerge from the closer pockets, resulting in an increased number of participating nodes acting as intermediate nodes in the tree. In other scenarios, the misalignment of the root of the tree, sink, with the clusters increases the cost for TAG.

Spatial Layout and Communication Holes. Real SN deployments cannot stay fully connected in a regular grid structure although many routing and in-network aggregation



(a) The spatial distribution of the network configuration and the various positions of the sink

(b) The performance of the aggregation algorithms for different sink positions with an increasing distance to the global centroid position for a 20% node participation level

Fig. 4. A network configuration with different sink positions for a 20% node participation level

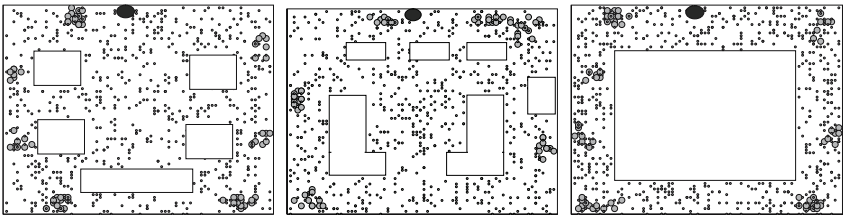
algorithms are commonly tested on such basic structures. Due to constrained communication capabilities, a network might be disconnected at certain places leaving gaps that we name as communication holes. In the context of in-network aggregation, if a given network suffer from communication gaps while still remaining connected via alternate communication routes, it is of interest to understand how the presence of holes effect the performance of an aggregation algorithm.

To investigate the effect of communication holes we simulate three different SN configurations. The configurations are shown in the deployment snapshots in Figure 5(a)–Figure 5(c), where the bordered regions represent communication holes. In each of the deployments, we set up a 10% pocketed participation scenario and Figure 5(d) shows the performance of each algorithm in these spatial layouts. In TAG, we see that a collection of communication holes can affect the formation of the aggregation tree in one of two ways. Firstly, the holes might break the most direct communication paths to pockets and hence the tree has to invariably take a longer route. This effect can be observed from the high cost of TAG in the first deployment (Figure 5(a)). However, a second more interesting scenario is where the presence of holes actually reduces the communication cost by restricting the tree into a set of corridors that naturally spans the pockets. The cost of TAG is reduced by 35% between the initial and final deployment.

SC is rather unaffected by the presence of holes where the change in its cost between the initial and final deployments is just 5%. PDT also performs almost unaffected by the presence of communication holes and shows a 10% reduction in data transmissions. PDT might suffer in cases where there is a communication gap between two neighboring pockets in the trajectory.

4.4 Discussion

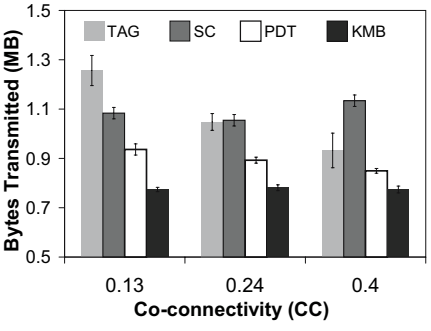
With extensive experiments, we presented the advantages of the location based PDT algorithm for in-network aggregation over well-known in-network aggregation



(a) A network configuration with deployment co-connectivity of 0.13 (five holes relatively close to the border)

(b) A network configuration with deployment co-connectivity of 0.24 (six randomly distributed holes)

(c) A network configuration with deployment co-connectivity of 0.40 (one large hole in the middle of the network)



(d) The impact of different co-connectivities on the aggregation algorithms for a 10% node participation level

Fig. 5. Various network configurations simulating deployments with different types of communication holes and a 10% node participation level

algorithms in different SN settings. Our results validate the hypothesis that a variable node participation scenario affects the performance of existing algorithms. In addition, the spatial features of the scenario do also have an effect on the performance.

The performance of an algorithm in processing selective queries can be presented as a function of the number of nodes the data collection paths utilize while collecting data from the participating nodes. The high cost of the tree-based scheme in highly selective queries can be explained by the random strategy used in the creation of an aggregation tree where no query specific optimization is considered during the tree construction process. We observe that for low node participation levels, the tree can improve its performance when the query sink is aligned with the data pockets in a way that paths to distant pockets naturally emerge from the pockets that are close to the sink. Similarly, a tree-based strategy shows considerable improvement in performance if the communication channels in the network are constrained by holes. However, in realistic settings, such cases may be rare and may not justify building a random tree for low participation scenarios. In contrast to such special cases, PDT always identifies constrained regions to grow a collection path in an efficient manner. Since the data collection is optimized to

minimize the number of non-participating nodes en-route to data sources, PDT shows an overall reduction in data transmission even in high node participation scenarios. The experiments also show that the cost of PDT rises with increasing participation levels or decreasing spatial correlation levels, however for long running queries with many epochs, it is at least as good as the other well-known techniques.

As an interesting result, we observe that static clustering shows comparable results to the tree-based strategy even though it is not configured as dynamically as the tree-based strategy. Similar to the tree-based strategy, we also observe that static clustering performs better if the location and size of pockets correlate with that of the statically configured clusters and hence the data collection mechanism. The static clustering algorithm proposes to define the cluster size parameter depending upon the degree of spatial correlation in the network [17]. A major challenge in static clustering hence is determining the correct cluster size for queries with complex, selective, predicates.

In our experiments where the node participation levels are low, data transmission by the multi-path algorithm has greater costs than all the other schemes. This is an interesting observation since in exhaustive participation scenarios multi-path achieves the same number of messages per node as a tree [15]. Although the goal of the multi-path algorithm is to achieve accuracy in lossy environments rather than increasing efficiency of the aggregation strategy, the large cost of multi-path in low node participation scenarios may require adopting a hybrid method such as [15].

One important feature of the PDT algorithm is that there is a trade-off between the latency of data collection and the data transfer costs for a given query. In order to minimize the number of non-participating nodes in the aggregation process, PDT creates paths with possibly higher latencies. Since in a sensor network saving energy is the primary concern, reducing response time may not impact many query types. Thus, the latency is not analyzed as a separate parameter with our experiments.

5 Selectivity Under Changing Conditions

A central task for SN deployments is the monitoring of naturally occurring phenomena. Typical examples are the monitoring of wildlife paths in a natural reserve or the study of cattle movements in a farm. These application domains highlights the need for robust algorithms for selective query processing in SNs under continuously changing conditions. We analyze two types of change: (1) change of the physical phenomena that is monitored by the SN, and (2) change of the monitoring-network itself. The movement monitoring of wildlife in a natural reserve falls into the first category, while sensor failures are an example for the second category.

Changing physical phenomena pose a challenge for methods that are designed for processing selective queries. For a given selective query, the set of sensors that need to respond may be different at every sampling period. In fact, this behavior is expected as the main purpose of monitoring tasks is to observe and record changes in the physical phenomena in the first place. Thus, methods that are tailored for processing selective queries need to be able to cope with changing sets of participating nodes that respond to a query during the lifetime of that query.

We are currently adapting and experimenting with the PDT algorithm under realistic monitoring tasks as they occur with changing natural phenomena. The main insight for processing selective queries under changing conditions is that different observed states of the physical phenomena by a SN are in fact temporally as well as spatially correlated. Hence, the sensor readings do not change abruptly and randomly between two sampling periods but instead change occurs in relation to a source (or a set of sources), which leads to predictable patterns. A typical example is temperature measurements: the sensed temperature values are related to the Earth's revolution around the Sun. Similarly, cattle usually does not move randomly, but acts as a herd and moves within the constraints of the pasture, preferring certain places during day and night. Thus, we expect that finding efficient data collection paths under changing participation levels can be optimized using an approach that is based on PDT and includes predictive models. For low frequency sampling, the change between two periods could hide its behavior. However, as users are interested in capturing this behavior, the frequency of observations will be adjusted accordingly.

We envision two directions in adapting selectivity-aware algorithms such as PDT to monitor continuously changing physical phenomena: first, a reactive model that would only alter the data collection paths after a certain amount of change occurs in the sensor readings; second, a proactive model that predicts the changes and adjusts the collection path before the actual change occurs. A reactive algorithm might be easier to implement and cheaper to maintain but will be less responsive to changing physical phenomena. A proactive version, on the other hand, might be more difficult to maintain and develop, but may be more adaptive to the changing phenomena. It seems likely that the overall benefits of both strategies will differ for various application domains.

In addition to the change in the monitored physical phenomena, network parameters under which a SN operates can also change. We distinguish *uncontrolled* and *controlled* changes. An example for the first category is a production fault of some of the sensors in the network. This can occur independently from the deployment strategy and the physical conditions under which the SN operates. We assume that such situations rarely create a major disruption, and more importantly, do not specifically effect selective queries significantly. Recovery from such failures could be handled by the lower layers of the sensor node's system software. Nevertheless, PDT algorithm can be easily adapted for SNs where nodes can fail. The data collection paths can be tuned between sampling periods to bypass failing nodes efficiently.

Energy depletion rates are an example of a controlled change in a SN. Certain sensors may run out of energy faster than the others in a network, because they might be used more often in data collection paths. This type of change in a network is observable. Energy-aware algorithms that can adjust to these changes for routing already exist [25]. For selective query processing, techniques such as the PDT algorithm can also be adjusted to cope with similar situations. Location aggregation can be performed in tandem with the aggregation of the energy levels. Thus, data collection paths, per query, can be tailored for different energy levels of the nodes.

If the change in a SN is the result of a sustained use of certain set of sensor nodes, another alternative could be considered: redeployment of the sensor nodes. An interesting research direction is emerging with mobile sensor networks [26,27,28] where

nodes can be redeployed for improving the quality of the data collected as well as for the efficiency of collection of data in tandem with improving the SN life expectancy.

6 Conclusions

Efficient processing of selective queries in SNs is crucial for effective sustained monitoring of physical phenomena. Existing data collection methods in SNs do not take an explicit position in processing selective queries and thus may not benefit from significant possible energy savings. In our work, we have presented the benefits of optimizing data collection paths and creating methods tailored for selective queries. We introduced the PDT algorithm, an in-network data collection method for long running selective queries. In extensive simulations, we showed that PDT is more energy efficient than other major aggregation techniques under various scenarios. We have also observed that the PDT algorithm computes a collection path to a well-known approximation of the global optimum, i.e., the minimal Steiner tree.

The efficient data collection problem in partial node participation scenarios can be modeled as a minimal Steiner tree problem. The PDT algorithm discovers pockets of participating nodes using purely local information about the network and approximates a minimum Steiner tree for data collection from these pockets. We show that this leads to significant energy savings in different node participation scenarios. In addition, we define spatial parameters to characterize a specific network deployment.

There are several research directions that we suggest as for future work. For a given query, the node participation can change over time. For example, a change in the physical conditions for the sensed environment can lead to changes in node participation. For a selective query this change can have an impact on multiple fronts, i.e., by increasing or decreasing the node participation levels, or by changing the distribution of participating nodes such as the emergence of new pockets or breakdown of old pockets. Therefore, it is important to introduce robust strategies for data collection to continuously adapt to these changes. On another front, changing network conditions can force the data collection algorithms to look for better ways of gathering data. Node failures, as a trivial case, can require algorithms, such as the PDT, to reconsider their choices in data collection paths. If the change in network conditions can be controlled, such as the energy depletion rates of different sensors, then adaptive selective query processing algorithms for controlled energy consumption should be considered. One particularly promising research direction is under scenarios when sensors have the capability of changing their location. In this case, increasing the quality of the data that is being collected in a partial node participation scenario while decreasing the data collection and node relocation costs is an interesting research topic.

References

1. Madden, S.R., Franklin, M.J., Hellerstein, J.M., Hong, W.: TinyDB: an acquisitional query processing system for sensor networks. ACM Trans. Database Syst. 30(1), 122–173 (2005)
2. Yao, Y., Gehrke, J.: Query processing for sensor networks. In: Proceedings of the Conference on Innovative Data Systems, pp. 233–244 (2003)

3. Madden, S., Franklin, M.J., Hellerstein, J.M., Hong, W.: TAG: a Tiny AGgregation service for ad-hoc sensor networks. SIGOPS Oper. Syst. Rev. 36(SI), 131–146 (2002)
4. Nath, S., Gibbons, P.B., Seshan, S., Anderson, Z.R.: Synopsis diffusion for robust aggregation in sensor networks. In: Proceedings of SenSys, pp. 250–262 (2004)
5. Madden, S., Franklin, M.J., Hellerstein, J.M., Hong, W.: The design of an acquisitional query processor for sensor networks. In: Proceedings of SIGMOD, pp. 491–502 (2003)
6. Oliveira, C.A.S., Pardalos, P.M.: A survey of combinatorial optimization problems in multicast routing. Comput. Oper. Res. 32(8), 1953–1981 (2005)
7. Kou, L., Markowsky, G., Berman, L.: A fast algorithm for Steiner trees. Acta Informatica 15, 141–145 (1981)
8. Takahashi, H., Matsuyama, A.: An approximate solution for the Steiner problem in graphs. Math Japonica 24, 573–577 (1980)
9. Silberstein, A., Braynard, R., Yang, J.: Constraint chaining: on energy-efficient continuous monitoring in sensor networks. In: Proceedings of SIGMOD, pp. 157–168 (2006)
10. Chou, J., Petrovic, D., Ramachandran, K.: A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks. In: Proceedings of INFOCOM, vol. 2, pp. 1054–1062 (2003)
11. Bonfils, B.J., Bonnet, P.: Adaptive and Decentralized Operator Placement for In-Network Query Processing. In: Zhao, F., Guibas, L.J. (eds.) IPSN 2003. LNCS, vol. 2634, pp. 47–62. Springer, Heidelberg (2003)
12. Bawa, M., Gionis, A., Garcia-Molina, H., Motwani, R.: The price of validity in dynamic networks. In: Proceedings of the SIGMOD, pp. 515–526 (2004)
13. Considine, J., Li, F., Kollios, G., Byers, J.: Approximate aggregation techniques for sensor databases. In: Proceedings of ICDE, pp. 449–460 (2004)
14. Cardei, M., Wu, J.: Energy-efficient coverage problems in wireless ad hoc sensor networks. Computer Communications 29(4), 413–420 (2006)
15. Manjhi, A., Nath, S., Gibbons, P.B.: Tributaries and deltas: efficient and robust aggregation in sensor network streams. In: Proceedings of SIGMOD, pp. 287–298 (2005)
16. Chu, D., Deshpande, A., Hellerstein, J., Hong, W.: Approximate data collection in sensor networks using probabilistic models. In: Proceedings of ICDE, p. 48 (2006)
17. Pattem, S., Krishnamachari, B., Govindan, R.: The impact of spatial correlation on routing with compression in wireless sensor networks. In: Proceedings of IPSN, pp. 28–35 (2004)
18. Xu, Y., Heidemann, J., Estrin, D.: Geography-informed energy conservation for ad hoc routing. In: Proceedings of MobiCom, pp. 70–84 (2001)
19. Yoon, S., Shahabi, C.: Exploiting spatial correlation towards an energy efficient clustered aggregation technique (CAG). In: Proceedings of the ICC, pp. 82–98 (2005)
20. Gupta, H., Navda, V., Das, S.R., Chowdhary, V.: Efficient gathering of correlated data in sensor networks. In: Proceedings of MobiHoc, pp. 402–413 (2005)
21. Krishnamachari, B., Estrin, D., Wicker, S.B.: The impact of data aggregation in wireless sensor networks. In: Proceedings of ICDCSW, pp. 575–578 (2002)
22. Robins, G., Zelikovsky, A.: Improved Steiner tree approximation in graphs. In: Proceedings of SODA, pp. 770–779 (2000)
23. Doar, M., Leslie, I.M.: How bad is naive multicast routing? In: Proceedings of INFOCOM, pp. 82–89 (1993)
24. NS-2: The network simulator NS-2 documentation, `http://www.isi.edu/nsnam/ns/ns-documentation.html`
25. Yu, Y., Govindan, R., Estrin, D.: Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Technical Report TR-01-0023, University of California, Los Angeles, Computer Science Department (2001)

26. Somasundara, A.A., Jea, D.D., Estrin, D., Srivastava, M.B.: Controllably mobile infrastructure for low energy embedded networks. IEEE Transactions on Mobile Computing 5(8), 958–973 (2006)
27. Wang, G., Cao, G., Porta, T.F.L.: Movement-assisted sensor deployment. IEEE Transactions on Mobile Computing 5(6), 640–652 (2006)
28. Hull, B., Bychkovsky, V., Zhang, Y., Chen, K., Goraczko, M., Miu, A., Shih, E., Balakrishnan, H., Madden, S.: CarTel: a distributed mobile sensor computing system. In: Proceedings of SenSys, pp. 125–138 (2006)