University of Alberta

Extending the Lifetime of Wireless Sensor Networks with Spatial Data Aggregation

by

Shoudong Zou

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

©Shoudong Zou
Fall 2009
Edmonton, Alberta

Examining Committee

To My Mom,

With all my love, I dedicate this work.

Abstract

In this thesis, we propose mechanisms to extend the lifetime of wireless sensor networks. In-network data aggregation is considered on both tree-based and flow-based routing protocols during the process of data collection to reduce redundant transmissions. In the flow-based data collection design, we introduce the concept of *flow loss multiplier* to express the impact of data aggregation over correlated data. The application has the freedom to set the flow loss multiplier to reflect its specific knowledge of correlation.

We also introduce traffic balancing as a complementary technique to data aggregation. It helps avoid exhausting the energy of any sensor node while leaving large amounts of energy at other nodes. In tree-based data collection schemes, we adjust the tree structure judiciously to balance energy consumption before any node's failure due to total residual energy depletion. In flow-based schemes, after aggregation, data flows are split and the fragments are spread to increase network lifetime.

We investigate the impact of performing greedily data aggregation at the "best" aggregation site regardless of its location, the results of our analysis show that only applying 2-way data aggregation may limit the ability to explore more complex aggregation possibilities. To address this problem, we propose an aggressive data aggregation for a specified application, contour map reconstruction. Based on the simulation results, our aggregation scheme is shown to be able to eliminate large volume of contour data and retain satisfying data accuracy.

Acknowledgements

First and foremost, I must express my sincere gratitude to my supervisors, Professor Janelle Harmes and Professor Ioanis Nikolaidis. Their guidance has been an invaluable resource throughout all stages of this thesis. I thank them for their enthusiastic support, insightful comments and advice, and all the discussions we had. I am especially grateful for their patience while this work continued longer than expected.

I own a great deal of thanks to my parents. Without their love and encouragement, I never would have come this far in the first place. They have always supported me regardless of the path I have chosen to follow. I also thank Shoubin Zou, my older brother, for his love, support and tolerance.

I thank my wife, Liping Dong for all the sacrifices she made; my son, Blake Zou, for all the fun and inspiration he brought to me.

Finally, but certainly not least, I would like to thank Dr. Yanxia Jia, Dr. Xudong Wu, Dr. Li Cheng, Dr. Yuxi Li, Dr. Qiang Ye, Dr. Xiang Wan, Dr. Zhingpeng Cai, Dr. Lihang Ying, Dr. Fan Deng, Dr. Yang Wang, Xiaozhen Niu, Baochun Bai, Jinhui Shen, Tianhao Qiu, Peng Wang, Xiaoran Cao, Zhuang Guo, Jiyang Chen, Nicholas Boers, Baljeet Malhotra, Sunil Ravinder and Qinghua Ye for their help during my PhD study.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Recent technological advances in micro-electro mechanical systems (MEMS) and wireless communications have enabled the deployment of small and low-cost sensor nodes [37]. These sensor nodes are usually equipped with limited programmable computing, sensing and wireless communication components. Sensor nodes are often randomly deployed. They are usually battery-powered and difficult, or impossible, to recharge. Therefore, dense deployment is employed to extend the spatial coverage but also to increase the fault-tolerance and robustness of the system.

Once deployed, these sensor nodes self-configure into a Wireless Sensor Network (WSN) with the objective to monitor physical phenomena of interest that occur within their area. The emergence of WSN technology is revolutionizing a wide range of applications in both civilian and military domains [37, 87, 88, 77, 61, 18, 64], such as environmental surveillance, structural health monitoring and target tracking.

In a common WSN application scenario, each sensor node periodically monitors its vicinity and reports its observation (sensor data) to the base station (sink node) for storage and further processing. Owing to the periodic data collection, time can be thought of as divided into many equal-sized epochs, and within each epoch (or "data collection round"), sensor data from each source is gathered at the sink. If the sensor nodes are not the direct neighbors of the sink node, a multi-hop routing mechanism is required and those intermediate nodes have to relay data on behalf of others to the sink in addition to reporting their own data.

1.1 Characteristics of Wireless Sensor Networks

Scarce energy supply is the most significant constraint in WSNs. Sensors are too small to carry a battery with large capacity. In addition, they are often dispersed in inaccessible areas. It is difficult to replace or recharge their batteries. Once depleted of energy, a sensor node will stop providing any support to the system, which may further lead to the termination of the functionality of the whole WSN, because each sensor is also responsible for routing. Therefore, any system design has to take this energy limitation into consideration and provide a solution for prolonging the operational lifetime of the system. Since all the sensor data will be gathered at the sink node for storage or further processing, it is assumed accessible and has unlimited energy supply.

Another feature of a sensor node is its limited transmission range. Wireless sensors are small and possibly cheap devices, which result in its limited wireless transmission capacity. Most WSN applications require that all or part of dispersed sensor nodes report their measurements. Due to the restricted transmission ability, multi-hop routing has to be utilized. Since the sink node is usually the unique site at which all sensor data has to be gathered, sensor nodes along the frequently accessed path (to the sink) have much larger traffic burden than others. This traffic load imbalance results in biased energy consumption, which brings a significant challenge to extending the system lifetime of WSNs.

Although the capability of each sensor is limited, a number of deployed sensor nodes could cooperate to accomplish a large and complex monitoring task. This collaboration typically occurs in the form of data aggregation among a subset of sensor nodes carrying similar sensor data. Due to the nature of the phenomena being monitored (usually smoothly developing in both temporal and spatial domains) and the dense sensor node distribution (to fulfil sensing coverage requirement of a given WSN application), the sensed data are highly correlated. Data redundancy among multiple similar sensor observations can be removed while they are routed to the sink node in order to save energy consumption by avoiding delivering redundant data. This approach is usually referred to as in-network data aggregation [45].

For many applications, accurate node position is important for efficient cooperation of the deployed sensors and functionality of the sensor network. Most research work about WSN assumes GPS receivers are available on all or some sensor nodes [39]. However, sometimes GPS devices may not be feasible for sensor nodes due to limited hardware cost, or they do not work well due to limitations of satellite coverage or degraded satellite signals. In this thesis, we first design GPS free data collection schemes. Then, for a particular WSN application, contour map reconstruction, we design data collection approaches based on accurate sensor node positions.

1.2 Maximizing Lifetime Data Collection with In-network Aggregation and Workload Balance

The lifetime of a WSN can be defined as the time after which a certain fraction of sensor nodes deplete their energy. In particular, in this thesis, we assume lifetime of a WSN ends when any single sensor node runs out of its energy. This is the most strict lifetime definition and it can be used as an "acid test" to verify the ability of a data gathering scheme to handle stringent requirements such as scarce sensor energy supply and high sensor network density requirement from the specified applications. The reason is that if it is impossible to recharge sensor nodes, once any single sensor exhausts its energy supply, the functionality of the whole WSN may be compromised or even terminated.

No matter which definition of the lifetime is adopted in the system design, as we discussed before, traffic load imbalance results in heterogeneous energy consumption among deployed sensor nodes leading to early cessation of operation of some part of the system. In addition, dense sensor deployment guarantees the network connectivity but also indicates a high degree of sensing coverage overlap, which introduces the possibility of strong data correlation within the network. On the other hand, it has been realized that local computation is much cheaper than wireless communication in terms of energy consumption [8]. So it is desirable to utilize in-network data aggregation to reduce data redundancy among sensor nodes that possess similar sensor observations. In this way, we trade computation for wireless traffic reduction, thus extending system lifetime.

In this thesis, we concentrate on designing data collection mechanisms to maximize lifetime by performing in-network aggregation, as well as balancing traffic load across the whole network. The former helps alleviate the heavy wireless transmission burden accumulated at those nodes close to the sink; the latter helps avoid creating areas, anywhere in the network, in which nodes run out of energy much earlier than others. Without explicit mention, throughout this thesis, data aggregation is mainly from spatial data correlation (similar readings are usually obtained by closely located sensor nodes).

1.3 Example Applications of In-network Aggregation

Wireless sensor networks have application to many military and civilian scenarios such as target tracking, environmental monitoring, traffic monitoring, seismic detection, etc. In this section, we describe several scenarios where a large amount of sensor data is required to be transmitted across the network and performing data aggregation is beneficial during the process of data collection. In different application scenarios, there exist different data correlation patterns. Given a WSN application, a corresponding data collection strategy (based on an appropriate data aggregation model) should be implemented to benefit from in-network aggregation as much as possible.

1.3.1 Environmental Monitoring

Sensors are often deployed to collect environmental information in order to study and explain the impact of environment on the behavior of living organisms. In an environmental monitoring system, each sensor node is equipped with several different sensing components in order to measure temperature, light, humidity, rainfall, etc. Sometimes cameras with high resolution are also deployed providing visual surveillance on, for instance, rare and endangered species [10, 59]. For example, in [10], there are two types of data transported across the network, numerical data (e.g.

temperature) and high-resolution images. Both of them are periodically generated and have to be gathered at the sink node. The numerical readings are subsequently classified into five categories: “normal”, slightly above and below “normal”, and “extreme outlier” on both sides. Simple environmental readings are only reported once they fall outside the “normal” category. In this exception reporting scheme, the value range of each category is determined by the data accuracy requirements of the application.

In [22], the numerical readings obtained from different sensor nodes over a period of time (e.g. a fixed number of rounds) can be modeled as a function of the time when they were generated and the location where they were generated. Only key coefficients of the function need to be delivered from each sensor to the sink, and the sink can restore all the sensor readings based on the received function coefficients (within a deviation tolerance).

Image data is also collected but less frequently in [10], to provide casual observations while numerical readings are reported as normal. During the image collection, it is possible for a relaying sensor to perform aggregation over the image it received. If the relaying sensor itself obtains a similar image, it can use it to only relay the part of the image it received, because the image obtained by the relaying node itself will be gathered at the sink in its integrity. Upon receiving the partial image and the intact reference image, the sink can restore the aggregated one.

From this example we can see at least three ways of aggregation: based on granularity and “characterization” of data, based on a model described by coefficients, based on the image and its reference counterpart.

1.3.2 Target Tracking

Target tracking is another typical WSN application in both military and civilian domains [76, 44, 65, 33, 79]. The objective is to assess and maintain an accurate trajectory estimation as a potential target traverses the region under surveillance. There may be several targets traveling at the same time [79]. Each of them is usually treated as a point source of an acoustic signal [90]. The signal amplitude attenuates with the distance from the source. When an acoustic source is located near one or multiple sensors, the signal is strong enough to be filtered from the background noise and the sensor data is rich enough to contribute to the tracking task. Each sensor has a limited sensing range; only closely located sensors have the probability to capture the target's acoustic signal. To make sure the system can detect and locate multiple objects, each deployed sensor periodically senses its vicinity and reports to the sink. So the direct implication of cooperation in a WSN in this example is that sensor data between nearby sensors tend to be (spatially) correlated. This correlation can be exploited to reduce data communication, assuming neighboring nodes process jointly the acoustic data.

Besides traditional signal processing sensor networks, new target tracking sensor networks composed of binary proximity sensors have been proposed [76, 44]. Compared with the acoustic-amplitude sensors for target distance estimation and the direction-of-arrival sensors for direction estimation [52], the output of a binary sensor is simple (only one bit denoting if the object of interest is or is not within its sensing coverage). The basic idea in this type of application is to use line segments as an approximation to the real trajectory of the object (within a sufficiently small interval). During the operation of the sensor network, each sensor around the target periodically reports one bit, and a binary "word" is expected to be collected at the sink. For each pair of consecutive binary words collected from the sensors, only a few bits flip. This allows the temporal correlation to be captured by the Hamming distance between successive words, and the same (Hamming distance) representation can be the basis to filter unnecessary wireless transmissions [68, 69].

Two aggregation techniques are applied in this application. The first one is performed by means of collaborative signal processing in which the target location is estimated by some sensor node with the help of other ones. The second aggregation technique is based on error-correcting codes[68], where temporal-correlated observations can be jointly coded, thereby reducing redundant transmissions.

1.3.3 Structural Health Monitoring

The main objective of Structural Health Monitoring (SHM) is to detect the existence and localize the damage in large civil constructions, such as skyscrapers, bridges and stadiums [15, 91, 64, 17]. In order to obtain insight into the physical state of a building, SHM systems usually measure structural response to ambient vibration or forced excitation. Once the damage occurs, a SHM system reports the existence and location of that damage by detecting and analyzing the structural property changes.

The structural response measurements are usually generated at a very high data rate [91]. At each sensor location, the structural response is sampled and collected as a time series of sensor readings for further structural analysis. In [91], a two-fold data aggregation technology is incorporated during the data collection to reduce wireless traffic. The first scheme is event detection, which suppresses energy costly wireless data traffic when there is no interesting event. A threshold is set up to make sure only interesting events will be reported. To further reduce data volume and the large latency over data acquisition, wavelet compression based progressive transmission is applied as the second strategy. The vibration data is written to the sensor's local memory without any compression. But a separate procedure periodically reads data from local storage and compresses vibration samples before sending the data summary to the sink node.

To further exploit internode data correlation, model-based data aggregation derived from structural analysis [64] can be performed without compromising the quality of the acquired data. The a-prior known correlation functions of structural response to events are utilized to facilitate data compression. These correlation functions are determined by the nature and design of the structural system. Once the compressed data is insufficient to support precise damage detection, the sink may issue a request for the original data from each sensor node. The sink's request however for the full data must happen quickly because the full precision data are overwritten by new data as time goes by. Again we see models used to produce aggregate data that are smaller (in volume) than the sensed data.

1.4 Contributions

Data collection is a major functionality for most sensor networks. Constrained by limited energy supply, maximizing lifetime becomes the most interesting research topic in WSNs. To reduce data redundancy transmitted across the network, in-network data aggregation is used. In this thesis, we consider the impact of data aggregation on both tree-based and flow-based data collecting protocols, introduce traffic balancing as a complementary technique to data aggregation for maximizing the system lifetime, and also study the interplay between data aggregation strategy and workload balancing in a WSN. The contributions of the thesis are in three main areas:

1.4.1 Tree-based Data Collection Schemes

A data collection tree is one of the most important and commonly adopted routing topologies in WSNs. To collect sensor data, a tree rooted at the sink has to span all sensor nodes possessing the desired information. Data aggregation benefits much from the tree structure as sensor data are collected while approaching the root. We propose several distributed heuristic algorithms for in-network data aggregation. We also examine the effect of our aggregation approaches in terms of different metrics, such as communication cost, energy cost and processing cost (only for large networks). Then, we design and evaluate a dynamic tree structure adjustment mechanism that proactively alleviates traffic load over some critical nodes whose residual energy is depleted faster due to large data relaying burden. Our performance study indicates that it is possible to effectively construct the aggregation tree based exclusively on local network knowledge and simple sensor devices without location information. Additionally, our proactive tree adjusting strategy is capable of both balancing traffic load and performing data aggregation, therefore significantly extending the system lifetime.

1.4.2 Flow-based Data Collection Schemes

Compared to tree-based schemes, flow-based data collection protocols are more flexible in terms of load balancing. They can split sensor data at any on-path node and transmit data fragments along different paths to the sink. Load balancing across multiple paths results in a corresponding balanced energy consumption and therefore prolongs the system lifetime. We formulate the optimization problem into an IP/LP model where the objective is maximizing the system lifetime. We also take into account the effect of data aggregation by defining a flow loss multiplier which represents the ratio between sensor data volume after and before aggregation. Heuristics are proposed for assigning an efficient data aggregation node for each data source. Simulation results show that our schemes significantly improve performance compared to tree-based schemes both in terms of longer system lifetime and less total data received (indicating more aggressive aggregation) at the sink.

1.4.3 Contour Map Reconstruction

We investigate the impact of performing pairwise data aggregation (in both tree-based and flow-based schemes) on the lifetime of an energy-constrained WSN. Our theoretical analysis indicates that to obtain the maximum lifetime, the effect of aggregation in terms of data volume reduction should be extremely high. Motivated by this result, we explore alternative applications, namely applications based on contour map reconstruction which provides a spatial summary of the entire sensor field regarding the phenomenon of interest. Such applications allow for aggressive aggregation across multiple sources. Simulation results confirm that our aggressive mechanisms remove a large amount of data redundancy and simultaneously ensure required data accuracy. It is shown how our new aggregation schemes can be integrated with both tree-based and flow-based data collection protocols for system lifetime extension, and what is the resulting performance.

Note that, throughout this thesis, the concept of a `flow loss multiplier` is used to model the impact of data aggregation. A discussion about this concept will be presented later on.

1.5 Outline of Dissertation

The rest of this dissertation is organized as follows. Chapter 2 gives a brief overview of the existing work related to maximizing lifetime data collection in WSNs. Since data aggregation is one of our major strategies for extending lifetime, in Chapter 3, we give a brief review of different data aggregation models and their corresponding application scenarios, and introduce a generic model that can express a large number of aggregation scenarios. Chapter 4 presents our tree-based schemes for extending the lifetime of data collecting in WSNs. We deal with the flow-based data gathering solutions in Chapter 5. In this chapter, we describe the IP/LP model with flow loss multiplier (denoting the effect of data aggregation) and our heuristics for choosing the efficient data aggregating sites. In Chapter 6, following the theoretical analysis about the impact of data aggregation over the system lifetime maximization, aggressive aggregation methods for contour data are proposed and evaluated. At last, Chapter 7 concludes this dissertation and discusses the future work.

Chapter 2

Literature Review

One important issue in WSNs is the energy constraint, which implies a need for energy-efficient routing protocols. Most energy-efficient data routing techniques try to optimize their design objectives that express different metrics: total energy consumption [78, 30, 7, 20], system lifetime [12, 34, 42, 95, 84] or total amount of data collected at the sink (during the system lifetime) [71]. To minimize the total energy consumed by all sensor nodes, a typical technique ([13]) is to direct traffic along the shortest path from source to destination. Although it minimizes energy consumption, this approach usually imposes unbalanced traffic load, resulting in rapid energy depletion on nodes that receive most of the traffic.

In some WSN applications, obtaining sensor data from each active sensor node instantly and continuously is impossible or unnecessary. In such cases, sensor data is usually stored locally at each node and extracted when it is required. For such energy-constrained store-and-extract scenarios, regardless of where the data comes from, the total amount of data gathered at the sink is the most important metric [71].

Since gathering data at the sink for further processing is an essential functionality of WSNs, a variety of data collection schemes have been proposed to address the energy consumption imbalance and to prolong the network lifetime. In this chapter, we briefly classify energy-efficient routing into several categories according to the data routing structures they rely upon. A number of tree-based strategies will be first discussed. Then the cluster-based hierarchical routing techniques will be studied. After that, we will give a brief review of flow-based data collection protocols. They are designed to deal with the inherent load imbalance of tree and cluster techniques. We also touch on some other research issues in WSNs that are closely related to our work.

2.1 Tree-based Data Collecting Schemes

It is convenient to deliver sensor data along a tree-based structure from each source to the sink (which is also the root of the tree). To collect all the sensor data, the tree has to span all source nodes. Each on-tree node needs to maintain at least information about its parent node, i.e. the next hop to which it should send the data upstream. Compared with other complicated structures, building a data gathering tree costs less energy for exchanging control messages dealing with path discovery and updating [70]. Additionally, since each node receives all the data from its children, data aggregation can be efficiently performed at any non-leaf node. However, for every source, tree-based routing protocols transmit data along a single path. Some nodes like those near the sink are easily overloaded by relaying large volume of data for their children, which leads to unbalanced traffic load and shortened system lifetime.

2.1.1 Aggregation Trees for Maximum Lifetime Data Gathering

To solve the problem of Maximum Lifetime Data gathering with Aggregation (`MLDA`), an efficient routing approach is proposed in [42] to transmit sensor data from all sensors to the sink. A collection of aggregation trees are built and utilized in order to gather all data at the sink. During this process, many-to-one data aggregation (a number of incoming unit packets are reduced into a single one) is performed enroute at the intermediate nodes between sources and the sink.

Inspired by the flow path augmenting algorithm which iteratively augments data flow along the shortest cost path [12], `MLDA` tries to find a flow augmenting data aggregation tree in each round of data collection until there is not enough energy left at some node for another data collection round. In `MLDA`, a data gathering schedule is defined as a collection of data aggregation trees, each of which is rooted at the sink node and spans all the sensors. Each data aggregation tree specifies within the current round of data collection, how the data is routed from all the sensors to the sink node, i.e. which tree is to be used.

To determine a data gathering schedule with maximum lifetime, an Integer Programming (IP) model is formulated. The original IP can be converted to a LP by replacing the constraints of integer variables with appropriate constraints of real numbers. Then the variable values in the solution to the LP can be rounded into integer variables which do not violate the constraints in the original IP. In this way, a good approximation to the optimal solution can be obtained in polynomial-time. Based on the derived schedule, a sequence of aggregation trees is built.

`MLDA` requires the location information of all sensors and the sink node. Each sensor is also assumed to have the ability to communicate directly with the sink node and any other sensor nodes. Obviously, these two preconditions are expensive.

Another tree-based scheme [83] is provided for data collection and aggregation. In [83], lifetime is defined via both first node failure and network partition. A data gathering tree is built in a distributed fashion in order to save energy consumption on collecting network knowledge and computing paths (from sensors to the sink node). First, each sensor node builds a local minimum spanning tree (MST) over itself and its one-hop neighbors. Based on these local MSTs of individual sensors, a graph spanning all nodes is generated at the sink node. Then, a minimum energy consumption tree is built for data collection and aggregation. To deal with node failure during the process of data collection, a tree structure adjusting approach is also proposed in [83] to maintain the paths for sensor data delivery.

Due to its distributed nature and tree structure adjustment, the proposed algorithm is scalable and robust. Simulation results also verify that its lifetime is extended and close to the traditional MST but with a distributed tree constructing process. However, data aggregation is not explicitly studied in [83].

2.1.2 Spanning Tree Over Connected Correlation-Dominating Set

In a sensor network with a high degree of data correlation, an approach exploiting data correlation during data collection is proposed in [32]. A Connected Correlation-

Dominating Set is a subset of sensor nodes whose sensor readings are sufficient to reconstruct data for the entire network, in addition, the selected subset of sensors has to be connected. The representative subset of sensors is chosen based on data correlation features discovered from a given sensor network scenario and the data accuracy required by the application.

According to the communication graph for a given WSN, several energy-efficient algorithms are designed to select a small set of sensors to form the Connected Correlation-Dominating Set. To evaluate the performance of the distributed schemes, an exponential centralized approximation algorithm is also proposed. It has been shown in [32] that this approximation algorithm returns a solution with $O(\log n)$ of the optimal size, here n denotes the total number of sensor nodes. Based on this approximation algorithm, the authors further provide a set of centralized heuristics which can return other near-optimal solutions. Simulation results show that the designed schemes generate small enough Connected Correlation-Dominating Sets and effectively extend the network lifetime.

Notice that the approach proposed in [32] only works for those cases where strong data correlation exists. Otherwise, the number of representative sensors will be close to the total number of deployed sensors. In addition, the application-specific data correlation pattern should be discovered before selecting the Connected Correlation-Dominating Set in order to guarantee that the data restoration of the whole sensor network will be within the required error bound.

2.1.3 Sensor Surveillance Tree

In the scheme discussed above, data redundancy among similar sensor observations obtained from a subset of sensor nodes is removed during data collection. Another method for reducing energy consumption is to schedule deployed sensors in order to maintain only a subset active for sensing and reporting. This option is examined in [51]. The authors consider a target-watching scenario and present a maximal lifetime scheduling strategy for sensor surveillance systems. The solution consists of three steps. First, assuming the positions of targets and sensors are static and known a-priori, the maximal lifetime is computed via a LP model. Meanwhile, a workload matrix whose element x_{ij} denotes how much time sensor i should monitor target j, is also derived to express the total length of time that a sensor should watch a target. Second, a sequence of schedule matrices are derived from a decomposition of the workload matrix. Finally, a corresponding tree rooted at the sink node is constructed according to each schedule matrix which specifies the active sensors monitoring targets for a period of time and the routes to transmit sensor data to the sink.

The objective is that as long as the system operation continues, any object of interest will be monitored by at least one active sensor and all the data should be sent back to the sink. Due to the large computational complexity, the scheme presented in [51] is not easily scalable for sensor networks with a large number of sensors and targets. Spatial data correlation is exploited by introducing sleep/active modes to ensure each target is covered by as few sensor nodes as possible. This mechanism may not be suitable for other applications where high monitoring granularity is required.

2.1.4 Multiple Trees for Real-time Data Collection

In [62], multiple trees are built to gather sensor data at a number of sinks which are the roots of these data collection trees. During this process, similar to [42] (discussed in Section 2.1.1), many-to-one data aggregation is performed at all the on-tree nodes to reduce total traffic volume.

An Integer Linear Program (ILP) is provided to formulate the optimization problem of finding data collection trees which minimize total energy consumption on collecting data over these trees with limited delay on data delivery. In this model, each link connecting two sensors which are within each other's transmission range, is denoted by a binary variable. The link is on a data collection tree if its corresponding variable is 1; otherwise its binary variable is 0. Then, the total energy consumption can be denoted as the summation of all the products of these binary variables and the amount of energy consumption on transmitting and receiving unit data packet over these links.

Due to the computational complexity, a distributed protocol is provided to minimize total energy consumption depending only on local information. In order to fulfil the required delay bound, for each pair of source and sink node, a path is constructed with an intelligent compromise between real-time data delivery latency and the energy consumption along this path. In [62], it is assumed that each sensor only reports to a single sink node. Since multiple sink nodes are used for data collection, the strategy of sink node assignment is also introduced to balance workload on each sink node.

The assumption of many-to-one data aggregation at each link simplifies the ILP model. In this model, only one unit packet is routed over each on-tree link within a data collection round. It can not be applied for some applications where fractional data may be generated after aggregation.

2.2 Cluster-based Data Collection Schemes

Cluster-based routing is another intensively studied field in WSNs. The basic idea is to organize the deployed sensor nodes into a hierarchical structure where data collection is performed in different levels, for examples, sensor data are first gathered at each cluster heads, then the data are routed to the sink via multi-hop relaying over cluster heads. Specifically, the sensors are usually divided into several clusters according to certain criteria. Each cluster is composed of a cluster head and many cluster members. Within a cluster, the head is responsible for collecting sensor data from its members and reporting to the sink after performing local data aggregation. The hierarchical structure makes cluster-based protocols scalable for large and dense WSNs. However, most cluster-based schemes assume either long haul transmission which ensures any sensor can reach the sink in one hop [34], or ideal data aggregation performed at the cluster head [49, 42]; otherwise, heavy traffic load will still be accumulated as sensor data is delivered upstream to the sink.

2.2.1 Low-Energy Adaptive Clustering Hierarchy

To introduce scalability and robustness in WSNs and tackle the problem of workload imbalance, a cluster-based routing protocol `LEACH` (Low-Energy Adaptive

Clustering Hierarchy) [34] is proposed to collect sensed data from an environment monitoring sensor network. In LEACH, deployed sensors are organized into a number of clusters. The optimal number of clusters is determined by the characteristics of the given sensor network, such as network topology and relative energy consumption on data processing and communication. Each cluster head acts as a gateway node between its cluster members and the sink node. Cluster heads forward data to the sink in one hop. Within each round of sensor reporting, many-to-one data aggregation is performed at each cluster head. Once the network starts operation, all the sensor nodes first randomly elect themselves as the cluster heads with a predefined possibility. Some nodes become cluster heads. The other sensors nearby select an appropriate cluster head to report their data. After a small and fixed number of data collection rounds, all the sensors randomly elect themselves cluster heads again in order to share the heavy energy consumption on the cluster heads.

Due to the randomized "rotation" of cluster head election and the local data aggregation at cluster heads, the total energy consumption is reduced and the system lifetime is extended. In addition, compared to other data routing protocols, LEACH reduces the latency of data collection because it avoids multi-hop data transmission between a source node and the sink. However, LEACH assumes each cluster head can reach the sink node in one hop, which makes it not applicable for a large monitoring area.

2.2.2 Power-Efficient Gathering in Sensor Information Systems

As an improved version of LEACH, a chain-based routing scheme for data collection over homogeneous sensors, PEGASIS (Power-Efficient Gathering in Sensor Information Systems), is proposed in [49]. The basic idea is that in order to prolong the system lifetime, each deployed sensor node only communicates with its closest neighbor and this neighbor node sends all the received data and the data of itself to the next node, this transmission continues until all data of interest are collected. In this way, all the sensor nodes can be constructed into a chain. Upon receiving incoming data packets, each on-chain node performs many-to-one aggregation. The chain construction starts from the node which is the furthest from the sink. A greedy approach is adopted to build a simple chain. Each time, only the neighbor which is closest to the current node will be selected and join the chain. Once added, the on-chain node will never be changed.

Unlike `LEACH`, `PEGASIS` avoids wasting energy on cluster formation and uses only one node within each data collection round to transmit aggregated data to the sink node instead of using multiple cluster heads. During this process, each sensor node takes turns to become leader (all the data of the chain are reported to it) and transmits all aggregated data to the sink. In this way, energy consumption is more evenly distributed than `LEACH` across the entire network. However, similar to `LEACH`, `PEGASIS` assumes each leader node can reach the sink in one hop, which makes it not applicable for a large sensing region. Additionally, `PEGASIS` introduces excessive latency by collecting sensor data along a single chain, which may make it inapplicable to some real time systems.

2.2.3 Hybrid Energy-Efficient Distributed Clustering

Also inspired by LEACH, a Hybrid Energy-Efficient Distributed Clustering protocol (`HEED` [95]) is proposed to prolong system lifetime. Cluster heads are periodically selected by considering each node's residual energy. Sensor nodes with large residual energy will be chosen with high probability because serving as a cluster head will consume much more energy than a cluster member. Once the cluster heads are determined, the remaining nodes have to choose a cluster head to join according to a secondary clustering parameter. To evenly distribute traffic load, a node should join a cluster head with the minimum node degree. Conversely, to create dense clusters for more efficient data aggregation, a node should select the cluster head with the maximum node degree, expecting a large amount of sensor data will be collected and effectively aggregated there. Detailed intra-cluster (within each cluster) and inter-cluster (across clusters) data routing strategies are also presented in `HEED` [95].

2.3 Flow-based Data Collection Schemes

In contrast with tree-based and cluster-based routing techniques, flow-based schemes balance workload by splitting the traffic and transmitting along multiple paths. For each path, partial data is delivered and the amount of data along this path is intelligently decided (usually at the sink node) in order to evenly distribute traffic load. Unfortunately, flow splitting destroys the integrality of sensor data, which makes in-network aggregation very challenging.

We briefly review the existing flow-based approaches whose objective is to compute data flows to maximize the system lifetime, the time at which the first transmission failure occurs due to energy outrage on a sensor node. We will present our solution to this problem later on in Chapter 5.

2.3.1 Flow Augmentation

The authors in [13] formulate the data routing problem as a LP model. To solve the problem, a distributed heuristic algorithm, Flow Augmentation (`FA`) is proposed.

`FA` is a shortest path routing algorithm built on a specific link cost metric which reflects both the energy consumption rate over the current hop of wireless transmission and the residual energy left at the transmitter and receiver. In `FA`, sensor data packets need to be routed from monitoring nodes which work on limited power supply to one, or a set of, sink nodes. Each pair of source and destination nodes represents a single commodity flow. `FA` operates on a sequence of data transmission rounds. Within each round, `FA` finds the cost-based shortest path for each commodity and routes data volume along the chosen path. During this process, each involved link will change the residual energy at the nodes at both endpoints of the transmission. `FA` iterates until one node exhausts its energy and can not transmit sensor data any more. The solution obtained from `FA` has been demonstrated by numerical simulation to be very close to the optimal solution.

`FA` also introduces an important parameter α, which denotes how frequently the link cost information will be updated. There is a tradeoff between maximal system lifetime and the link cost updating frequency. Generally speaking, more frequent updates will lead to a longer lifetime, because the new paths are selected according to the latest link status, i.e. residual energy status. However, the conclusion above is based on ignoring the overhead of refreshing link state. Too frequent updates will result in a large amount of energy consumption, therefore reduces system lifetime.

Notice that, although the lifetime upper bound has been computed by a LP model in [13], `FA` is a heuristic algorithm without any approximation ratio guarantee. Moreover, the order that commodities are routed within each round might have a significant impact on the performance of `FA`, which makes `FA` to perform even worse in some scenarios.

2.3.2 Local-Control Flow Algorithm

A Local-Control Flow Algorithm (`LCFA`) [72] is devised to solve the maximum lifetime routing problem in both static and slowly changing wireless networks including sensor networks. `LCFA` is a multi-commodity flow distributed algorithm which guarantees a near optimal solution. In [72], the problem of routing data between K pairs of source and destination nodes to maximize lifetime is formulated as a maximum concurrent flow problem by viewing each pair of source and destination nodes as a commodity. The data generating rate at each source node denotes the commodity data demand.

Given a wireless network, there are K commodities that need to be considered simultaneously. The idea of `LCFA` is analogous to simulating real-world fluid dissemination. The liquid follows a path from a node with high potential to one with a lower potential. At each node, for each of its outgoing links, there are K queues, each of which is maintained for a corresponding flow commodity. For each commodity, there may be several paths along which each of the data fragment is delivered from the source node to the sink (assuming the sink is the unique destination node for all commodities). A potential function based on the queue "height" is introduced at each intermediate node for each of its outgoing links, in order to compute the appropriate data volume which should be pushed to the other end node of an outgoing link. At the sink node whose potential is always zero for all the incoming link, all the arriving flow fragments are absorbed. By applying `LCFA`, the resulted network lifetime is ensured to be within an asymptotically small error to the optimal solution.

2.3.3 Distributed Subgradient Algorithm

In [56], the subgradient algorithm is used to solve the maximum lifetime routing problem. The dual problem is built on the original LP model by introducing Lagrange multipliers. A subgradient algorithm is proposed to determine the solution to the dual problem, and it is guaranteed to converge to an optimal routing solution.

In [56], each sensor node is assumed to be able to adjust its transmission power level according to the distance between itself and the potential receiver. In addition, a synchronized TDMA (Time Division Multiple Access) MAC (Media Access Control) protocol [2] is assumed to support the protocol. Each sensor node is active when it is required according to the schedule. In this framework, the data communication bandwidth associated with each link is assumed to be much higher than the total flow transmission rate over it, so the feasible solution is free from bandwidth constraints. For the sake of simplicity, energy spent on receiving data at the wireless transmission receiver is ignored.

After modifying the objective function of the primal (original) LP problem, the objective function of the dual problem becomes differentiable. The distributed subgradient algorithm works in a number of iterations. Within each iteration, the variables that are used to evaluate the dual objective function can be updated by each node based on local information. On the other hand, the corresponding solution to the primal problem is also improved and closer to the optimal solution. After enough iterations, an approximate solution to the primal problem with guaranteed error bound can be achieved. Two similar implementations are provided to compute the flows, one is completely distributed and the other is partially distributed in which a centralized computation needs to be carried out by the sink. Based on the simulation results, there is a trade-off between the amount of control information transmitted within each iteration and the total number of iterations before convergence. The fully distributed implementation shows a slower rate of convergence than the partially distributed one. However, the partially distributed implementation requires communication between all the nodes and the sink in each iteration.

2.4 Other Related Work

Besides the data collection schemes discussed above, there are many other routing approaches whose objective is also to spread traffic load and average energy consumption across the network. In this section, we discuss several load balancing approaches that differ from flow-based data collection schemes.

2.4.1 Special Data Collection Structures

Data aggregation usually happens when similar data converge at the same intermediate node at approximately the same time (spatial and temporal convergence) while being routed to the sink. Structured approaches may obtain optimal aggregation by specifying a certain pattern in which data packets are transmitted from each source to the sink. In contrast, opportunistic aggregation requires no fixed pattern but results in non-deterministic and limited aggregation performance.

To obtain efficient aggregation without the overhead of constructing and maintaining the data gathering structure, a structure-free approach which is mainly composed of two approaches, Data-Aware Anycast and Randomized Waiting, is proposed in [24]. Data-Aware Anycast chooses the next hop for each transmission out of a set of candidates. Without an explicit data aggregation structure, it can converge data packets to a few aggregation points where data redundancy will be removed. In addition, higher priorities are assigned to nodes closer to the sink in order to avoid forwarding packets away from the sink. To further improve aggregation performance, Randomized Waiting is introduced to address temporal convergence problems. Any nodes possessing data packets delay sending them out by an interval which is chosen based on a pre-determined threshold. In this way, the opportunity to perform aggregation over more correlated data is increased without inducing high latency to the sensor data collection.

In [60], another routing structure is developed which combines the advantages of tree-based and multi-path based approaches by applying them simultaneously in different regions of the WSN in response to changes in network condition. For a region where the communication loss is low, a tree-based structure is utilized to easily perform data aggregation and produce accurate TinyDB [57] query answers. A multi-path data collection topology is used in an area with a large number of communication failures. Instead of transmitting a partial result to its single parent in a tree-based topology, the multi-path scheme allows each node to broadcast its partial result to multiple neighbors to improve robustness in terms of message delivery. In this way, more than one copy of the result is sent to the sink along different paths. However, unlike flow-based data collection mechanisms whose multiple paths (for each data flow fragment) are strictly chosen for workload balance, the paths selected by the multi-path scheme in [60] are based on their wireless message loss rate.

2.4.2 Multi-Sink Data Collection Schemes

The recent research work [67, 66] show that in a multi-hop single-sink WSN, sensor nodes close to the sink spend much more energy on relaying accumulated data traffic. This unbalanced energy depletion phenomenon becomes obvious when each sensor has a fixed transmission range or an aggressive data aggregation cannot be performed due to high data accuracy requirement. Therefore, sensor nodes close to the sink tend to exhaust their energy quickly, leaving the area near the sink uncovered causing network functionality to be terminated.

To remedy this problem, multi-sink solutions have triggered a great deal of research interest [12, 97, 5, 86]. They can substantially extend lifetime since the average number of hops between any sensor node to the sink is greatly reduced, which subsequently saves total energy consumption on data collection. The lifetime upper bound on multi-sink WSNs is derived in [5]. Accordingly, the optimal locations of the deployed sinks that maximize the system lifetime are also provided. After careful analysis, simulation results validate the lifetime bound and the optimal choice of the sink locations.

In [86], a disjoint multi-path routing technique is proposed to balance workload as well as to improve the robustness of data collection. For each data source, there are two sink nodes (primary and back up), either one of which can be used to collect the data. Two colored trees, red and blue, rooted at two different sink nodes and assuring two node-disjoint paths (from the source to either destined sink node) are formed in order to make data collection resilient to any single sink failure. The colored-trees construction problem is formulated as an IP problem. The authors also devise a distributed path augmentation algorithm to build colored trees based only on the local network knowledge of each node's one-hop neighbors.

Multiple mobile sink nodes are deployed in [6] to mitigate energy consumption imbalance as well as the network capacity bottleneck which usually occurs around the unique and fixed sink node. The authors propose a two-stage data collecting strategy. First, the monitoring area is divided into several polygons with approximately the same size, and the sensor data from each polygon is aggregated at one representative node. Then, each sink collects aggregated data from each polygon when it passes by. Multi-path routing is applied on the second phase to spread traffic load. An analytical framework is set up to determine expected energy depletion. This helps the multi-path routing scheme avoid transmitting data over nodes with poor residual energy. The fundamental interrelationship between the degree of aggregation and aggregated traffic spreading has also been examined in [6]. However, there are no details in [6] addressing how to choose paths for delivering aggregated data.

Multi-sink WSNs can mitigate the uneven energy consumption among sensor nodes. In this thesis, we focus on maximizing system lifetime of the single-sink sensor networks. Since a multi-sink WSN is usually divided into several single-sink sub-networks, our solutions for single-sink networks also help in designing multi-sink sensor networks.

2.4.3 Summary

Three data routing structures are reviewed in this chapter. Tree-based structures are usually easy to build and it is convenient to perform aggregation on on-tree nodes where sensor data from different sources converge. The unbalanced traffic load may shorten the system lifetime. Cluster-based schemes organize the sensors into a hierarchical data collecting structure. Data aggregation can be applied at the cluster heads. Sensor nodes take turns to act as cluster heads, which distribute the heavy relaying duty of being cluster heads. However, cluster heads are assumed to communicate with the sink in one hop. It is not guaranteed in a WSN deployed over a large area. Data flows can be split and routed along different paths in flow-based strategies. They facilitates the traffic load balance across the entire network. But flow splitting destroys the integrality of sensor data, therefore this makes data aggregation challenging. Special structures and gathering data at multiple sinks are also discussed in this chapter.

Chapter 3

In-network Data Aggregation

Data aggregation in WSNs is usually application-specific. Different applications have different data correlation patterns, and the appropriate data aggregation technique should be applied in order to take full advantage of data correlation and improve overall system efficiency. All these diverse in-network aggregation strategies have to be built on data collection schemes. In this chapter, we give an in-depth study of existing data aggregation models according to their different requirements and their impact on the behavior of data collection.

Due to a possibly high density in sensor deployment and the continuous observation of physical phenomena of interest, there are two types of data correlation that can be exploited for data aggregation: temporal and spatial [89]. Before providing a survey of typical application-specific in-network data aggregation schemes in WSNs, we discuss the spatio-temporal data correlation shared by a variety of WSN applications. Subsequently, we introduce the *flow loss multiplier* concept to denote the impact of data aggregation on the data collection abstracted away from characteristics of any specific applications.

3.1 Spatio-temporal Data Correlation

In WSNs that have temporal data correlation, successive sensor data obtained at a source node changes little within a short period. The future sensor data can be predicted based on similar sensor observations achieved so far. For scalar sensor readings, such as temperature, humidity and precipitation, some feature extraction techniques [50] can be applied on the time series of sensor measurements in order to transmit a few coefficients based on which the raw data can be reconstructed with a high enough confidence instead of explicitly reporting every sensor reading to the sink. For more complicated data, such as video, and image sequences in general, the temporal correlation can also be exploited locally at each sensor node to achieve data compression. For instance, the source node transmits updates only if it does not deviate from the prediction too much.

Spatial data correlation is usually the result of dense sensor distribution and physical phenomena that affect a large monitored area [41]. For WSNs with plenty of spatial correlation, within the same collection round, sensor readings from one node could be partially reduced or even totally replaced using readings of another sensor located within the same related area. Of course, the final data restoration at the sink must satisfy the accuracy requirements.

A particular mode of use of WSNs is periodic data collection which gathers sensor data from each sensor location to the sink. For each round of sensor data collection, it is desired to reduce data transmission by exploiting spatial data correlation. It has been observed that spatial data aggregation is often caused by adjacent sensor nodes having overlapped sensing coverage [55, 21]. Therefore, for certain applications, it would be possible to model the data aggregation mechanism in an application-independent manner by expressing the aggregation potential as a function of Euclidian distance between two sensor nodes. In that case, the degree of spatial correlation increases with decreasing internode separation. Thus when two sensors coincide in space, their readings will be identical hence we can completely replace the readings of one with those of the other.

Throughout this thesis, we consider the potential of using spatial correlation for aggregation purposes. This is not to say that the temporal correlation cannot be utilized as well, but the application of the techniques contributed by this to exploiting temporal correlations is left for future work.

3.2 Data Aggregation Models

The help us formulate the data aggregation and routing problem in abstract terms, we need a model that is sufficiently accurate yet of manageable complexity. First, we categorize existing approaches to data aggregation by examining the interplay of data aggregation and data collection protocols. Within each category, we present one or more concrete data aggregation techniques. The observations gathered in this part motivate us for the formulation of an abstract indicator of aggregation potential that we will call *flow loss multiplier*.

3.2.1 Coding-based Data Aggregation

A variety of WSN applications make use of data compression of correlated sensor readings. Coding-based aggregation schemes, such as Explicit Entropy Coding (EEC) [20] and Distributed Source Coding (DSC) [68, 69], have recently received attention. They use the entropy (a measure of the uncertainty associated with a random variable) possessed by each data source as the lower bound on the data volume required to transmit from that node, and use joint entropy (entropy contained in a joint system of two or more random variables) to represent the net amount of data transmission after data aggregation. In particular, DSC refers to the compression of multiple correlated data sources without explicit communication among them. It has been shown in [20] that when EEC is applied, the volume of data redundancy identified and removed is determined by the specific path chosen for routing data. Therefore, coding and transmission structure have to be considered jointly, which makes choosing optimal routes for sensor data sources a NP-hard problem. However if DSC is applied, routing distributed coded sensor data along shortest paths is optimal with respect to minimizing the total communication cost ([data size]*[link weight]) on gathering correlated data at the sink. In practice, it is difficult or too energy-expensive to collect knowledge of the global data correlation structure among all sensor nodes across the network, which is essential to performing DSC.

In order to simplify the problem and minimize communication cost for correlated data gathering, a joint entropy formulation as a concave function (a function of a single variable is concave if every line segment joining two points on its graph does not lie above the graph at any point) of the number of sources is used in [30]. Assuming data correlation is independent of the data source location, a randomized tree construction algorithm is devised to ensure the tree is within an approximation factor to the optimal solution for all such concave functions.

A novel DSC scheme is introduced in [40] to allow each sensor to compress its data without exchanging reference data with others. It also provides each sensor with an adaptive compression rate to account for as the data correlation structure of the sensor network changes. Assuming any sensor reading can be quantized into a n-bit binary string, a tree-based codebook is constructed and each sensor reading can be presented by a i-bit ($i < n$) index which denotes its corresponding codeword position on the codebook tree. At the sink, each index is restored by looking up the codebook tree. Therefore, the transmissions are reduced without any loss of data accuracy.

Aiming at minimizing total communication cost over all potential coding/routing schemes, the authors in [53] prove that combining DSC and commodity flow routing is the optimal solution for a single sink sensor network as long as energy consumption is a convex function of link flow rate. A new metric, distance entropy, based on conditional entropy (the remaining entropy of a random variable given that the value of a second random variable is known) is introduced as the lower bound of the total communication cost. For some generic spatial correlation models, the neighboring correlation dominates the total correlation. A practical data collection scheme which only exploits neighboring data correlation is proposed and it is shown to be asymptotically optimal for a large number of real spatial data models.

To ensure each sensor node obtains all the other sensor readings of the network, wireless transmission increases with the increment of the number of sensors. It demonstrates that when increasing the scale of a sensor network, the transport capacity of each sensor node will decrease. To address this challenge, classical source coding based data aggregation combined with an intelligent routing algorithm is provided in [73]. Data aggregation is performed at intermediate relay nodes where multiple sensor measurements converge. During this process, some encoded sensor readings may be decoded and encoded jointly with other sensor readings. It has been shown in [73] that the proposed approach guarantees the total sensor data volume generated across the entire network is small enough to avoid traffic congestion.

For a typical WSN application, each sensor node observes some underlying physical phenomenon, quantizes its local observation, and reports the quantified sensor data to the sink node. The goal is to reconstruct a global picture of the physical phenomenon of interest within the required data accuracy criterion. So far, there are few practical WSN applications with coding-based data aggregation. Binary proximity sensor network for target tracking [76, 44] can be viewed as one example. However, the application of this data aggregation technique is based on the assumption that binary words from correlated sensors have constant Hamming distance (the number of different bits between two binary words), which makes this coding scheme applicable only for a subset of strongly correlated source nodes.

3.2.2 Query-based Data Aggregation

From the perspective of data storage and extraction, WSNs can be treated as a distributed database system, consequently query processing is a natural application for WSNs. A novel data-centric communication paradigm [38] is proposed for diffusing data to participating sensor nodes. A reinforcement-based path selection scheme is utilized for data routing. During this process, in-network data aggregation can also be applied. Both of them create an energy-efficient and robust data dissemination framework in WSNs.

A query processing system, `TinyDB` [57] can operate over a network of `TinyOS` [35] sensor nodes. It provides a SQL-like interface to help the users form queries for the data they are interested in. In certain WSN applications, instead of extracting all raw sensor readings, the user usually first requires a summarized report to check if an event of interest happened. An in-network data aggregation service, Tiny AGgregation (`TAG`) [58, 98] is proposed to address this problem. TAG supports SQL-style queries, which usually contain aggregate operators (min, max, avg, sum, count). After the user issues the query request, it is propagated and delivered to the corresponding sensor nodes. Then the sensor readings are routed back to the user usually along a tree-based routing structure which is rooted at the user and spans all involved sensors. During data collection, in-network aggregation is performed according to the particular aggregate operator. The basic idea is that instead of applying aggregate operators on the collection of all raw sensor data, the same result can be obtained by performing the operations on several different levels of data segments and merging partial query results enroute.

In [42], a data aggregation scheme is devised according to a database aggregate operator in order to eliminate redundant transmissions and also provide a compact but multi-dimensional view of the region being monitored. Query answers are mapped to data packets. By performing aggregation a particular sensor node along the path to the sink can transform many incoming data packets into a single one with the same size and send it to the next hop.

Besides simple query aggregate operators, a general framework for answering complex queries is presented in [31]. It is applied with the intention to substantially reduce unnecessary data transmissions, when periodically gathering a set of environmental attributes. For each sensor node, a large number of measurements are represented as a function of time at which the measurement is taken. Only the arguments of the function are transmitted to reconstruct the original readings with required fidelity. To exploit spatial correlation among closely located sensors, node location information can be introduced as a factor affecting the sensor readings into a regression model from which the coefficients determining the sensor reading function of time are derived.

Another model-driven data acquisition approach is raised in [22] for answering queries in WSNs. A statistical model of real-world processes built on historical data makes acquired sensor readings easy to understand and efficient to be represented. Once set up, the model provides approximate answers with probabilistic confidence. Data communication only happens when the approximate readings are not within acceptable error margins. Upon receiving recent sensor data, the model is updated to generate more accurate estimates in the future.

Sometimes the users want to extract all raw data from all operational sensors, thereby aggregate operators are not allowed. An approximate data collection approach is proposed in [16] to attack this problem. A family of replicated dynamic probabilistic models are proposed for intelligently exploiting both spatial and temporal data correlation under given network conditions. The model is maintained both at the sink and every deployed sensor node. Each sensor node reports its sensor data once the difference between the current sensor reading and the expectation exceeds the required error bound. This approach is adaptive to the gradually changing physical phenomena and assures data accuracy on collected data as well.

3.2.3 Application Specific Data Aggregation

Data aggregation is usually application-specific. In this subsection, we give a brief review of other application-specific data aggregation techniques in recent research work. We identify contour data aggregation as particularly interesting because it allows aggressive aggregation over contour data and it also has a variety of applications [80, 46, 11, 92, 29]. We propose our own contour data aggregation strategies and the supporting data collection framework in Chapter 6.

Contour Data Aggregation

Extracting contour lines instead of gathering raw sensor data at the sink node provides aggregated but adequate visual information about the underlying physical phenomenon of interest and spends less energy because of reduced wireless transmissions. A contour line is usually defined as a two-dimensional curve along which the attribute values are same. A contour map is composed of a set of contour lines where two adjacent lines are usually separated by equal step-values. Contours represent the global picture of a region being monitored as well as developing trends of phenomena in terms of both spatial and temporal dynamics. Contour map reconstruction has a wide variety of applications, such as forest fire prevention [96], disaster or pollution monitoring [15, 17], weather forecasting [48], surveillance [93], etc. Efficient data aggregation schemes can be applied in order to convey a faithful representation of the area of interest, trading off sensor data accuracy with total volume of data traffic and network resource consumption.

A contour line (isoline) aggregation approach is proposed in [81]. Sensor nodes reporting similar readings and geographically adjacent are grouped into isoclusters [80], denoted by areas delimited by contour lines. Within an isocluster, only those sensors close to the isolines report to the sink. Assuming sensor location information is known a-priori, sensor data from isoclusters is usually represented by polygons whose vertices are manipulated as they travel along the data aggregation tree. However, constrained by the particular data gathering structure, similar sensor readings from two closely located sensors may not be aggregated until they converge at the same ancestor node.

Considering sparsely distributed data samples and noisy measurements, a distributed algorithm for extracting approximate contour lines is developed in [46]. According to neighboring sensor readings, each node computes its gradient which points to nodes with higher readings. This eventually leads to the reference points (at which the extremum readings are obtained) of the sensor field. Based on the location of these reference points, sensor nodes are partitioned into several clusters and the corresponding contour lines are determined by similar sensor measurements obtained within the same cluster. Upon the receipt of all necessary information, approximate contour lines are finally constructed at the sink node.

The notion of event contour is introduced in [63] indicating the pattern of the event of interest such as altitude, temperature, velocity, rainfall, etc. Under the impact of the same event, only the nodes possessing more "rich" information report. Sensor nodes only report their current measurements if they have a large enough change since the last one or several data collection rounds. In addition to providing spatial and temporal data aggregation applicable on neighboring sensor nodes, a novel routing information delivering algorithm is also provided for aggregating sensor data that are multiple hops apart. If a sensor node receives a similar data report from another sensor located within a number of hops, its own report will be suppressed. Note that this approach can also be applied with other particular application-based aggregation schemes in different scenarios.

To represent the original contour map of a sensor field by a sequence of topologicallyequivalent polygons, the authors in [29] propose a new scheme which approximates the spatial summary of the given sensor network. Based on the given parameter k (number of reference nodes), all contour lines will be rebuilt at the sink with k reference vertices while the shape and nesting order of the contour lines are preserved. By treating the entire sensing coverage as multiple equal-sized small grids, a polygonal contour extracting technique is proposed in [92] for producing spatial-temporal patterns of the monitored event. To construct a contour map, each sensor node reports the status of the grid it covers along a multi-hop transmission path. As partial maps, the sensor data can be combined as a large polygonal contour region during the transmission process. In this way, the proposed algorithm trades geographical details of the event pattern (quality of the reconstructed contour lines) for significantly decreased wireless traffic.

Other Application Specific Aggregation Schemes

Similar to the statistical model proposed in [22] to represent and interpret raw sensor readings for answering queries, another model-based data aggregation technique [64] particularly for Structural Health Monitoring systems is proposed to dramatically reduce sensor data. The physical nature of the structural system is considered. A higher priority is usually assigned to the sensor measurement of the latest state of the phenomenon. Based on this model, the missing sensor readings can be estimated with a satisfying accuracy. In this way, the negative impact from data loss due to unreliable wireless transmission is mitigated.

Real time tracking is another major application of WSNs. An aggressive four-tier data aggregation architecture [33] is applied to reduce the large amount of data generated by frequent sensing process at a high resolution. Based on the sensor data obtained at each individual sensor and its neighboring nodes, a detection confidence is first generated as a number between 0 and 1 denoting how confident the sensor node is about the existence of the target within is sensing coverage. Then, a single classification vector composed of multiple confidence values from different nodes, can be used to denote the detection confidence of a group of sensors which monitor the same target. After that, sensor nodes detecting the same target join the same logic group. The group leader aggregates all vectors from its members and periodically reports to the sink. Finally, all reports from group leaders are collected and aggregated at the sink to generate a final report. According to the concrete application requirements, the multi-tier aggregation architecture can flexibly adjust the data accuracy of the aggregated sensor data, thus achieving the balance between energy efficiency and information availability and loyalty.

Another application which generates a large amount of sensor data and requires special aggregation schemes is video surveillance. This type of WSN is composed of multiple high resolution cameras. It is used for object detection, identification and tracking [27, 55, 43]. Within every data collection round, the deployed cameras sense their vicinity and generate a multi-view image of the same scene. All the images are gathered at the sink node for storage and further processing. The spatial data correlation among images is determined based on their overlapping sensing range [55]. During the multi-hop transmission of an image from its source to the sink, any intermediate node is capable of performing aggregation by representing it as a partial image based on the reference image obtained by itself. Upon receiving the partial image and the intact reference image of the same object, the sink will restore the image with the required fidelity.

3.2.4 Aggregation As An Abstract Problem

The benefit of data aggregation has been first theoretically confirmed in [45]. The authors propose data-centric routing schemes for data collection in WSNs. The performance of these schemes are compared with traditional end-to-end routing protocols in terms of energy saving and delay. Heuristic data aggregation schemes are also provided. Then, the impact of network density on data aggregation is examined. The simulation results support that data aggregation, even if performed in a greedy fashion, usually achieves significant energy savings in WSNs with high node density.

Realizing aggregation near the source is critical to energy efficiency and lifetime extension (the more data redundancy removed near the source, the less transmission workload on the relaying nodes), the authors of [25] focus on improving spatial and temporal convergence during the process of data collection. A data aggregation protocol, Data Aware Anycast, is devised for applying aggregation near data sources without using an explicit data transmission structure, thereby reducing the overhead of construction and maintenance of the structure. It delivers sensor data to the most correlated sensor node. Without explicit message exchanges in structure-less aggregation, intermediate nodes just forward data packets to nodes that can achieve aggregation. As the aggregated, or partially aggregated, packets approach the sink, data is further aggregated as the packets are forwarded along a tree rooted at the sink.

In a large WSN, bandwidth capacity bottlenecks are another severe constraint which sometimes can not be solved merely by data aggregation. If additional reduction of sensor reporting rate is necessary to operate within bandwidth constraints, it is desired to ensure all sources have equal chance to provide their data. A new aggregate fairness model which works for any routing protocol is proposed in [14]. In the particular scenarios, some sources have more important data to report than others. In this case, a weighted fairness should be introduced. To implement this model, a localized algorithm is also devised in [14]. For a multi-hop data delivery network, each intermediate node can adjust its forwarding rate for each incoming data flow, thereby avoiding unfair packet drops due to upstream congestion (congestion usually occurs near the sink node, where a large volume of data is often accumulated).

Some applications require guaranteed data precision during the process of data collection. Data quality is often measured by the quantitative difference between an approximate value (after lossy aggregation) and the original sensor reading. Larger data quality implies more data is carried which can result in high energy consumption on some nodes. In particular, in a multi-hop WSN, sensor nodes near the sink node have to relay large data volume for others. They are inherently heterogeneous in terms of energy consumption, e.g. depending on their topology.

The trade-off between data accuracy and energy consumption is investigated in [85] with the intent to extend the lifetime of WSNs. In order to save energy consumption on wireless data communication, only a subset of sensors report their readings as long as the approximate data value of the network which is obtained at the sink node, is within the overall precision constraint. In [85], the application specified precision constraint is partitioned and the corresponding quantitative error bounds are allocated to individual sensor nodes in a coordinated fashion for energy consumption balance across the network. The authors consider a typical WSN application with periodic sensor measurement collection. Each sensor node sends out a new sensor reading only when its value violates the allocated error bound. In this way, energy consumption across the entire WSN can be balanced via careful precision assignment.

Adaptive precision allocation in a multi-hop network operates by periodically adjusting the error bounds of sensor nodes. Once the necessary information from all the sensor nodes is collected, the sink node computes the optimal precision for each sensor and diffuses it downstream to sensor nodes for their adjustments. Large system overhead is introduced by periodically performing this precision allocation, which makes this centralized mechanism not immediately applicable to WSNs with severe energy constraints.

For some WSN applications, for instance, video surveillance [21, 55], complex operations are required for data aggregation. The corresponding energy consumption on data aggregation may not be negligible. In [54], Adaptive Fusion Steiner Tree (*AFST*) is designed to optimize on both transmission and aggregating cost. *AFST* is an aggregation-driven routing scheme, which is capable of adjusting its aggregation decision. It constructs a data collection tree which is rooted at the sink node and spans all the involved source nodes and some other nodes in order to minimize the sum of the given metric associated with each edge.

More precisely, when transmitting sensor data from each source to the sink, *AFST* compares the benefit and cost of the current aggregation operation and decides whether to perform the data aggregation or transmit the data to the sink without aggregation.

3.3 Flow Loss Multiplier

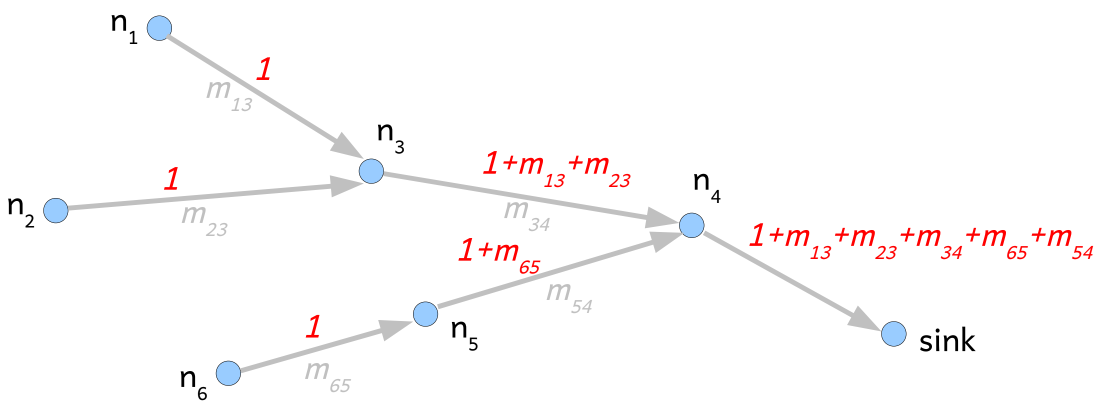


Figure 3.1: An example of flow loss multipliers

Moreover, *flow loss multiplier* can express forms of aggregation not limited to function of Euclidian distance. We will refine the *flow loss multiplier*'s meaning in subsequent chapters.

We have seen in the previous sections the efforts to defining models that capture the *quantitative* impact of data aggregation, and, in particular, of spatial data aggregation. We introduce a means to express the quantitative impact through the *flow loss multiplier*. Specifically, the flow loss multiplier m_{ij} is defined between any two nodes i and j that reside in the WSN. This number is between 0 and 1 and it captures the fraction of data volume (generated at a particular source, say i) that is left *after* aggregation with reference data obtained from its corresponding aggregation node, say j. Figure 3.1 shows an example of how the flow loss multiplier is used to compute total data volume after data aggregation. The numbers above the arrows represent the transported volume of data. Without loss of generality, we assume each node generates a unit of sensor data within each data collection round.

We will use the flow loss multiplier without necessarily describing how it is derived, which we will leave it up to the application. For the sake of example though we could consider a flow loss multiplier which is a function of the Euclidean distance. In such a case, as long as correlated sensor nodes are located with each other's sensing range their flow loss multiplier is a non-zero and less than one value. When they are outside each other's range the flow loss multiplier could be set equal to 1, i.e., no aggregation, and when they coincide the flow loss multiplier is 0, i.e., one of the two sensors can be completely replaced by the data produced by the other (perfect aggregation). This simple model is used, e.g., in Chapter 4, where we introduce a simple circle sensing coverage. Needless to say, the sensing range and the shape of sensing coverage are determined by the features of each type of sensor and the environment being sensed, so this definition is highly idealized. We hasten to add that whereas the flow loss multiplier *as used in the simulations* reported in Chapters 4 and 5 follow this ideal circular model, the schemes in the same sections are applicable regardless of how the flow loss multipliers are derived. Hence, often, throughout this thesis the assumption of flow loss multipliers derived from the circular sensing area overlap is done for the purpose of simulation convenience rather than because of an inherent limitation of the proposed schemes.

While it is true that for many WSN applications [76, 33, 79, 54], the gain of performing data aggregation degrades with the increase of Euclidean distance between a pair of sensors, there are other applications, for instance contour map reconstruction [11, 29] where sensors far apart may still report similar sensor readings. In Chapter 6, we will explicitly discuss contour map reconstruction and the corresponding data aggregation model. The flow aggregation potential can still use abstract models like that of flow loss multiplier but with some suitable adjustment.

For the sake of completeness, we describe a type of aggregation which although more elaborate (and in fact applicable to both spatial and temporal aggregation) can still be captured by flow loss multipliers. This is coding-based aggregation techniques, where the joint entropy is used to denote the net amount of data transmissions after performing aggregation over two correlated sources. The correlation between these two sources guarantees that the joint entropy is not more than the sum of the entropy of each individual data source. Considering data from source i being aggregated at the source j, the difference between joint entropy and entropy of j is the net data volume of i required to be delivered to the sink. In this case, the ratio of this difference to the entropy of i can be used as flow loss multiplier of source node i.

3.4 Summary

In this chapter, we discussed two types of data correlation existing in WSNs, spatial and temporal correlation. Then we presented several concrete data aggregation techniques. We also defined *flow loss multiplier* and showed it can be utilized to demonstrate the aggregation potential during correlated data collection under different WNS application scenarios.

In the remaining chapters, the flow loss multiplier is the abstract quantifier of the impact of aggregation (over a pair of source and aggregation node) without having to know the specifics of the particular application.

We will present our tree-based and flow-based data collecting schemes in the two subsequent chapters. In order to evaluate their performance, we will compare our schemes to other competitive solutions. The criteria for selecting opponent algorithms are two fold. First, the performance (mainly in terms of lifetime extension) of the chosen algorithms is good enough to compare with. Second, data aggregation has to be easily integrated with the chosen data collecting approaches. Therefore, we can also analyze the impact of data aggregation on our data collecting schemes.

In Chapter 4, Flow Augmentation (FA, [13]) is chosen as the opponent algorithm for our tree-based schemes. Different to other candidates ([32, 42, 51, 62, 83]), FA is derived from the linear programming model formulated for maximizing the network lifetime. The simulation results show FA achieves the near-optimal network lifetime. In each round of data collection, sensor data are routed from each sensor to the sink along a particular link cost shortest path. Data aggregation can be applied on any on-path node.

In Chapter 5, Adaptive Fusion Steiner Tree (AFST, [54]) is chosen for our flow-based schemes to compare with. AFST provides a near-optimal solution to the problem of correlated data collection in terms of minimized energy consumption. During the process of data collection, aggregation can be intelligently performed for the maximized benefit from data aggregation.

Chapter 4

Tree-based Data Collection Schemes

In WSNs, sensor data gathering from each deployed sensor to the sink node is the dominant cause of data traffic. Intelligently designed routing protocols for collecting data should be able to significantly improve the system's lifetime. A tree structure is one of the most important and commonly adopted routing topologies in data communication networks. In this scenario, a tree rooted at the sink is built to span all sensors. Such trees are usually constructed so that the cost from each sensor to the sink is minimized, where the cost is determined by the WSN application requirements. Each on-tree node only needs to forward the sensor data obtained from all its children in the subtree plus the data observed by itself. The total sensor data of the network is transmitted upstream along the tree until the sink is reached.

4.1 Static Data Collection Trees with Data Aggregation

We start by considering a hop-based Shortest Path Tree (SPT) as the data collecting structure. To build a hop-based SPT, each node determines its hop-count, h (minimal number of hops to the sink) via Distance Vector routing (see e.g. [9]) and only choose one nearby node with smaller hop-count as its parent. As it turns out, in a dense sensor network, each node has many neighbors that have the same hop-counts. Forming the SPT to collect sensor data requires that we break ties among next-hop neighbors that have the same hop-count. Remarkably, what our results indicate is that a clever way of breaking ties (without much network information to go by) makes a lot of difference.

Another advantage of tree-based data gathering schemes is that data aggregation can be easily performed over collected data at any on-tree node because during the period of system operation, sensor data from a single source will always be delivered to the same parent node. The convergence of sensor data at some node upstream makes data aggregation beneficial for tree-based data collecting strategies. In this chapter, we introduce a simple proximity based model for data aggregation in order to examine the impact of data aggregation on our tree-based collecting schemes. In addition tree-based algorithms are easy to apply and maintain in most WSN application scenarios.

4.1.1 Problem Formulation

First we introduce some notation to help describe the problem in abstract terms. It will be also used in subsequent network models and simulations.

We consider a network of N homogeneous sensor devices deployed in a 2–dimensional space. We use index $i = 0$ to denote the sink, and $i = 1, \ldots, N$ to denote the sensing nodes. The sink does not perform any sensing, it merely collects the data. Let $V = \{0, 1, \ldots, N\}$ denote the node/vertex set, inclusive of the sink. Each sensor is capable of communicating within a radius of R_t, which is the most important parameter in our communication model. $d_{i,j}$ is the Euclidean distance between any two nodes i and j. We use $\mathcal{N}_i$ to denote the set of all neighbor nodes of i within its transmission radius, i.e., $\mathcal{N}_i = \{j \mid d_{i,j} \leq R_t,\ i \neq j,\ j \in V\}$. Assume G is the graph defined as $G(V, E(R_t))$ where $E(R_t) = \{(i,j) \mid d_{i,j} \leq R_t,\ i \neq j,\ i,j \in V\}$, i.e. the edges between nodes are close enough to allow direct communication between them. Without loss of generality we assume that a successful sensor deployment ensures that G is connected.

Energy is a very important factor to data collection protocol design in WSNs. During the process of data collection in WSNs, energy is spent on idle listening, transmitting, receiving and mode switching (e.g. switching from transmitting to receiving). However, transmitting and receiving sensor data is still responsible for the largest weight in the energy budget compared to other energy consumption factors

[8]. A series of experiments have been performed in [26] to obtain knowledge of the energy consumption behavior of actual wireless devices. The amount of energy consumed on transmitting and receiving a data packet is composed of two components. The fixed component is determined by the wireless communicating device's state; and the incremental component is proportional to the size of the data packet.

In WSNs, short sensing and transmission range of the sensor device results in dense sensor deployment, where an energy-efficient medium access control (MAC) layer is necessary to provide a variety of functions that support the wireless transmission across the network [94]. For instance, packet collision has to be avoided as much as possible, if it happens, the packets involved have to be discarded and retransmitted. To focus on the efficiency of data collecting heuristics themselves, we assume the sensor network has an ideal MAC layer. There is no collisions and the sensor wireless transmitting and receiving circuit turns off when there is no need for data communication. During data collection, each node can adjust its radio-power in order to save energy and reduce overhearing, and also ensure enough signal strength detected at the receiver. To study the impact of data aggregation on our schemes, we adopt a simple energy consumption model [34, 66] which also confirms the energy consumption model described in [26]. Specially, the amount of energy spent on transmitting or receiving a unit of data can be denoted below:

$$E = ad^{\alpha} + b \tag{4.1}$$

where d is the Euclidean distance between the transmitter and the receiver, $2 <= \alpha <= 6$, and a and b are both device-related positive constants. The value of path loss exponent, α is determined by the wireless transmission environment.

Each sensor node monitors an area around it. The area sensed by sensor i is denoted by A_i which is approximated by a circular disk of radius R_s. We adopt the convention that the information collected by a sensor is proportional to the area it senses. Hence, in our simple proximity based data aggregation model, the redundancy between the sensed data of two sensors is proportional to the overlap between their corresponding sensed areas. Accordingly, the area sensed by a subset of sensors is the union of their corresponding A_i's. This simple proximity based model for aggregation is connected to the communication model by assuming that $R_t \geq 2R_s$. In this way, two sensors with overlapping sensing areas are capable of communicating directly with each other. Therefore, it is safe to assume that without position knowledge, sensors can communicate to its neighboring nodes and compute the sensing overlapped areas with its neighbors only via the Euclidean distance.

We note that the objective of any data collection algorithm in a sensor network is to forward sensor data along a spanning tree subgraph of $G(N, E(R_t))$ with the root of the tree at vertex 0. For notational convenience, we denote as $c(i)$ the set of children of node i in the data collection tree, and $p(i)$ the parent of node i in the data collection tree (as a convention $p(0) = 0$). Each node i in the data collection tree defines a set, T_i, of all the nodes (including i) that belong to the subtree rooted at i (note that $T_0 = V$, while in the case of leaf nodes $T_i = \{i\}$). Let S_i denote the sensing area covered by all the nodes in subtree T_i, i.e., $S_i = \cup_{j \in T_i} A_j$. Then, the size of S_i can be denoted as $|S_i|$.

The notation we have introduced and will introduce later is summarized in the Table 4.1.

Table 4.1: Summary of Notation

Notation	Definition
V	node/vetex set
$E(R_t)$	edges set determined by communication range R_t
N	number of sensor nodes
A_i	area sensed by node i
R_s	sensing range
R_t	transmission range
$d_{i,j}$	Euclidean distance between node i and j
$\mathcal{N}_i$	neighbor node set of sensor i
$p(i)$	parent of node i (according to current tree)
$c(i)$	offspring set of node i (according to current tree
T_i	subtree rooted at node i
S_i	sensing area covered by T_i
h_i	hop-count of node i
E	initial energy assigned to the sensor node
α	energy critical threshold
m_{ij}	flow loss multiplier between source i and node j

Communication Cost Metric A node in the tree removes redundancy that exists among its children's sensor data as well as redundancy between its own sensor data and those from its children. It forwards up the tree the resulting (non-redundant) information. Assuming the information content is proportional (by a constant factor γ) to the sensed area (after removal of redundancies), then the total communication cost of collecting data along the tree is $\sum_{i=1}^{N} \gamma |S_i|$. Therefore the task of minimizing the communication cost is in essence:

$$\text{minimize} \quad \sum_{i=1}^{N} \gamma |S_i| \tag{4.2}$$

where of course γ is irrelevant to the optimization, and the real impact on finding the optimal tree with the minimum communication cost is in the way the tree is structured, which determines the T_i's and, consequently, the S_i's.

Energy Cost Metric If the transmission power is the same for each transmitted bit and it is the same regardless of the node, then minimization of the energy cost metric for the entire tree is equivalent to the minimization of the communication cost (Equation 4.2), differing only in the constant factor γ. However, a more refined energy cost can be considered whereby, once the data collection tree has been determined, the transmission radius, r_i of each node is adjusted to just enough (but always less than R_t) to reach its parent node in the tree. Thus, $r_i = d_{i,p(i)} \leq R_t$. Assuming the loss exponent capturing the attenuation of transmissions with distance is α, then the total energy cost is $\sum_{i=1}^{N} \gamma' |S_i| d_{i,p(i)}^{\alpha}$, thus the corresponding energy optimization problem is:

$$\text{minimize} \quad \sum_{i=1}^{N} \gamma' |S_i| d_{i,p(i)}^{\alpha} \tag{4.3}$$

for a constant γ' which denotes the proportional relationship between energy consumption and the multiplication of the data volume and the exponential function of $d_{i,p(i)}$. We note that we generally have no expectation that sensors are capable of fine transmission power adjustments to arbitrarily set r_i. Therefore, the results of optimizing this metric should be seen only as an indication of how much better the energy consumption would have been if the hardware complexity could allow for transmission power adjustments.

Processing Cost Metric We also provide a very approximate metric for the processing cost at each node in the data collection tree. We consider the process of redundancy removal to be solely expressed by the cost of identifying cross-correlations between data sets. At its simplest, cross–correlation calculation involves quadratic processing cost (more specifically, cost on the order of the product of the data set cardinalities). Thus, let $c(i)$, the set of children of node i, be expressed as $c(i) = \{i_1, i_2, \ldots, i_m\}$ indicating (in an arbitrary order) the set of the IDs of the m children of node i in the subtree rooted at i ($|c(i)| = m$). We define the auxiliary sets $f_{i,j}$, recursively: $f_{i,0} = A_i$ and $f_{i,j} = f_{i,j-1} \cup S_{i_j}$. The basic idea is that if the total processing cost (for redundancy removal) spent on node i is determined by the joint sensing coverage of node i_j (j-th child of i) and the processing cost on removing data redundancy among i and its the other $j-1$ children ($f_{i,j-1}$). Hence, the total data processing cost can be approximated by:

$$\sum_{i=1}^{N} \sum_{j=1}^{|c(i)|} (|f_{i,j-1}| \, |S_{i_j}|) \tag{4.4}$$

We refrain from considering the optimization problem which minimizes Equation 4.4 because on top of determining each subtree rooted at i (T_i), it will also have to establish an order of the elements of i's children ($c(i)$). According to Equation 4.4, only one particular order of $c(i)$ can result in the minimized total processing cost. While this is conceivably doable, the cost expression supplied will eventually depend on the particular application. Indeed, the processing cost presented here assumes: (a) determining cross-correlation is the dominant process cost, (b) cross-correlation is performed in a pair–wise fashion, while one could opt for simultaneous cross–correlation among more than two sequences at a time, and (c) nodes possess no location information, so they have to indirectly (via cross–correlation) discover redundancies.

4.1.2 Distributed Tree Construction Algorithms

There are several requirements for a data gathering scheme to be designed as a distributed solution. First, WSNs usually work within a hostile environment which induces more frequent node and link failure. It is extremely energy expensive to adjust a data gathering structure in a centralized manner. Second, the large number of deployed sensor nodes require network scalability. In some particular circumstances, collecting global information from a large scale network is not affordable especially for a resource scarce sensor network. Finally, a distributed data collecting scheme is flexible and adaptive to a dynamic sensing task which may be required by some special WSN applications.

In order to evaluate the performance of different tie-breaking strategies in terms of metrics we stated above, we consider five distributed heuristic approximation algorithms for solving the problem of efficiently building the data gathering tree in sensor networks with redundancy removal, i.e., for finding an approximate solution of the optimization in Equation 4.2. The algorithms are a combination of the Distance Vector routing (DV) protocol (see e.g. [9]), but simplified because the only destination of interest is the sink, plus a tie-breaking scheme that characterizes the particular algorithm. As we noted earlier ties are common in a densely deployed network so the effect of tie-breaking rules is expected to be significant. The result of the DV run is that each node i determines the hop-count, h_i, to the sink.

The algorithms rely only on local information and are therefore scalable as the network grows. None of them uses location information. Three of the algorithms (`Random`, `MaxDegree` and `MaxSubset`) are based on a simple-to-determine graph property, namely the cardinality of the set of neighbors $\mathcal{N}_i$. Nodes announce to their immediate neighbors the cardinality of their corresponding neighbor sets. Two of the algorithms (`MaxDist` and `MinDist`) are based on the existence of a monotonic distance-dependent metric (e). A realistic example of such a metric is the received signal strength $e_{i,j}$ (from transmissions of j as received at node i). Specifically if, by convention, transmissions (e.g. certain control traffic transmissions) are at the same transmission power level regardless of the sensor emitting them, a low received signal strength (averaged over a sufficiently long period of time to account for channel imperfections) implies a longer distance from the transmitter to the receiver. Hence, if $d_{i,j_1} \leq d_{i,j_2} \leq \ldots \leq d_{i,j_m}$ (where $j_k \in \mathcal{N}_i$) then $e_{i,j_1} \geq e_{i,j_2} \geq \ldots \geq e_{i,j_m}$.

Specifically, the studied approximation heuristics are:

Minimum Distance to Next Hop (`MinDist`) In this algorithm, a tie is broken by choosing the "nearest" neighbor (in the sense of largest $e_{i,j}$) out of all the neighbors which are of the same hop-count. Notice that within node i's neighbor set $\mathcal{N}_i$, there are some nodes whose hop-counts are 1 less than i. The other neighbors either have equal or larger hop-counts than i. Thus, if $\mathcal{N}_i^\star$ is the set of neighbors of i that have less hop-counts, i.e., $\mathcal{N}_i^\star = \{j \mid h_j = \min_{m\in\mathcal{N}_i} h_m \ , \ j \in \mathcal{N}_i\}$ then, the tie-breaking is:

$$p(i) = \{j \mid e_{i,j} = \min_{k\in\mathcal{N}_i^\star} e_{i,k} \ , \ j \in \mathcal{N}_i^\star\} \tag{4.5}$$

The intuition behind `MinDist` is that gathering sensor data via a SPT will save energy consumption and reduce delay in data collection. In addition, based on our spatial data aggregation model, the most sensing area overlap will occur between each sensor node and its closest neighbor. It is desirable to remove data redundancy as much as possible by performing data aggregation at a close neighbor, as long as this neighbor does not divert us away from the hop-based shortest path to the sink.

Maximum Distance to Next Hop (`MaxDist`) The idea behind `MaxDist` is the opposite of `MinDist`, we attempt to take the longest "stride" away from the current transmitter. However, sensor data is still routed along a SPT, which ensures that it takes the same number of hops to collect sensor data for both `MaxDist` and `MinDist`. Unfortunately, `MaxDist` sacrifices opportunities for better redundancy removal. We present this heuristic anyway as a benchmark to demonstrate the different performance between `MinDist` and `MaxDist`.

Symbolically:

$$p(i) = \{j \mid e_{i,j} = \max_{k\in\mathcal{N}_i^\star} e_{i,k} \ , \ j \in \mathcal{N}_i^\star\} \tag{4.6}$$

Random Next Hop (`Random`) This algorithm breaks ties by uniformly randomly selecting the next hop among all the candidate sensors with minimum hops to the sink. This approach is introduced here for illustrating that a more "educated" selection, in the vein of `MinDist`, does have some impact. Indeed, one would expect `Random` to perform in-between the performance of `MinDist` and `MaxDist` (it turns out that this is the case). Assuming uniform($\mathcal{A}$) is a function denoting the random uniform selection of an element of set $\mathcal{A}$, then the tie-breaking rule is simply:

$$p(i) = \{j \mid j = \text{uniform}(\mathcal{N}_i^\star)\ \} \tag{4.7}$$

Maximum Degree of Next Hop (`MaxDegree`) `MaxDegree`'s tie-breaking selects as the next hop the neighbor that possesses the maximum cardinality neighbor set among all minimum hop-count neighbors. The intuition behind it is that the node with the most neighbors is bound to have several overlaps with its neighbors. Choosing that kind of neighbor as the next hop will greatly improve the opportunity of getting rid of data redundancy among many nodes as soon as possible.

$$p(i) = \{j \mid |\mathcal{N}_j| = \max_{k \in \mathcal{N}_i^\star} |\mathcal{N}_k|\ ,\ \ j \in \mathcal{N}_i^\star\} \tag{4.8}$$

Maximum Subset of Next Hop (`MaxSubset`) `MaxSubset` also looks at the neighbor set of the neighbors of node i to break ties. Consider j, a neighbor node of node i. We observe that some nodes included in the neighborhood of node j, are not going to select j as their parent because their hop-counts are less than node j and our routing is based on minimum hop shortest path. Sometimes, `MaxDegree` does not distinguish these nodes as not being part of the potential "fan–in" towards node j, and misses better choices. `MaxSubset` is based on the "partial degree" of node j, which is the number of nodes within j's transmission radius whose minimum hops to sink are *greater* than the minimum hops for node j itself. Only those nodes have the possibility to choose node j as their parent and should be counted as likely to have their redundancies removed by j. We use $\mathcal{N}_j'$ to denote the set of all neighbor nodes of j within its transmission radius except those nodes whose hop-count is less than or equal to node j, i.e., $\mathcal{N}_j' = \{t \mid h_t > h_j, t \in \mathcal{N}_j\}$. Then,

$$p(i) = \{j \ \mid \ |\mathcal{N}_j'| = \max_{k \in \mathcal{N}_i^\star} |\mathcal{N}_k'|\ ,\ \ j \in \mathcal{N}_i^\star\} \tag{4.9}$$

4.1.3 Simulation Results

To evaluate the performance, we generate random configurations and compare the algorithms based on communication cost, energy cost and processing cost. The communication cost metric was the main design criterion for our heuristics, however, it is of interest to see how the algorithms perform for the other two metrics as well. The configurations are generated as follows. A sink node is placed at the origin of a 2-dimensional coordinate system. Experiments were made with other placements of the sink node but there were no significant differences in relative performance. Sensor nodes are placed uniformly randomly within a square area defined by the origin $(0, 0)$ and (f, f). Ten independent instances (node placements) are generated and the results for communication cost, energy cost and processing cost are averaged for each algorithm. The maximum and minimum values (that is, the range) over the 10 instances are also recorded.

To further explore the performance of our heuristics, we also compare them with another important topology, Minimum Spanning Tree (MST). MST organizes nodes into a tree by connecting them via edges whose total cost (e.g. Euclidean distance) is minimum. The edge cost can be measured in different ways depending on particular application scenario. In our case, short Euclidean distance between individual sensor nodes indicates large overlap of sensing coverage, which also leads to large amount of data redundancy removal. In this way, we intent to capture all the maximized pairwise data aggregation between each sensor node and its parent across the tree. In addition, since energy consumption is proportional to the exponent of Euclidean distance of each edge, MST may spend less energy cost on data collection. There also exists distributed version of MST [28]. In our experiments, we use Euclidean distance as edge weight and modify the regular MST via assigning the sink (root of the tree) as parent to all its one-hop neighbors. Then, the modified MST (`MMST`) is built based on this star topology. The idea comes from our observation that it is efficient for the neighbors of the sink to transmit sensor data directly to the sink node.

In the first set of experiments we consider small configurations where it is computationally feasible to find optimal results. We construct an omniscient exhaustive search algorithm over all spanning trees [74, 75] which determines the optimal spanning tree under either the criterion of minimizing communication cost from Equation 4.2 or that of minimizing energy cost from Equation 4.3. This is done by calculating the union of sensed areas via Voronoi tessellations [3]. However, the exhaustive search is computationally very costly to perform and assumes knowledge (exact location information) that we lack in the presented heuristics. Due to its computation load, we were unable to determine the optimal solution on configurations of more than 13 nodes. The set up for the small configuration experiments have area 25 x 25 (that is, $f = 25$), transmission range $R_t = 10$ and sensing range $R_s = 5$.

In the real world, the range of α in Equation (4.1) changes from 2 to 6 according to the wireless transmission environment [66]. In our experiments, any α between 2 and 6 will be large enough to amplify the weight of Euclidean distance on energy consumption and introduce large energy cost on long-range wireless data communication. Therefore, we set the path loss exponent, α, to 2.

To illustrate the behavior of the optimal algorithms and the heuristics, we examine the data collection trees for a particular instance of the 13 node case in Figure 4.1. The optimal trees for communication and energy are shown in Figures 4.1(a) and 4.1(b) respectively. Note that Node 9 in the minimum communication cost sink tree connects to Node 1; while in the minimum energy cost sink tree, Node 9 conserves energy by connecting to Node 11 which is closer. In Figures 4.1(c) and 4.1(d) the sink trees for the heuristic algorithms `MaxSubset` and `MaxDegree` respectively are shown. Unlike the optimal cases, both heuristic cases connect Node 7 to Node 1 which has fewer hops from the sink than Node 4. Node 8 has a choice of connecting to Node 4 or Node 7 and this tie is broken differently for the two heuristics. In the `MaxDegree` case it connects to Node 7 which has more neighbors; while in the `MaxSubset` case it connects to Node 4 which has more neighbors further from the sink. At last, `MMST` is depicted in 4.1(e). It is much different in shape than the other data collection trees. In order to connect all sensor nodes by short edges, node 9, 10 and 11 have to take a long path (large number of hops) to reach the sink.

We next consider results averaged over the 10 iterations for the small configurations. With the given area and transmission range, fewer than 8 nodes results in many cases where paths to the sink do not exist for some nodes (underlying

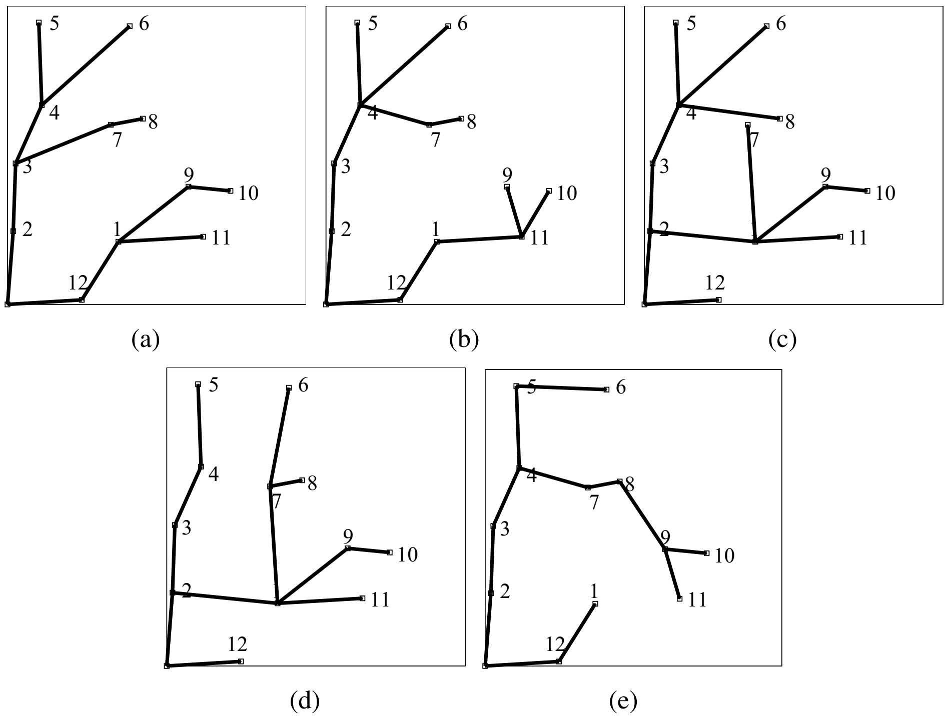
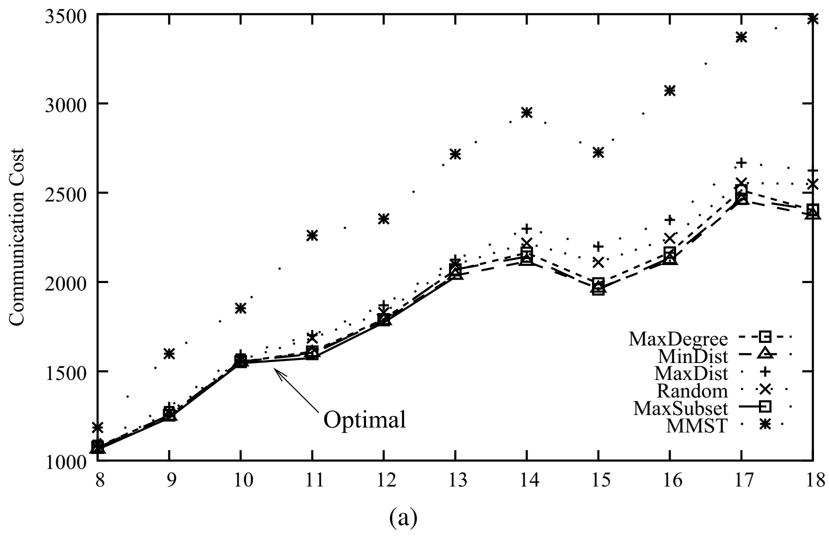


Figure 4.1: Small network example (a) optimal (communication) tree, (b) optimal (energy) tree, (c) `MaxSubset` tree, (d) `MaxDegree` tree, (e) `MMST`.

graph is not connected). Therefore, we vary the number of nodes from 8 to 18 for the heuristic algorithms with the optimal results computed for up to 13 nodes. In Figure 4.2(a), we show the averaged communication costs for the optimal communication cost sink trees, `MMST`s and the heuristic sink trees. As can be seen in this figure, the heuristics are very close to the optimal, while `MMST` has much larger communication cost than others. This indicates that forming shortest hop trees produces close to optimal results for these small cases. Figure 4.2(b) shows that the energy costs of the heuristics are more different from the optimal solution which is recomputed for the new metric. The reason is that reducing energy was not the main design criterion of these algorithms, neither would it have been possible because knowing the $d_{i,j}$ values (possibly via knowledge of their exact location) is not within the assumptions about the sensors hardware complexity for these experiments. However, the energy cost obtained by `MMST` is close to the optimal solution because less energy is consumed on data transmission over its small Euclidean distance edges. In Figure 4.2(b), we also show the energy cost spent by trees which



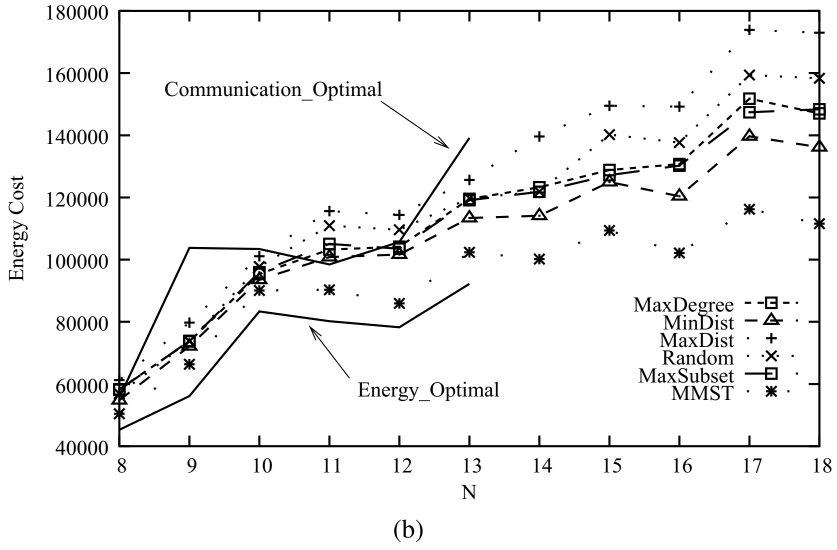


Figure 4.2: Small network (a) communication cost, (b) energy cost.

are optimal in communication cost. The result testifies that optimizing for different objective results in different data collection trees. Moreover, for both cost metrics, the tie-breaking heuristics do not greatly affect the results since in small networks there are few ties to break.

In the next set of experiments we look at larger scenarios with many more nodes. In these cases we have a 100x100 grid ($f = 100$) with nodes of transmission range $R_t = 20$ and sensing range $R_s = 10$. We first consider a particular instance of finding a sink tree for 300 nodes. For these large networks, there are clearer patterns in the sink trees produced by the heuristics. The sink tree for `MinDist` in Figure

4.3(a) shows that this heuristic produces trees with small line segments strongly oriented towards the sink. The fan-in, which indicates opportunity for merging redundant data, is good. The fan-in improves for the heuristics of `MaxSubset` and `MaxDegree` as seen in Figures 4.3(b) and 4.3(c) respectively. These algorithms seek explicitly to improve the fan-in by choosing parents with high potential for a large number of children. `MaxSubset` has the largest fan-in. These designs, however do result in longer segments than the `MinDist` heuristic. `MaxDist`, which can be seen in Figure 4.3(d), has the least-ordered appearance with very long segments because it will choose as a parent the minimum hop neighbor that is furthest away. At last, `MMST` is shown in Figure 4.3(e). It guarantees almost all shortest edges are chosen to construct the data collection tree but ends up with much longer paths (in terms of hops) from the furthest nodes to the sink.

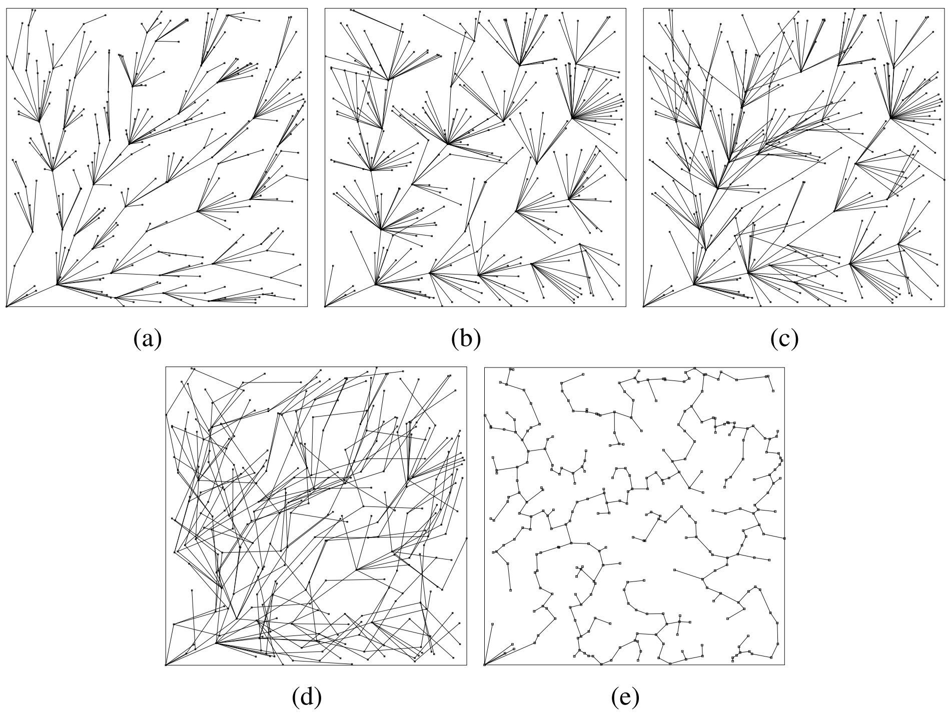
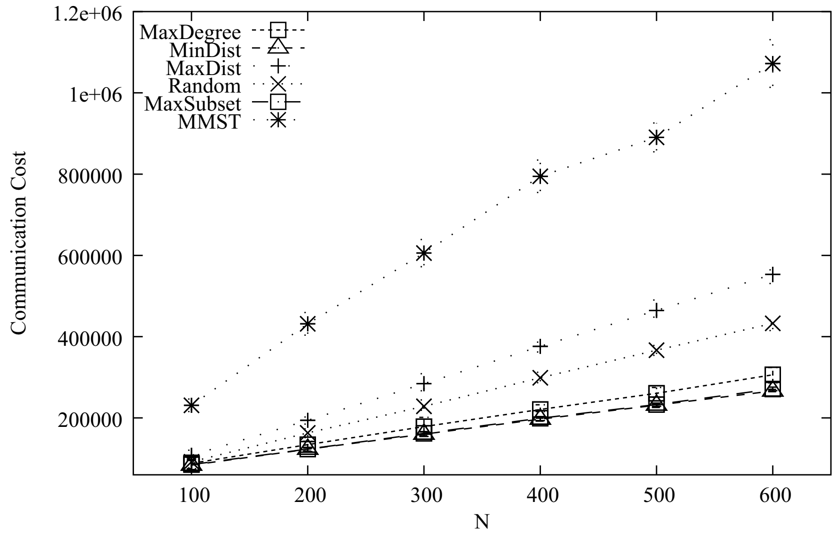


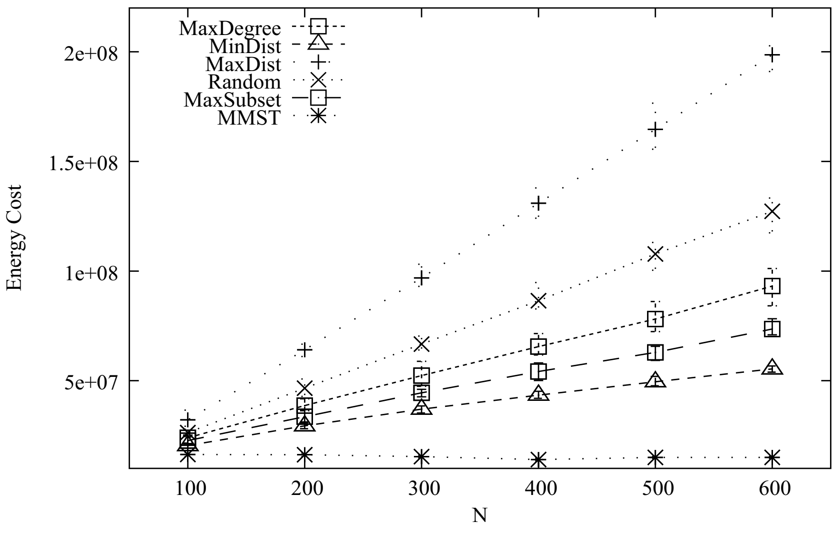
Figure 4.3: Large network example (a) `MinDist` tree, (b) `MaxSubset` tree, (c) `MaxDegree` tree, (d) `MaxDist` tree, (e) `MMST` tree.

We next compare the communication cost and energy cost of the heuristic algorithms for a range of 100 to 600 nodes. We also present the simulation result for data processing cost. In general, for all of the metrics, as the number of nodes increases, the costs increase as well. This can be explained by the fact that leaf nodes are not able to aggregate data until those data reach their parent and as increasing the number of nodes deployed potentially increases the number of leaves. The only exception is MMST depicted in Figure 4.4(b) for energy cost, which will be explained later on. In Figure 4.4(a), the communication cost of the heuristic algorithms is found for different numbers of nodes. The error bars on the graphs represent the range (min and max values) over the 30 iterations. As the number of nodes is increased, the differences in the heuristic tie breakers becomes more significant. This occurs because there are more chances for ties as the network becomes denser. Obviously, all our SPT based heuristics outperform MMST because compared with SPT, MMST has larger hop-count for each sensor node, which leads to unnecessary communication cost spent on routing sensor data along a longer path (in terms of hops) to the sink. MinDist and MaxSubset have the lowest communication cost; while MaxDegree has only slightly higher cost. All of these tie-breaking algorithms seek to increase the amount of overlapping data: MinDist by connecting to close nodes and MaxDegree and MaxSubset by connecting to nodes with more potential to combine data. Their communication costs are significantly lower than random tie-breaking, indicating that intelligent tie-breaking is important. MaxDist has the highest communication cost because it misses the opportunity of data aggregation by connecting to nodes that are further away.

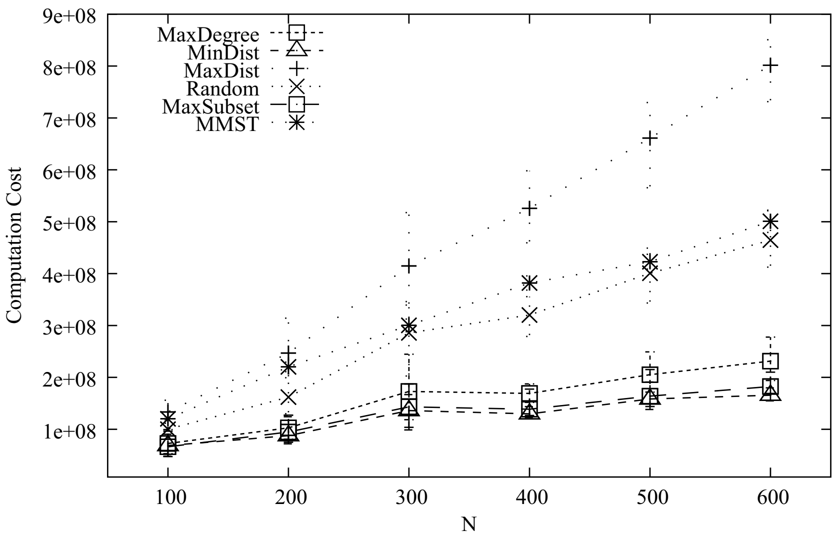
In Figure 4.4(b), the energy cost of the heuristic algorithms and MMST is shown. Since energy cost takes the exponent of edge distance as a major factor, MMST demonstrates great efficiency in terms of energy cost. It possesses the lowest rank among all compared algorithms. In addition, the increase of network scale brings little effect on its performance. The relative performance of the 5 heuristics is similar to the communication cost results except that MinDist now has lower energy cost than MaxSubset. This is reasonable since by choosing the closest neighbor, MinDist will reduce the squared (path loss exponent is set to 2 in this case) distance factor of the energy metric (at least locally). This can also be seen in the examples of Figure 4.3 where the line segments are noticeably longer for MaxDegree and MaxSubset than for MinDist. MaxDist has the longest segments and



(a)



(b)



(c)

Figure 4.4: Large network (a) communication cost, (b) energy cost, (c) processing cost.

therefore highest energy cost because it connects a node to the furthest minimum hop neighbor. In doing so, it defers the removal of redundancy to further up the tree resulting in higher communication and processing costs as well. As we discussed before, a larger path loss exponent may excessively reward some data collection algorithms which transmits over short links, e.g. `MinDist` and `MMST`, and make the difference between them and the other algorithms even larger than before. However, the rank of them may not change.

In Figure 4.4(c), the processing cost of the heuristic algorithms is shown. The relative results are very similar to the communication cost metric results of Figure 4.4(a). The general method already chooses a minimum hop path so that the ability to combine data is the dominating requirement of the tie-breakers for both communication and processing cost metrics. Again, the performance of `MMST` is even worse than `Random` because `MMST` fails to process redundancy removal in an efficient way. Compared with `MaxDegree`, `MaxSubset` and `MinDist`, for each interior node, `MMST` usually ends up with less fan-in which induces longer hops (more processing effort) for the same amount of redundance removal.

Taking into account the removal of data redundancy, we proposed five distributed heuristic algorithms for constructing data collection trees. We also evaluate their performance through detailed simulation based on different metrics. Communication cost reflects the impact of data redundancy removal on data gathering process. According to our simple proximity based data aggregation model, processing cost describes the computational load on each involved node for removing data redundancy. Processing cost may vary with different data correlation scenarios. Since it is an application-driven metric, we will not apply it to our future simulations where a general correlation model is assumed. As we stated before, the extremely limited energy resource is the major challenge in WSNs. Therefore, among all three metrics, energy consumption will become the major concern in the rest of this thesis. As well as minimizing energy consumption among entire networks, to maintain the system functionality as long as possible, balancing energy consumption across the network becomes another issue in WSNs. Based on the observation of previous experiments, tree based data gathering schemes have one intrinsic problem, the accumulated data traffic makes the nodes close to the sink have much more load than leaf nodes. For instance, the largest energy consumption differential ratio is about 7.3 in a 13-node network. This ratio goes up with the increment of the network scale. In the next subsection, we propose tree adjustments to release heavy traffic duty on some sink-close nodes induced by a fixed data gathering tree.

4.2 Energy-Critical Node Aware Spanning Tree with Dynamic Tree Structure Adjustment

We observe that the sensor node's double task of sensing and forwarding different amounts of traffic from other nodes results in energy depletion which is heavily dependent on the data collection structure. An uneven traffic load drains some nodes' residual energy much faster than others and leads to loss of sensing coverage or even worse, loss of network connection. In this case, balancing workload among all deployed sensor nodes has to be considered as well as energy-efficiency.

To address this problem, we propose a distributed spanning tree building scheme by identifying energy-critical nodes (whose residual energy is equal to or below the predefined threshold) and making them leaf nodes, or as a last resort, balancing the traffic load handled by them, so as to release them from heavy duty. During the process of data collection, the tree structure can be judiciously adjusted to balance energy consumption before any node's failure due to total residual energy depletion. This tree structure adjustment can be performed locally with few control message exchanges and as few as possible sensor nodes involved.

4.2.1 Energy-Critical Node Aware Spanning Tree

In this subsection, we present the algorithm, "ENergy Critical node Aware Spanning Tree" (`ENCAST`). The objective is to attach critical nodes as leaf nodes of the data collection tree such that, (a), data from critical sensors can still be collected and, (b), critical nodes are no longer responsible for forwarding upstream data of other nodes. Property (b) needs to be violated if it becomes impossible to reach certain nodes (critical or non–critical) without the assistance of critical nodes. A pattern of energy–critical nodes could result in a partitioned network if we plan to avoid using energy–critical nodes as data relays. When this becomes the case, critical nodes are recruited for data forwarding, thus violating (b). In that case, (b) is replaced by another property, (c), that the load across critical nodes that become interior tree nodes is kept balanced. By load we mean the number of children attached to a node. Clearly this form of load balancing is restricted by the particular topology. Certain nodes may not be able to assist as many children as other nodes, and a "natural" imbalance ensues. Thus, ENCAST attempts only to approximate property (c).

ENCAST operates in two phases. The first phase is a breadth first traversal starting at the sink. If no critical nodes exist, the data collection tree will be the tree derived from this first phase. The breadth first search is a well solved problem, and distributed versions exist with small message complexity, e.g., [4]. When critical nodes exist, any critical node visited in the breadth first search blocks any further traversal from the visited critical node. That is, at the end of the first phase, critical nodes can only be leaf nodes of the resulting tree. Clearly, by blocking the traversal to proceed further than a critical node, certain nodes may not be reachable via a tree path from the sink that involves exclusively non-critical nodes. The tree constructed from the first phase is not a complete spanning tree, and off–tree nodes can be a mix of critical or non–critical nodes. The second phase extends the tree produced by the first phase in order to reach the off-tree nodes.

We note that if there are no critical nodes, the tree resulting from the first phase is a Shortest Path Tree (SPT) in terms of *hop-count*. However, it has been noted in the past [100, 99] that in dense sensor networks, many SPTs exist. The existence of many SPTs is due to the fact that nodes have many neighbor nodes and many of these neighbors have the same minimum-hop distance from the sink. In this case, a secondary (tie-breaking) criterion can be used to select the "best" parent node among all potential minimum-hop parent nodes. We consider two tie-breaking criteria. The first is MinDist (proposed in the previous subsection) according to which the selected parent is the minimum-hop node nearest (in terms of Euclidean distance) to the child node. MinDist requires ranking the neighboring nodes according to their distance based on the received signal strength, and it has been shown (refer to the previous simulation for tree-based heuristics) to result in significant data aggregation potential. The second is the Weighted Randomized Parent Selection (WR) scheme [99] according to which a (minimum-hop) parent is randomly chosen in inverse proportion to the number of nodes that are neighbors of the potential parent. WR exhibits potential for balancing the resulting tree, producing presumably a more even load across sensors.

```
1. $CR := \{v \in T : v \text{ is critical }\}$;
2. **foreach** $i \in CR$ **do**
3.     $critical\_ancestor[i] := i$;
4. $P := CR$
5. **while** $|T| < |V|$ **do begin**
6.     $C := \emptyset$;
7.     **foreach** $i \in P$ **do**
8.         **if** $j \in N_i$ **and** $j \notin T$ **then** $C := C \cup \{j\}$;
9.     **foreach** $j \in C$ **do begin**
10.        $k := \arg\min_{l \in \mathcal{N}_j \wedge l \in P} w[critical\_ancestor[l]]$;
11.        $T := T \cup \{j\}$;
12.        $p[j] := k$;
13.        $critical\_ancestor[j] := critical\_ancestor[k]$;
14.        **while** $k \notin CR$ **do begin**
15.            $w[k] := w[k] + w[j]$;
16.            $k := p[k]$;
17.        **end**;
18.    **end**
19.    $P := C$;
20. **end**
```

Figure 4.5: The second phase of `ENCAST`.

The second phase of `ENCAST` (Figure 4.5) augments the partially constructed tree by allowing critical nodes to become parent nodes of other off-tree sensors but under the approximate enforcement of property (c) above. Let $V = \{0, 1, 2, \ldots, N\}$ be the set of N nodes plus the sink, where index 0 stands for the sink. The network topology can be described as the undirected graph $G(V, E)$. Let T be the set of nodes in the partially constructed spanning tree produced by the first phase. We use CR to depict all critical nodes explored so far. Let $\mathcal{N}_i$ be the set of nodes neighboring node i, that is nodes within transmission radius of node i. Additionally, assume $p[\,]$ is the array holding the index of the node selected as parent in the tree, that is if $p[i] = j$ then j is the parent of i. We use two sets, P, the potential new parent nodes for C ($P \subset T$) and, C the set of nodes accessible as children from the nodes in P that are not already in T ($C \subset V \setminus T$). An array $w[\,]$ (initialized to zeros) holds the current "workload" value for each critical node.

To initialize the second phase of `ENCAST`, we use another array $critical\_ancestor[\,]$

to keep track of the traffic load on original critical ancestor node of the currently visited node. For any on-tree node (rooted at the sink), its critical ancestor is its ancestor node which is in the same branch of the current node and also the sink node's direct child. Because in most network configurations, the traffic load of the nodes close to (specifically one hop away from) the sink determines the system lifetime. We start with those critical ancestor nodes and try to assign them as other off-tree nodes' parents. Line 8 produces the set of nodes that can be reached from the set of potential parents and are not yet part of the spanning tree T. Line 10 demonstrates that this parent assignment continues in a greedy way. After that, workload information is updated (line 14-17). We note that line 19 suggests that the calculation progresses in "waves": once a set of potential parents become parents of a set of child nodes, then the new potential parents are the children explored in the previous step. Before executing the second phase of `ENCAST`, only critical nodes are included in P. After the first iteration, the potential parents set, P, can include critical and non-critical nodes.

With minor modification, the second phase can be reused when the residual energy of some on-tree node with heavy traffic load drops below the threshold. Once some node becomes energy critical due to fast energy depletion, it will force all its children and offspring to transmit sensor data to the sink node through other relaying nodes. After the construction of the first data gathering tree, tree adjustment will be repeated once a energy critical node appears. Since rapid tree adjustment is necessary to keep the current data gathering structure adaptive to dynamic unbalanced energy consumption, it is desired to apply a distributed tree adjustment strategy in order to minimize system overhead and manage tree adjustment as locally as possible. Finally, the second phase can be expressed in a distributed fashion, since all relevant information is local (up to two hops away). Notice that during the execution of the second phase of `ENCAST`, only the nodes involved in the current "wave" have to make wise decision based on updated workload knowledge, which means the latest $w[\,]$ does not need to be reported back to nodes in CR (critical node set). The added complication is the need to employ synchronization in advancing the calculation "wave". We modify an approach from [4] to solve this problem.

4.2.2 Dynamic ENCAST Adjustment

A newly deployed network may initially include no energy-critical nodes. Assume all nodes start with full energy reserves, E, and assume we start with either a MinDist, WR, or arbitrarily calculated SPT tree. Sooner or later node failure will occur. However, ENCAST allows for proactively adjusting the tree as nodes become critical. An invocation of ENCAST operates on one (the current) instant of the network with respect to which nodes are currently considered critical.

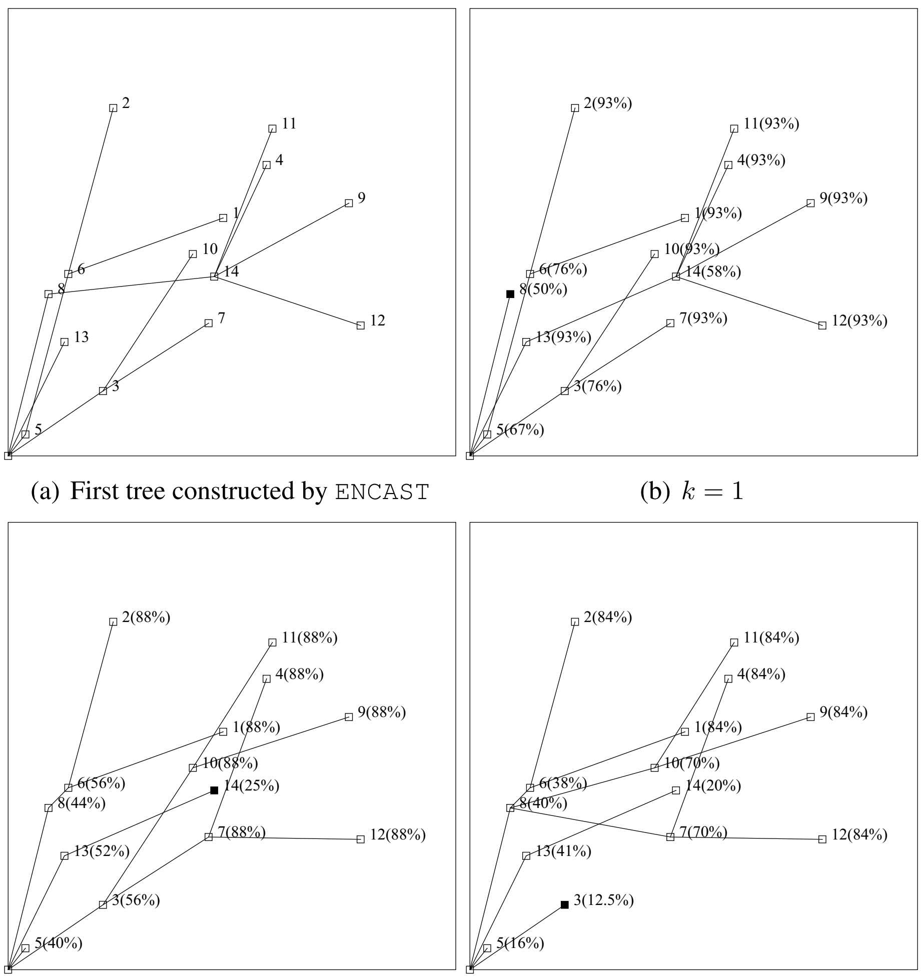
Once a node discovers itself energy critical, it first broadcasts DETACH_REQ message to all its children informing them their parent will not relay their sensor data anymore. Upon receiving DETACH_REQ message, each involved child node checks its neighbor list and looks for an alternative parent node which is not critical and does not have critical ancestor nodes. No matter whether data aggregation is performed or not, if all distributed sensor nodes start with equal initial energy, the sink's direct neighbor nodes usually have the most heavy workload and run out of energy first. To evenly balance workload among those potential energy critical nodes, each on-tree node also maintains its critical ancestor. Each DETACH_REQ message has to contain the sender's ancestor node in order to avoid any child node choosing as parent a node with the same ancestor as itself. The candidate with minimal ancestor workload will be first chosen and an ATTACH_REQ will be sent to it. Once the sender receives a confirmed ATTACH_ACK message from its alternate parent, it will report sensor data to the new parent from the next data collection round. If the critical node's child has no other parent candidates or receives a refused ATTACH_ACK message from all its alternative parent nodes, it will send refused DETACH_ACK back to the critical node. Meanwhile, it also identifies itself as a critical node and sends DETACH_REQ to all its children in order to reduce its original ancestor node's traffic load. This process may continue and terminate at the leaf nodes of the previous data collection tree.

Now an intelligent rule for characterizing nodes as critical and deciding when to adjust the structure of ENCAST is necessary. We trigger ENCAST when the residual energy of any sensor drops at or below αE, where E denotes the initial energy assigned to each sensor and $\alpha \in [0, 1]$. The event causes the first data collection structure adjustment. Likewise, we establish several subsequent thresholds. The first node to cross the particular threshold triggers the adjustment. The series of thresholds is defined as:

$$\{\alpha^k E \mid k = 1, 2, ...\} \tag{4.10}$$

where k denotes the k-th time the data collection tree is going to be rebuilt. Nodes below the corresponding threshold are defined as critical. Therefore, α determines the timing for the adjustment of the data collection tree and is important to the system lifetime. If $\alpha = 0$, the system will never re-adjust the tree created at the beginning. Adjusting the data collection tree based on an exponentially degraded series of thresholds has two benefits. First, changing the tree too frequently is avoided, especially early on in the lifetime of the network. Frequent changes could cause the same energy critical node to be identified in successive re-adjusting phases adding no real value since the algorithm has already attempted to place them as leafs. Second, when the system is approaching the end of its lifetime, the data collection tree will be checked and possibly changed much more frequently than at the beginning of operation, thus potentially spreading the workload better over low energy nodes. However, we will closely inspect the resulting overhead induced by re-adjusting the data collection tree. We will subsequently measure the lifetime extension via tree re-adjusting with consideration of energy consumption on control message exchange.

To demonstrate how dynamic ENCAST adjustment works, we examine an example of a small sensor network which is composed of 15 sensor nodes including the sink. These 14 sensors are randomly placed within a $25 * 25m^2$ area. The sink is located at the left bottom corner. All nodes are assigned the same initial energy except the sink. For the sake of simplicity, we assume all sensor nodes are data resources with the same data generation rate and there is no data aggregation performed for this simple case. Now each node's traffic load is proportional to the number of children included in its subtree. Without loss of generality, we set $\alpha = 0.5$. A sequence of data collection trees obtained by ENCAST are shown as Figures 4.6(a)–4.6(d). We also present the residual energy percentage at each sensor node. Data collection begins with tree 4.6(a). At that time, every sensor node has full energy. After a number of data collection rounds, node 8's residual energy drops to 50% ($\alpha = 0.5, k = 1$, refer to Equation 4.10) and becomes critical due to its heavy transmission workload. Then, it triggers ENCAST adjustment and the data collection tree is transformed to 4.6(b). Here we use a solid square denote the critical node in order to distinguish it from other nodes, and the residual energy ratio is presented for each node. Figure 4.6(c) and 4.6(d) depict the second and third ENCAST adjustment.



(a) First tree constructed by ENCAST

(b) $k = 1$

(c) $k = 2$

(d) $k = 3$

Figure 4.6: 15 node data collection trees derived from ENCAST.

4.2.3 Simulation Results

We use simulation to explore the performance of ENCAST. In the simulation, we randomly distribute sensor nodes within a $100 \times 100(m^2)$ area with the sink placed at corner. The transmission range of each node is $R_t = 20m$. After the distribution, a data collection tree is built to gather the sensor data from each sensor at the sink. We assume the application requires the data collection to be done every 5 minutes ($300s$). For the sake of clarity and without loss of generality, we assume, in each round, each sensor obtains the same amount of sensing data ($1KB$) from its sensing components and reports to the sink.

The performance metric of most interest to us is system lifetime. The sink is a special node that always has enough energy. The sensors are given an initial ("full") energy of $E = 10J$. We assume energy consumption due to data reception and transmission is determined by the wireless communication environment and the packet size [26]. The major parameters set for the simulations are listed in Table 4.2. To focus on the behavior of ENCAST, we still assume ideal MAC.

Table 4.2: Simulation Parameters

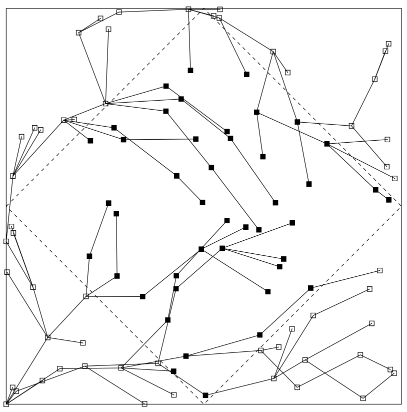
Parameter	Value
Unit of lifetime	second
D (flow data demand)	$1Kb/300second$
E (full initial energy)	$10J$
E' (low initial energy)	$3J$
size of area of interest	$100 * 100(m^2)$
R_s (sensing range)	$10m$
R_t (transmitting range)	20m
e^t	100nJ/bit
e^r	30nJ/bit
Control message size	0.1KB
α	0.5, 0.66 and 0.75
β	0.1

Lifetime for Critical Areas: Static ENCAST

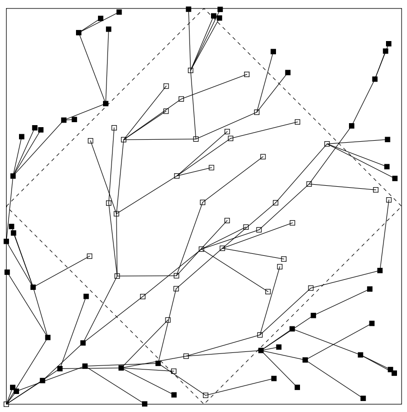
In the first set of experiments we consider the case of designing data collection trees for critical areas of sensor nodes. These areas could arise in a number of scenarios. In some applications, a subset of sensor nodes have much less residual energy because of the non-uniformity in the energy reserves of the deployed sensors, for instance, heterogeneous sensor distribution [23]. In other cases, the same network is used for both periodic sensing and query-based sensing. There may be particular nodes of more interest than others that are queried more often, causing potential overuse of those nodes. Critical nodes may also be the result of imperfectly distributing nodes in space, resulting in uneven coverage and, thus, a proportionately heavier traffic forwarding burden for some of the sensors.

To study the ability of ENCAST to handle critical areas, we evenly divide a square area defined by the origin $(0, 0)$ and $(100, 100)$ into 2 equal-sized areas. Area 1 consists of the central nodes and Area 2 consists of the nodes in the four corner triangles. The dotted outline of these areas is shown in Figures 4.7(a)–4.7(d). With a uniformly random distribution of nodes these two areas will have the same expected number of sensor nodes. We designate sensor nodes as non–critical or critical where non–critical nodes are given full energy ($E = 10J$) and critical nodes are given low energy ($E' = 3J$). Two cases are considered. In the first case, all sensor nodes in the center are assigned low energy; while corner nodes are given full energy. In the second case, all sensor nodes in the corner are designated as critical; while central nodes are non–critical. In this way, we can evaluate the performance of ENCAST under different scenarios in which critical and non-critical nodes are all deployed over the entire area of interest. We apply the non-iterated (static) version of the ENCAST algorithm (namely, no tree adjustments) to form the data collection tree and then the sensors periodically collect data until one of the sensors runs out of energy. We also compare these results with a collection tree based on MinDist and WR which makes no distinction between critical and non–critical sensor nodes.

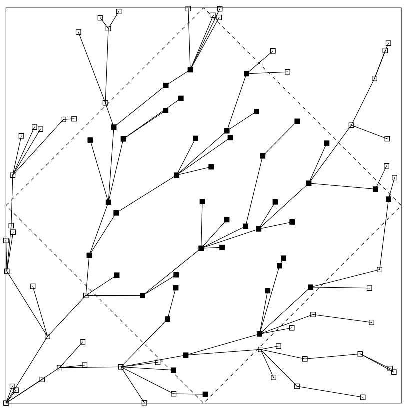
As we introduced before, WR [99] is a Weighted-Randomized parent selection algorithm which tries to build a workload balanced data gathering tree. During the process of the tree construction, for the current node, each potential parent is assigned a weight according to that parent's degree in the communication graph. The probability for choosing a parent node is inversely proportional to its degree. In this way, WR builds a balanced data collection tree. MinDist is one type of hop-based SPT, which has been verified the most energy-efficient structure among all the SPT heuristics presented in the previous subsection.



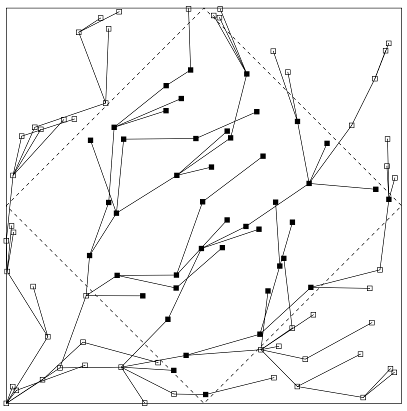
(a) ENCAST & interior critical nodes.



(b) ENCAST & corner critical nodes.



(c) MinDist (no critical nodes).



(d) WR (no critical nodes).

Figure 4.7: 100 node data collection trees.

In Figure 4.7(a) we show a particular example of a 100 sensor network when ENCAST is utilized to create a data collection tree for critical central nodes (the small solid squares within the central area stand for the critical nodes). Note that the critical nodes mostly become leaves of the tree and the paths to the sink for these nodes are sometimes much longer than the shortest path. In this way, less traffic relaying burden will be imposed on the energy critical nodes. Figure 4.7(b) shows a collection tree created by ENCAST for the case of critical corner nodes. In this case, the nodes at the corners are energy critical and some nodes close to the sink (which is assumed located at the origin) unavoidably have subtrees but the algorithm tries to spread the load more evenly over these nodes. Trees created by algorithm MinDist and WR are shown in Figure 4.7(c) and 4.7(d). From the trees shown in Figure 4.7(a), we observe that ENCAST is capable of adjusting the sensor data collection tree according to the residual energy at each sensor node.

In Figure 4.8, the system lifetime is given for the two scenarios with increasing size of network. The error bars represent 95% confidence intervals over 30 independent runs. In both scenarios, the results show that system lifetime is improved when the tree is designed with attention to critical sensors. That is, regardless of the size of networks, the lifetime is longer for `ENCAST` than `MinDist` and `WR`. In addition, since `WR` considers workload balance during the process of tree construction, it outperforms `MinDist` in both cases. When `ENCAST` is employed with critical sensor nodes in the center, the density of the network has a large impact. As the network becomes denser, the central sensor nodes connect through longer paths and the nodes close to the sink corner (left-down corner of the area) become unbalanced with large subtrees. This results in some sink corner nodes becoming critical and eventually dying. In contrast, if critical sensor nodes are in the corners, `ENCAST` spreads the load more evenly over the sink corner nodes resulting in less sensitivity to density. Higher density means larger subtrees but also more sink corner nodes to spread the load over. For denser networks, despite the fact that the critical corner case starts with low energy in the crucial sink corner nodes, the balancing of their subtrees by `ENCAST` provides longer lifetime than if the same number of critical nodes were centrally located.

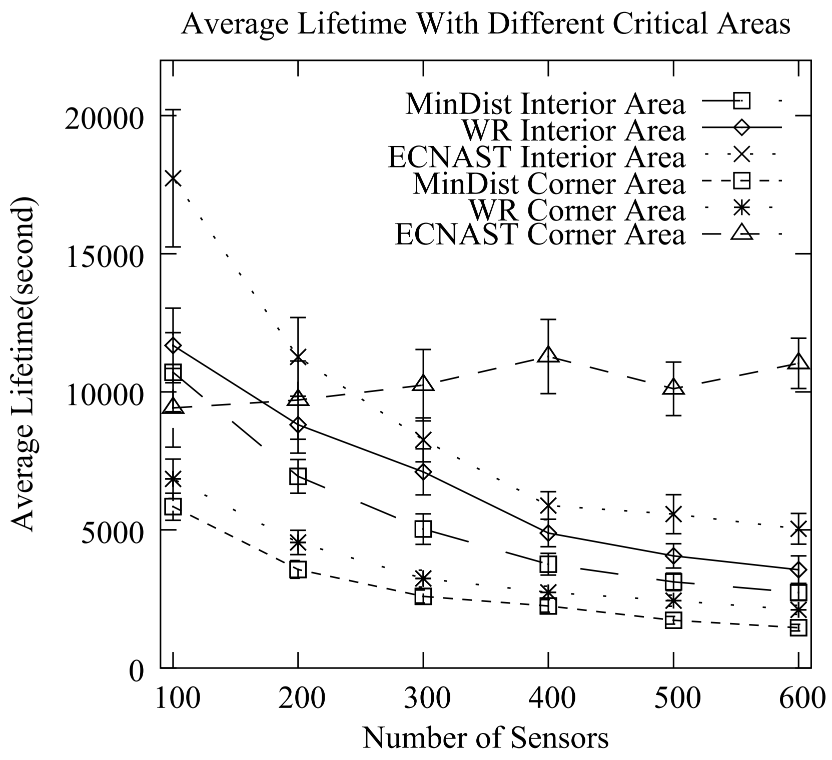


Figure 4.8: Lifetime & different critical areas.

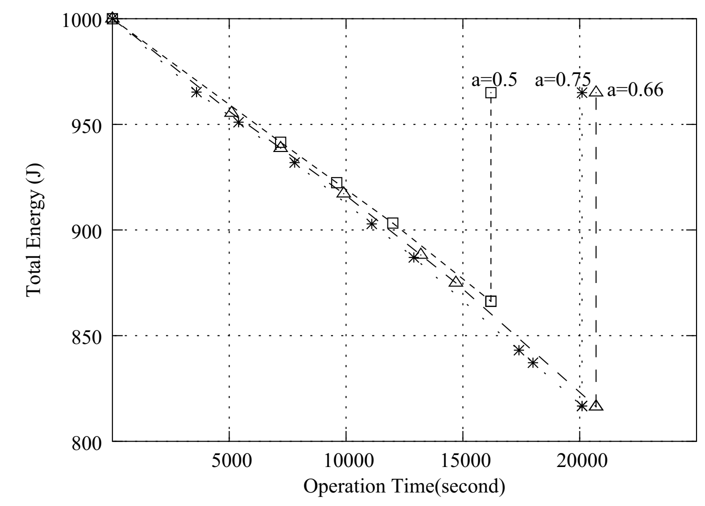
Dynamic ENCAST Adjustment

In the second set of experiments, we explore the effect of different threshold values in the iterative application of ENCAST. In this scenario, all sensor nodes start with full energy. As we described before, a hop-based Shortest Path Tree (SPT) with MinDist tie breaking is constructed by ENCAST in a distributed fashion. Sensor nodes then periodically collect and deliver data until the energy of some nodes become critical according to the threshold or until one of the sensor nodes runs out of energy. Once some node's residual energy drops below the corresponding threshold which is determined by $\alpha \in [0, 1)$, ENCAST will trigger data collection tree adjustment in order to release the workload of the current critical node. Since frequent tree adjustment may waste valuable energy on control message transmission, we introduce another parameter β to stop dynamic ENCAST adjustment once some node's residual energy is too low. We compare the performance of the algorithm with different threshold (α) values. We set $\beta = 0.1$, which means ENCAST will assure the current critical node's residual energy is above 10% of the initial energy before triggering itself.

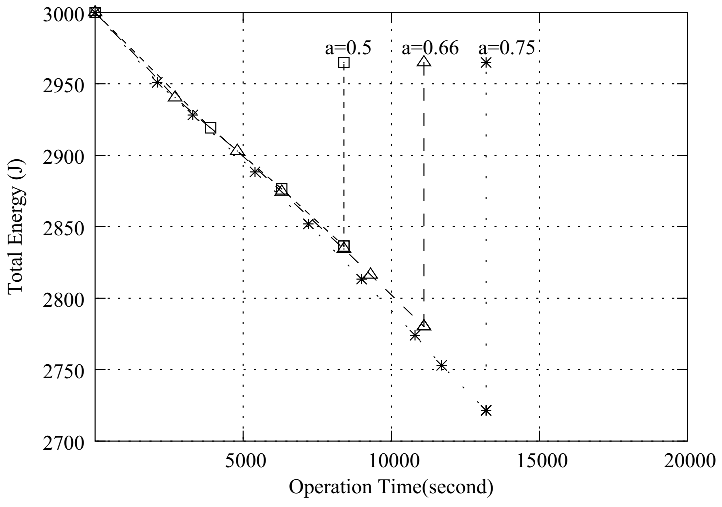
First we consider the metric of system lifetime. Lifetime is measured for the different algorithms as the number of sensor nodes increase. Adjusting the tree to avoid overloading critical nodes provides significant extension in system lifetime over the static `MinDist` and `WR`. To examine the system overhead induced by dynamic `ENCAST`, we assume the size of a control message is $0.1KB$. During the process of data collection tree adjustment, energy consumption due to control message exchange is also taken into account and treated the same way as energy consumed on sensor data transmission. A large α will trigger the tree adjustment more frequently than a small one. Thus, with the same number of sensors, a higher α always has a larger amount of energy consumption due to system overhead during the network operation time.

To see what happens inside a particular sensor network configuration under a fixed α, we trace the total residual energy of all the nodes in the network versus the system operation time. The summation of residual energy provides us with an indication of how "similar" the depletion rate has been across nodes. We would like the lifetime of a network to end when the sum of the residual energy over all nodes is as close to zero as possible. Figure 4.9(a) displays the total energy for the different algorithms in the sparse case of 100 nodes. Figures 4.9(b) and 4.9(c) show the same metric for the denser cases of 300 and 500 nodes respectively. For each algorithm, the total residual energy is calculated at the time of triggering the data collection tree adjustment and this is given as a point on the graph for that algorithm. The last point on each curve is the end of the network lifetime under the corresponding algorithm. We use vertical arrows to identify those "last" points.

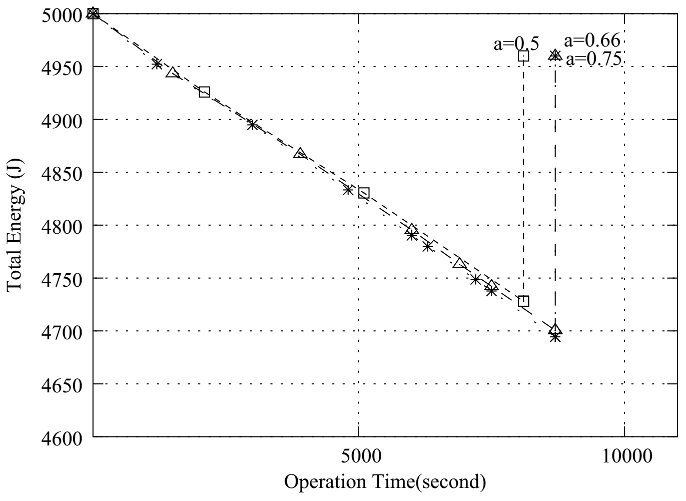
From Figures 4.9(a)–4.9(c) we find that with an increase in the node density, the threshold,α, should be raised correspondingly in order to achieve a longer lifetime. Since too frequent adjustment of the data gathering tree will introduce too much system overhead, we set the upper bound of α to 0.75. However, we also find that a higher α will not guarantee a longer system lifetime. This is because a large α possibly labels a sensor node energy critical too early and the tree that is rebuilt may waste too much energy by transmitting data along a circuitous route. If we



(a) 100 nodes



(b) 300 nodes



(c) 500 nodes

Figure 4.9: Lifetime & total residual energy.

assume the distribution of sensors is uniformly random, the number of the sensor nodes close to the sink (placed at the corner) is small in a sparse network. Then a large α means overload of other crucial nodes near the sink at an inappropriate time. With more sensors employed in the fixed size region, the density of the sensor nodes is also increased and our algorithm can find more candidates around the sink to share the load.

To further explore the performance of our dynamic ENCAST, we also compare against another proactive approach, Flow Augmentation (FA [13]) which routes sensor data from each sensor to the sink along a shortest path set up according to a particular link cost metric. To determine each link cost, energy consumption on transmitting and receiving unit data packet, and the residual energy at two involved end nodes are considered. By routing sensor data from each source node to the sink along the currently shortest path, FA avoids overloading the nodes with low residual energy, therefore extending the time to the first transmission failure because of energy outage. During the execution of FA, all chosen shortest paths have to update their link cost due to energy consumption.

FA is derived from the linear programming model formulated for maximizing the system lifetime. This model helps FA obtain the closely maximized network lifetime under a centralized framework. A simple distributed Bellman-Ford [19] shortest path algorithm can be utilized over each link's cost which may be updated frequently. This makes FA can be implemented in a distributed fashion and an ideal component algorithm for our distributed heuristic, ENCAST. However, to achieve the nearly optimal performance, FA has to frequently update the cost of each link along which the sensor data is routed recently (the residual energy at two end nodes of each link decreases). This will impose a large amount of energy consumption on transmitting control messages across the network. In order to evaluate how "bad" the performance obtained by ENCAST, comparing with an ideal FA, we implement FA as a centralized algorithm without system overhead on updating link cost. To fairly compare ENCAST against FA, we update link cost information in FA at the same time when ENCAST adjustment is triggered.

To examine the workload balancing effectiveness of dynamic ENCAST, we do

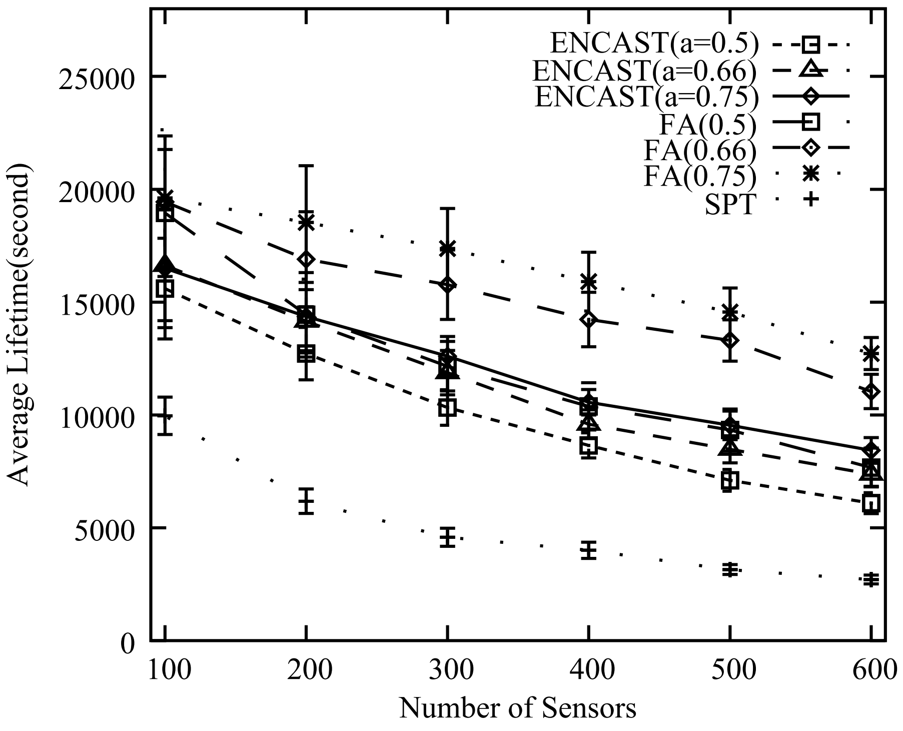


Figure 4.10: System lifetime without data aggregation.

not perform any data aggregation for the first set of experiments. Figure 4.10 shows system lifetime obtained from our dynamic ENCAST and opponent FA with different α. We find that the performance of our distributed algorithm considering system overhead is very close to the centralized FA, especially when α is small, namely link cost knowledge update of FA is not frequent. For algorithm FA, with the increment of α, the corresponding system lifetime derived is also improved. The reason behind is that there is no energy consumed on network link knowledge update for FA. In addition, routing decision made based on the recent node residual energy will balance the transmission load among the entire network as even as possible, therefore extending system lifetime. We also compare with a static SPT (MinDist) which outperforms other schemes in our previous experiments. The simulation results verify that applying dynamic tree adjustment to release heavy transmission duty from energy-critical nodes will significantly extend system lifetime.

Although extending system lifetime is the key task of ENCAST, data collection delay is also important to data collection, which represents the response speed to a data gathering request. Therefore, we use diameter of data gathering tree (D)

which is a reasonable indicator of delay in the absence of information specific to the MAC, to evaluate the resulting data collection structure generated by ENCAST. From Figure 4.11, we conclude that the data gathering trees generated by ENCAST have almost the same maximum hops (to the sink) as SPT. On the other hand, FA keeps choosing a less loaded path for routing data flow for the current pair of source and destination, which consequently makes FA have a longer network diameter than ENCAST.

In the second set of experiments, we compare ENCAST to FA again under the same environment. However, data aggregation is considered at this time. Since FA does not explicitly take into account the impact of data aggregation, we introduce *flow loss multiplier* which will also be utilized in our flow-based data collecting model, into the experiments. *flow loss multiplier* is a fraction number ($m_{ij} \in (0, 1)$) denoting the percentage of data volume (generated at source i) left after data aggregation which can be performed at any on-path node j. Although ENCAST is a tree-based data collection algorithm which benefits much more from data aggregation (lots of subtrees converge at a unique parent node) than FA, to fairly evaluate the impact of aggregation on both algorithms, for each pair of source and destination, data aggregation is only performed once at the on-path (on-tree for ENCAST) node j possessing the smallest m_{ij} where i is the data source. The simulation results are shown in Figure 4.12 and Figure 4.13. The ranking order is similar to the previous result in terms of both lifetime and data collection structure diameter. Since aggregation extends system lifetime, the gap between FA and ENCAST is enlarged because with the extended operation time, FA distributes data flows with a finer granularity and without any cost on system overhead. The diameter almost remains the same as before, which indicates that aggregation has little effect on the maximum hops of the generated data gathering structure.

4.3 Summary

Designing an efficient data collection tree is a key challenge in sensor networks. To address this problem, we first described five distributed heuristic algorithms, which

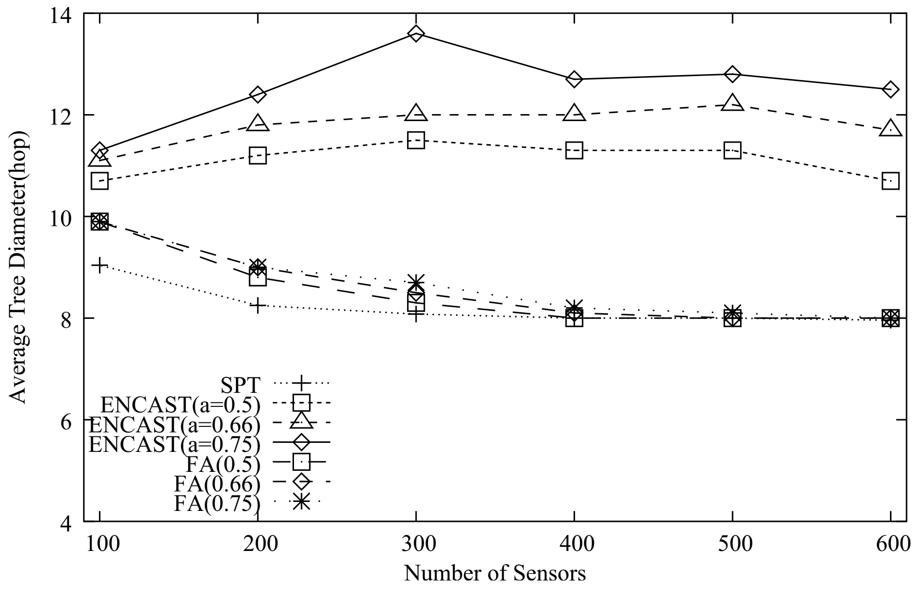


Figure 4.11: Averaged diameter of data gathering trees without data aggregation.

take into account the removal of data redundancy. Our results have indicated that when restricted to heuristics that operate using exclusively local information, and if the hardware complexity of the sensors is low (equivalently: using sensors that do not possess location information, neither by GPS, nor by any other means), it is still possible to effectively form the aggregate data tree in a way that exploits the reduced data traffic from sensor data aggregation. The results are particularly encouraging when one considers the fact that `MaxSubset` and `MaxDegree` require no complexity beyond the piggybacking of $\mathcal{N}_i$ on DV messages. At little added hardware cost `MinDist` appears to outperform the rest, suggesting a possible potential for greedy approaches, i.e., communicate with the node closest to you, since it represents the highest degree of overlap. On the other hand, distributed MST requires a large number of control message exchanges and location knowledge of all sensors as well. In addition, compared with our hop based SPT heuristics, `MMST` always has much larger total hops which is directly proportional to the total data collection delay. Despite its close optimality in energy cost, it is still desired to trade it for a more practical and simple algorithm with scalability especially in WSNs.

When considering extending system lifetime, the intrinsic limits of gathering data along fixed tree structure directs our efforts to release heavy traffic load on

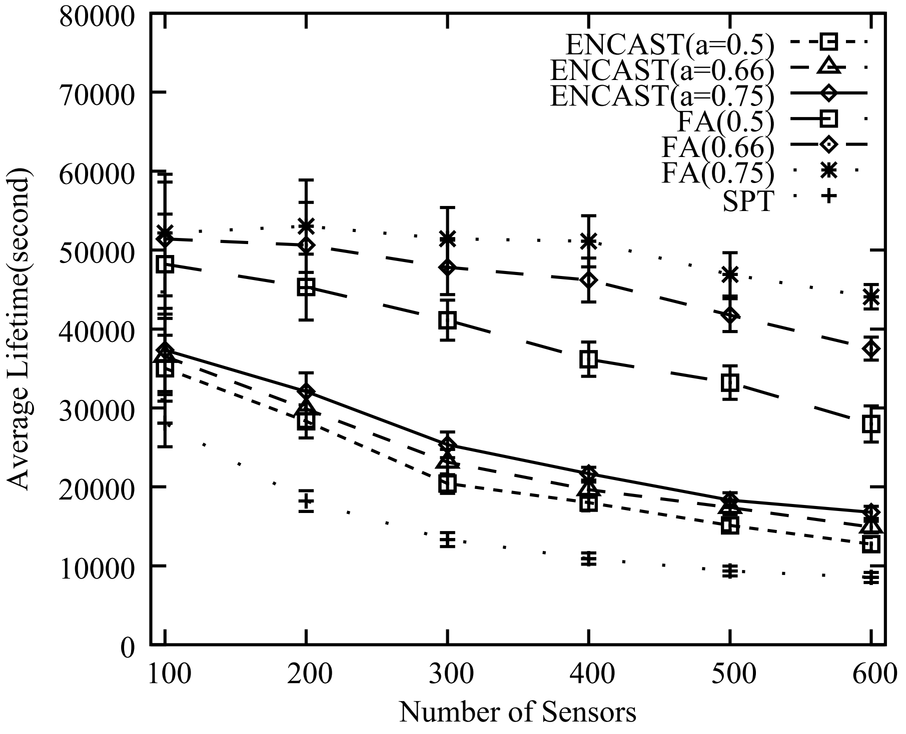


Figure 4.12: System lifetime with data aggregation.

some frequently accessed nodes so as to balance energy consumption across the network. We propose `ENCAST` to construct data collection trees with energy-critical nodes as the leaves. When this is not possible, the algorithm tries to balance the load on the critical nodes. We also propose an iterative application of this algorithm to handle cases where nodes become critical over time. Our performance study shows that our distributed algorithm significantly improves system lifetime compared to static tree-based data collecting schemes (e.g. `SPT`) with small amount of control message transmission and short data collection delay.

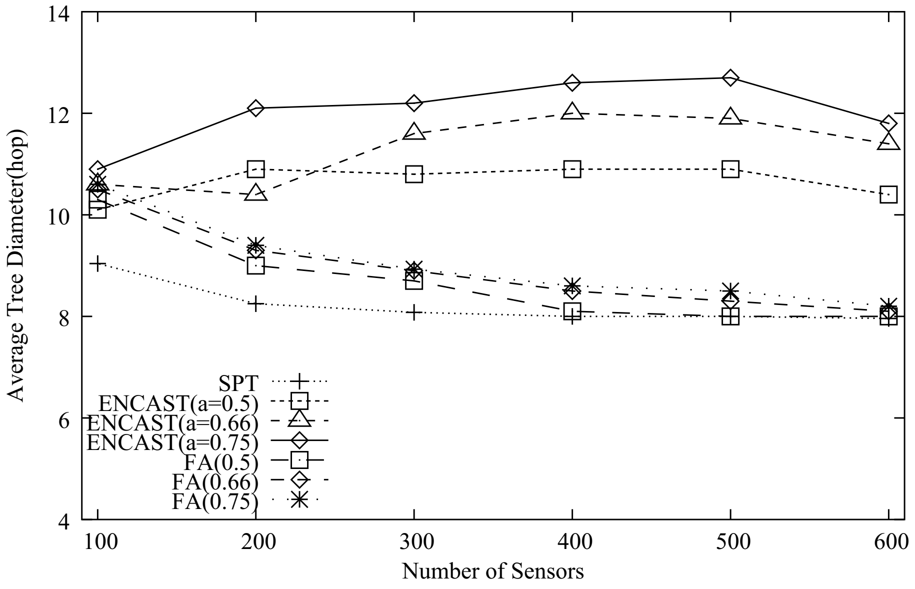


Figure 4.13: Averaged diameter of data gathering trees with data aggregation.

Chapter 5

Flow-based Data Collection Schemes

Based on the simulation results of `ENCAST` shown in the last chapter, we have observed that, besides minimizing total energy consumption across the network, wireless traffic balancing is another important aspect which has significant impact on system lifetime. Any traffic load imbalance will result in biased energy consumption among nodes along frequently used paths; while nodes not involved in relaying large data volume for other sources retain high energy reserves. As we discussed in Chapter 2, many flow-based multi-path data gathering schemes have been proposed to alleviate heavy traffic concentration on “hot-spot” nodes. However, they all focus on routing flow segments along different transmission paths to the sink for the sake of traffic balancing. It would be beneficial to introduce in-network aggregation into flow-splitting approaches for traffic reduction as well as load balancing. In this chapter, we combine these two techniques together.

We advocate a simple and powerful compromise by noting that on one hand, aggregation is hindered when flows are split, while on the other, splitting flows facilitates balanced energy consumption across different paths, as opposed to single path routing. The synthesis presented in this chapter is based on sending sensor data one or several hops away from the sensor data source, performing the aggregation there (which due to data correlation will probably be quite effective) and only then split this flow across different paths for the sake of balanced network use. Therefore, the proposed schemes perform both data aggregation and flow splitting (load balancing) in order to benefit from both techniques.

In this chapter, we first explain how *flow loss multiplier* works in flow-based data collection schemes. A mixed linear/integer programming (LP/IP) model is formulated for the problem of determining the optimal aggregation nodes that maximize network lifetime. Then, we provide a simplified but practical aggregation approach, First Hop (FH) data aggregation. Heuristics for FH aggregation are proposed to obtain significant redundant transmission reduction and to prolong the system lifetime. To facilitate performance evaluation, we adopt a flow loss multiplier that depends on the spatial relationship among sensed areas. Simulation results show that FH aggregation provides significant increase in network lifetime for both flow-based and tree-based delivery schemes, and that flow-based mechanisms provide better network lifetime than tree-based but at the cost of added complexity and overhead.

5.1 Maximum Lifetime Data Collection with Aggregation

In order to introduce data aggregation into our flow-based data collection framework, we propose several flow-based aggregation schemes in order to exploit rich data correlation and reduce wireless data traffic in WSNs. To present our data aggregation schemes, we first introduce some concepts which are also important to the problem formulation for the flow-based data collection. For each data flow (D_c) generated by sensor node c within each data collection round, it is desired to choose an efficient aggregation node j as the reference information (D_j) source to reduce data redundancy. Only the aggregated data volume D'_c, where $D'_c < D_c$, is sent out from j after aggregation. Thus, along the data delivery path from j to the sink, all nodes involved will save energy by relaying the reduced data volume, D'_c.

Since it is difficult to perform data aggregation after flow splitting, the routing protocol has to guarantee the integrality of the raw sensor data at least up to the point of aggregation for those data flows over which data aggregation is going to be performed. Therefore each data flow is assumed to be routed along a single path until it reaches the aggregation node. After that, for the sake of workload balance, aggregated data is free to be split into several flow segments and each one may be transmitted along a different path. To denote the impact of data aggregation on original data flow D_c, we use *flow loss multipliers* which have been explained in our tree-based schemes. If aggregation is performed at node j over D_c, $D'_c = D_c * m_{cj}$ is the data volume after aggregation.

We formulate the problem of Maximum Lifetime Data Collection with Aggregation node Selection (MLDCAS) as an IP/LP optimization model as follows (Figure 5.1). We assume all sensor nodes are initially assigned the same amount of energy. After the nodes have been deployed, a communication graph $G = (V, E(R_t))$ is derived, where vertex set V contains all sensors plus the sink and edge set $E(R_t)$ is determined by sensor transmission range R_t. We also use $\mathcal{N}_i$ to denote all i's neighbors which are located around i within Euclidean distance R_t.

$$\begin{aligned}
&objective: \ \min \ EC \\
&s.t.: \\
&X^c_{c,sink} = 1 \quad (c \in \mathcal{N}_{sink}) && (1) \\
&\textstyle\sum_k Y^c_{kj} = 1 \quad (c \in V \setminus \ (\mathcal{N}_{sink} \cup \{sink\})) && (2) \\
&\textstyle\sum_k Y^c_{kj} \times m_{cj} + \sum_{r \in \mathcal{N}_j} X^c_{rj} - \sum_{i \in \mathcal{N}_j, i \neq c} X^c_{ji} = 0 \quad (c, j \in V \setminus \{sink\}) && (3) \\
&\textstyle\sum_{c \in V \setminus \{sink\}, j \in \mathcal{N}_i} e^t \times D_c \times X^c_{ij} + \sum_{c \in V \setminus \{sink\}, k \in \mathcal{N}_i} e^r \times D_c \times X^c_{ki} + \\
&\quad \textstyle\sum_{p^c_{kq}: i \in p^c_{kq}, i \neq q} (e^t + e^r) \times D_c \times Y^c_{kq} + \sum_{p^c_{kq}: i = q} (e^r \times D_c \times Y^c_{kq} + e^t \times D_c \times Y^c_{kq} \times m_{cq}) \\
&\quad \leq EC \quad (i \in V \setminus \{sink\}) && (4) \\
&\textstyle\sum_{(c,j) \in l} \sum_k Y^c_{kj} < |l| \quad (l \in L) && (5) \\
&0 \leq X^c_{ij} \leq 1 \quad (\forall c, i, j) && (6) \\
&Y^c_{ki} \in \{0, 1\} \quad (\forall c, k, i) && (7)
\end{aligned}$$

Figure 5.1: IP/LP formulation of MLDCAS.

Besides the symbols we explained in the previous chapter (refer to Table 4.1), we also introduce several new variables and list them in Table 5.1. In this model, we enumerate all qualified paths from each source node c ($c \in V \setminus \ (\mathcal{N}_{sink} \cup \{sink\})$)

to any of its possible aggregation node j ($j \in S_c$). We introduce S_c to denote c's correlation subset which contains any node possessing correlated sensor data to c. S_c can be derived from a particular WSN scenario. Generally speaking, it may contain any node except the sink and itself ($j \in V \backslash (\{sink\} \cup \{c\})$). We use a $0-1$ integer variable Y_{kj}^c to represent whether path p_{kj}^c is chosen or not. Here k is the path index, j is the aggregation node of path p_{kj}^c, and m_{cj} is flow loss multiplier denoting data correlation between c and j. Although index k identifies the exact path along which data is routed until the end point (aggregation node j), we explicitly use j to denote a path for convenience in the problem formulation (Figure 5.1). The other group of variables are X_{ij}^c's. Each of them is a fractional number between 0 and 1, denoting the portion of data volume (before aggregation) generated at c and routed from i to j.

Table 5.1: Summary of Notation

Notation	Definition
p_{kj}^c	k-th path from c to j
Y_{kj}^c	integer variable, denoting whether path p_{kj}^c is chosen or not
X_{ij}^c	the portion of data volume (before aggregation) generated at c and routed from i to j
EC	sensor energy consumption within a time unit (energy consumption rate)
e^t	energy consumption on transmitting a data unit
e^r	energy consumption on receiving a data unit
D_c	data demand (Kb/s) of sensor c
$h(c)$	aggregation site of sensor c
l	an element of L
L	set of all the possible path loops resulting lost in data aggregation
S_j	node j's data correlation subset, any node within it will generate correlated data with j

The optimization variable is EC, which is the minimal energy consumption across the network within a time unit, for instance 1 second. By assuming unified initial energy volume, we can figure out the maximized lifetime by dividing the amount of initial energy by the minimized EC (over all sensor nodes) which is determined by the node with the most heavy traffic load. Constraint (1) makes sure every sink's neighbor node have to transmit its sensor data directly to the sink. Constraint (2) is used to limit each data source to only send out its sensor data once within a time unit. Y^c_{kj} is an $0-1$ integer number denoting whether the sensor node is routed along the path between source node c and its potential aggregation node j which is indexed as k-th path within the path set of c. Constraint (3) is introduced for flow conservation. Considering each data flow coming out of source c, for any intermediate node i, if i is the aggregation node of c ($i = j$), flow loss has to be taken into account after aggregation (the unit sensor data will be aggregated at i and only a data fraction m_{cj} flows out of i); otherwise flow conservation holds, which means the total amount of flow out of i has to be equal to the total amount of flow into i. Constraint (4) denotes that energy consumption rate on each sensor node can not exceed EC. D_c is the data demand (Kb/s) generated by c. e^t and e^r denote energy consumption on transmitting and receiving a data unit. In order to avoid any data loss in aggregation, we add constraint (5) to restrict any loop from forming in the aggregation node assignment. For example, if sensor data of source c is aggregated at node j, then j's sensor data can not be aggregated at c. Constraints (6) and (7) are utilized to denote that all Xs are fractional numbers and all Ys are either equal to 0 or 1.

We introduce function $h : V\backslash\{sink\} \to V$, which maps each data source c to its aggregation node $h(c)$. Consequently, a directed graph $G' = (V, A)$ can be derived where $A = \{< c, h(c) > | c \in V \setminus \{sink\}\}$ based on a particular aggregation node assignment. On G', each "logical edge" $< c, h(c) >$ is possibly a path which is composed of many physical edges. L is the logical loop set containing all possible loop ls, and $|l|$ is the number of hops in loop l. For instance, if $c \in S_j$ and $j \in S_c$ then logical edges $< i, j >$ and $< j, i >$ do not belong to A at the same time. To introduce this constraint into `MLDCAS` (Figure 5.1), Y^c_{kj} and Y^j_{kc} can not be set to 1 simultaneously.

5.2 First Hop In-network Aggregation

The number of qualified paths in the problem of `MLDCAS` exponentially increases with the increment of the network scale. Therefore, it is desired to avoid enumerating all possible aggregation sites for each data source and the large number of paths connecting this source and any particular aggregation node. To remedy this problem, we propose that the aggregation node for a sensor (data source) be the next hop node on the path to the sink node, called the First Hop parent. This is motivated by the fact that physical phenomena monitored by WSNs tend to behave in a localized fashion with respect to Euclidean distance. Spatial data correlation exists in a wide range of WSN applications. In our first hop aggregation scheme, the flow loss multiplier is a function of the Euclidean distance between data sources which characterizes spatial correlation among sensor data. This is also verified by the traces obtained from real WSN scenarios [41]. Another consideration for limiting the scope of aggregation to one-hop neighbors is that identifying correlation with distant sources usually involve non-trivial communication and computation, and knowledge of application details as well. Additionally, first hop in-network aggregation sharply narrows the number of paths down to the cardinality of each sensor's one-hop neighbor set, which makes it an appropriate candidate to balance data redundancy removal and the feasibility of our solutions.

The process of selecting a first hop parent with which to aggregate in a pairwise fashion is essentially equivalent to choosing an upstream parent node in a tree rooted at the sink. Many nodes can choose the same upstream parent but the performed aggregation is considered to be pairwise at all times. Some redundancy may be left in the aggregated data. This is acceptable because it means that one data stream does not have to wait for another data stream to arrive in order to perform 3-way or n-way ($n > 2$) aggregation. Before the first hop parent forwards the data, it reduces the volume of data by applying a data aggregation scheme suitable to the particular application. After first hop aggregation, the aggregated data may be split and delivered along several paths to the sink.

To extend the lifetime of a WSN where rich data correlation mainly exists between the source and its one-hop neighbors, we combine First Hop (FH) in-network aggregation and load balanced data flow splitting and delivery together. Then, we reformulate the optimization model of Figure 5.1 as the Optimal First Hop Aggregation Routing (`OFHAR`) problem. This is shown in Figure 5.2 which simplifies but approximates the problem of `MLDCAS`. By introducing `OFHAR`, the search space of the aggregation node for a data source i is narrowed from the entire network except the sink ($V - \{sink\}$) down to the source node's neighbor set ($\mathcal{N}_i$). The optimization variables include FH data routing decision variables Y's, edge flow variables F's (within a time unit) and EC. Y_{ij} is a $0-1$ integer variable, whose value is 1 if and only if source i chooses j as its FH parent. F_{ij}^c denotes the flow generating from c and going through the edge (i, j), which is always greater than or equal to 0. The objective is still to minimize EC. The first set of constraints are used to denote that each commodity source i has to choose a unique node within $\mathcal{N}_i$ as its FH parent. The second set of constraints ensure that once the FH parent is fixed, all the information has to be totally transferred from the source to its FH parent.

The third set of constraints are derived from the commodity conservation. Each data flow have to be routed to the sink node. Along each path, all the nodes (except the sink) can be divided into 3 categories: the source node i, the FH parent (aggregation node) and the nodes on the rest of the path. At source node, D_i is generated in each time unit. The flow is aggregated at the FH parent node and the flow left after aggregation is determined by the flow loss multiplier. Flow commodity conservation is maintained on the other on-path nodes. The fourth set of constraints are used to represent that the energy consumption rate (EC) of each sensor node (i) is determined by its wireless transmission and reception. The fifth set of constraints are utilized to avoid creating any loops during the process of FH aggregation parent selection. Compared with the corresponding constraints in Figure 5.1, the constraints are much simpler because only 2-hop aggregation loop may happen in FH data aggregation. The next set of constraints ensure all variable Ys are $0-1$ integers. At last, all flow segments has to be larger or equal to 0.

5.2.1 Maximum Lifetime with Explicit First Hop Aggregator Selection

To tackle the problem of `OFHAR` in an efficient way, we can divide the problem into two phases that are solved separately. In Phase 1, for each data source i, we determine the FH parent $h(i) \in V$. The sensor data will be transmitted from each i to $h(i)$ and aggregated there ($h(i) \neq 0$). Energy will also be reduced accordingly

$$\begin{array}{l}
objective: \ \min \ \ EC \\
s.t.: \\
\sum_{k\in\mathcal{N}_i} Y_{ik} = 1 \quad (i \in V\setminus\{sink\}) \\
F^c_{ci} = D_c * Y_{ci} \quad (c \in V\setminus\{sink\}, i \in \mathcal{N}_c) \\
\sum_{j\in\mathcal{N}_i, j\neq c} F^c_{ij} - \sum_{k:i\in\mathcal{N}_k, k\neq c} F^c_{ki} = \begin{cases} D_i \\ (i\in V\setminus\{sink\}, i = c) \\ \\ 0 \\ (i\in V\setminus\{sink\}, i \notin \mathcal{N}_c) \\ \\ m_{ci} * D_c * Y_{ci} \\ (i\in V\setminus\{sink\}, i \in \mathcal{N}_c) \end{cases} \\
\sum_{c,j\in\mathcal{N}_i} e^t * F^c_{ij} + \sum_{c,k:i\in\mathcal{N}_k} e^r * F^c_{ki} \le EC \quad (i\in V\setminus\{sink\}) \\
\sum_{(i,j)\in l} Y_{ij} \le |l| \quad l \in L \\
Y_{ij} \in \{0,\ 1\} \qquad (\forall i, j) \\
F^c_{ij} \ge 0 \qquad (\forall i, j, c)
\end{array}$$

Figure 5.2: LP/IP formulation of OFHAR.

at the transmitter and receiver. In Phase 2, given a concrete FH parent assignment, FH aggregation can be performed. This phase can be viewed as a new multicommodity network flow problem with updated commodity sources and demands, and the lifetime T can be obtained from solving the LP built on this updated model.

After FH data aggregation at node $h(i)$ which is determined by the solution derived from Phase 1, the aggregated commodity flow (originally generated at node i) is potentially split into multiple flow fragments and each of them is routed through a different path. Our objective is to figure out the maximum lifetime of the sensor network which can be achieved in Phase 2. Based on the same parameters, the formulation of the Maximizing Lifetime after First Hop Aggregation (MLFHA) problem is shown in Figure 5.3.

The optimization variables include EC and F^c_{ij}'s assuming $h(i)$ have been pre–determined. The first set of constraints are derived from a sensor node's limited

$$
\begin{aligned}
&objective: \ \min \ \ EC \\
&s.t.: \\
&\textstyle\sum_{c,j\in\mathcal{N}_i} e^t * F^c_{ij} + \sum_{c,k:i\in\mathcal{N}_k} e^r * F^c_{ki} \le EC \qquad (i,c \in V\setminus\{sink\})
\end{aligned}
$$

$$
\textstyle\sum_{j\in\mathcal{N}_i} F^c_{ij} - \sum_{k:i\in\mathcal{N}_k} F^c_{ki} = \begin{cases} D_i & (i\in V\setminus\{sink\}, i=c) \\ 0 & (i\in V\setminus\{sink\}, i\neq c, c\notin h^{-1}(i)) \\ m_{ci} * D_c & (i\in V\setminus\{sink\}, c\in h^{-1}(i), k\neq c) \end{cases}
$$

$$
\textstyle\sum_{j\in\mathcal{N}_i} F^c_{ij} - m_{ci}F^c_{ci} = 0
$$

$$
F^c_{ci} = \begin{cases} 0 & (i\in V\setminus\{sink\}, c\in\mathcal{N}_i, c\notin h^{-1}(i)) \\ D_c & (i\in V\setminus\{sink\}, c\in\mathcal{N}_i, c\in h^{-1}(i)) \end{cases}
$$

$$
F^c_{ij} \ge 0 \qquad (\forall i,j,c)
$$

Figure 5.3: LP formulation of MLFHA.

energy budget. During a time unit, the total energy consumed at node i on receiving and transmitting data flows should not exceed EC. The second set of constraints are set up based on network flow conservation. For the commodity source node ($i = c$), its net data volume sent out should be equal to D_i. If it is just an intermediate node for flow relaying ($i \neq c$) and it is not the current commodity source's FH parent ($c \notin h^{-1}(i)$), its total data volume sent out is equal to the total amount of data received. We use $h^{-1}(i)$ to denote the set of commodity sources that share i as their common FH parent. If node i is commodity source c's FH parent, the data volume D_c generated at c will be reduced to $m_{ci} * D_c$ after FH data aggregation is performed at i. To restrict the unique FH parent selection for source node c, a binary variable $F^c_{ci}, i \in \mathcal{N}_c$ is defined. That is, if $h(c) = i$, all the data flow generated at c will be transported to i; otherwise, no data flow from c will be received by i. Obviously, all variables are non-negative, while the commodity demands must be greater than zero.

5.2.2 FH Parent Heuristics

In this subsection, we propose several heuristic algorithms to choose the FH parent for each sensor node. First, we introduce several concepts to help describe the heuristics. As we presented before, $\mathcal{N}_i$ is the neighbor set of node i. In our FH aggregation model, it is also the candidate set for FH parent selection for node i. Based on the hop-count to the sink, each element of $\mathcal{N}_i$ can be classified into one of 3 subsets: $\mathcal{N}_i^-$, $\mathcal{N}_i^=$ and $\mathcal{N}_i^+$; representing the subset of i's neighbor nodes with fewer (only one hop), equal and more (only one hop) hop-counts to the sink, respectively.

Minimal Distance $\mathcal{N}_i^-$ Neighbor (`MinDistFH`)

The first algorithm is to select a FH parent from nodes one-hop closer, that is, from $\mathcal{N}_i^-$. If there exists more than one candidate, the minimal Euclidean distance is employed to break the tie, i.e. the node in $\mathcal{N}_i^-$ with minimal distance to i will be chosen. Compared with other candidates, choosing FH parent from $\mathcal{N}_i^-$ guarantees sensor data is delivered one hop closer to the sink, which implies energy efficiency.

$$\texttt{MinDistFH} : h(i) = \arg \min_{j \in \mathcal{N}_i^-} d_{ij} \tag{5.1}$$

The benefit of choosing the FH parent out of $\mathcal{N}_i^-$ is that we can guarantee there is no loop in FH parent selection. In addition, no matter how effectively aggregation can be performed, this method at least routes the data closer to the sink.

α–based $\mathcal{N}_i^- \cup \mathcal{N}_i^=$ Neighbor (α-`FH`)

Selecting $h(i)$ from $\mathcal{N}_i^-$ limits the available choices. We propose another scheme, $\alpha - \texttt{FH}$, which uses a distance threshold α to determine whether to choose $h(i)$ only from $\mathcal{N}_i^-$ or to consider $\mathcal{N}_i^=$ as well. However, enlarging the size of the candidate set will introduce potential FH loops. With the help of a sensor's unique identifier, we will avoid creating FH loops.

If R_s describes the radius around which the sensor senses the environment, and d_{ij} is the Euclidean distance between sensors i and j, then if $d_{ij} \geq 2R_s$, the probability that node i and j's data are correlated is small. We therefore use $\alpha * 2R_s$ ($\alpha \in (0,1)$) as a threshold to determine if candidate node k is "close enough" to i. For current node i, we first choose the closest node (k) from N_i^-, if d_{ik} is less than $\alpha * 2R_s$, then $h(i) = k$; otherwise we try to find a better candidate from $\mathcal{N}_i^=$.

We assume each sensor node has a unique ID, which can be assigned before WSN deployment. We also assume j is the node closest to i in $\mathcal{N}_i^{=}$. Compared with other candidates in $\mathcal{N}_i^{=}$, there is a larger probability that node i and j monitor the same target simultaneously, which implies better data aggregation can be performed at j. This assumption assures the maximized benefit by enlarging the candidate set for choosing FH parent. To avoid creating loops we force the node (i) with larger ID to choose a smaller ID node (j) as its parent when the two nodes have the same minimal hop-count to the sink, i.e. $i > j,\ j \in \mathcal{N}_i^{=}$. So, if $d_{ij} < d_{ik}$ and $i > j$, then $h(i) = j$; otherwise $h(i) = k$.

$\alpha -$ FH is a modified version of `MinDistFH`. Given a FH parent assignment obtained from `MinDistFH`, for every node i, compare the distance $d_{i,h(i)}$ with threshold distance $\alpha{*}2R_s$. If $d_{i,h(i)} \leq \alpha{*}2R_s$, leave $h(i)$ as i' FH parent; otherwise, try to find another neighbor node $k \in \mathcal{N}_i^{=}$, if $d_{ik} < d_{i,h(i)}$ and $i > k$, use k as the new $h'(i)$. We do not enlarge the candidate set to the entire $\mathcal{N}_i$, because we do not want to risk extra energy consumption by transmitting sensor data away from the sink.

Modified MST FH Parent Selection (`MMST`)

We observe that the FH parent-child relationships of the previous methods form a tree. They connect $N + 1$ vertices, using N edges (that is, each of the N non-sink nodes connects to a parent) and they are constructed such that there are no loops. The previous schemes consider only information that can be locally exchanged with neighbors in constructing the tree. This motivates us to consider other spanning tree constructs that minimize more global cost functions to represent parent-child connections, namely a Minimal (distance) Spanning Tree, `MST`. Note that there are distributed versions of MST [28].

To improve the energy-efficiency, we define a Modified Minimal Spanning Tree (`MMST`) to form the FH parent tree. By modified, we mean we first assign sink (0) as the parent to each of its neighbor nodes (located within R_t to the sink); then, the `MMST` is built based on this star topology. During this latter process, the Euclidean distance between two end nodes of each edge is used as the cost of that edge. The modification improves energy efficiency since the sink is the destination for all commodities and it has no energy constraints, hence it will not be a node that restricts system lifetime. `MMST` is expected to perform better than `MST`.

Random FH Parent Selection (`RandomFH`)

We also introduce another heuristic to select FH parent node for each data source via uniformly randomly choosing the FH aggregation node among all the candidate sensors (the neighbors of the current source node). We use it as a benchmark to evaluate other heuristics to check if our heuristics with different "intelligence" will outperform the random FH parent selection.

5.2.3 Performance Evaluation of First Hop Aggregation

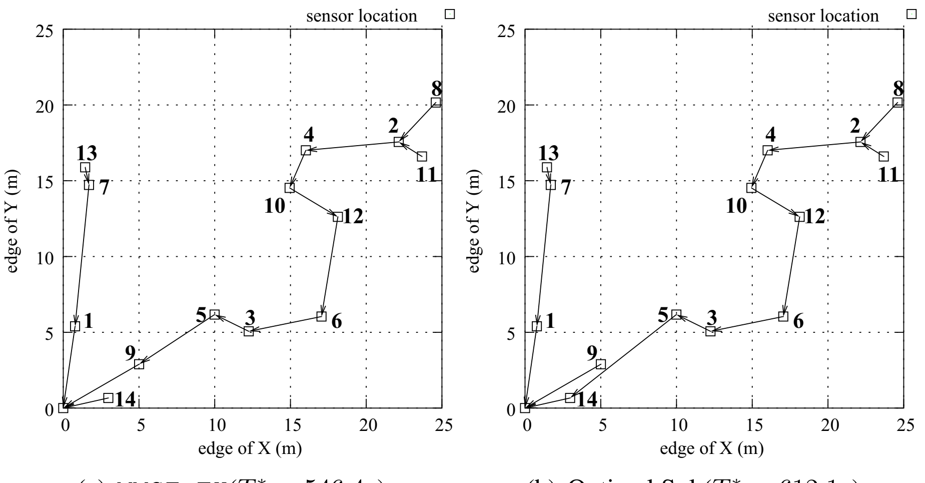
In this subsection, we first briefly motivate the effectiveness of our FH aggregation heuristic algorithms by comparing our heuristics with the optimal solution of `OFHAR` for small networks. Clearly, due to its computational complexity, the optimal solution can only be determined for small "toy" network instances. Then, we evaluate the performance of the heuristic FH aggregation tree building algorithms, `MinDistFH`, $\alpha-\mathtt{FH}$, and `MMST-FH` in terms of maximum lifetime, message overhead and maximum pathlength. Both flow-based and tree-based aggregation mechanisms are considered and a comparison is made to a flow-based non-aggregation mechanism as well. We also compare our algorithms with a minimal cost matching-based energy-efficient data aggregation scheme.

Effectiveness of FH Parent Heuristics

We use the first aggregation strategy and set the flow loss multiplier to a function of the Euclidean distance ($m_{ij} = f(d_{ij})$) between the two nodes that non-increasing, and satisfies $f(0) = 0$ and $f(d_{ij}) = 1$ when $d_{ij} \geq 2R_s$. Specifically, in the experiments described below, we assume the data aggregation parameter, m_{ij}, is determined by the size of overlapped sensing coverage of node i and j, i.e. $m_{ij} = \frac{(|A_i| \cup |A_j| - |A_i|)}{|A_i|}$, where A_i is the sensed disc of node i. We also restrict our attention to WSNs distributed in 2-dimensional space. and perfectly circular (with radius R_t) transmission coverage, i.e., $E(R_t) = \{(i,j)|d_{ij} \leq R_t,\ i \neq j,\ i,j \in V\}$, that is $E(R_t)$ contains all pairs of nodes (i,j) where the Euclidean distance between them (d_{ij}) does not exceed R_t. Thus, the neighbor set is defined as $\mathcal{N}_i = \{j|d_{ij} \leq R_t,\ i \neq j,\ j \in V\}$, i.e. any node j located within R_t distance is i's neighbor. Here symmetry is assumed, i.e., if $j \in \mathcal{N}_i$ then $i \in \mathcal{N}_j$ and vice versa. We also assume that $R_t \geq 2R_s$, thus ensuring that two sensors with overlapped sensing coverage (potential data aggregation) are capable of communicating directly with each other.

We first compare the FH parent selection results obtained from our heuristics to the optimal solution for small cases of 15 sensor nodes including the sink. The 14 sensors are randomly placed in a $25 * 25m^2$ area. All nodes are assigned the same initial energy except the sink. We use exhaustive search for parent choices to obtain the optimal solution to `OFHAR` (refer to Figure 5.2). We find the FH parent trees obtained from `MMST-FH` are exactly the same as the optimal solutions for more than 20 out of a total of 30 network configurations. In the other cases, there is only 1 or 2 different edges between these two FH parent trees. Below we show the FH parent trees for the optimal and heuristic cases for one of the 15-node networks.

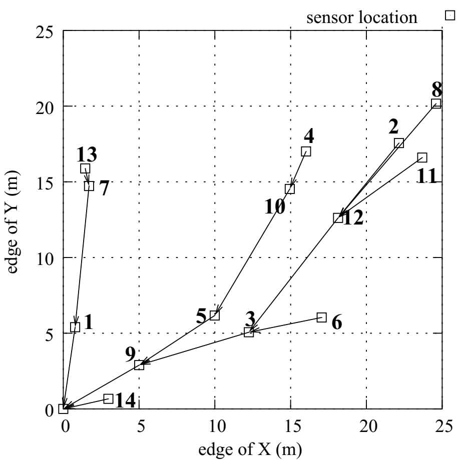
The FH aggregation parent assignments obtained from `MinDistFH`, $\alpha - $`FH` and `RandomFH` are shown in Figure 5.4(a), 5.4(b) and 5.4(c).



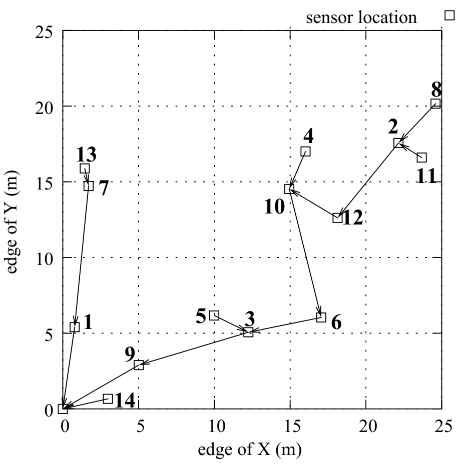
(a) `MMST-FH`($T^* = 546.4s$) (b) Optimal Sol.($T^* = 612.1s$)

Figure 5.5: `MMST-FH` vs. optimal FH parent assignment.

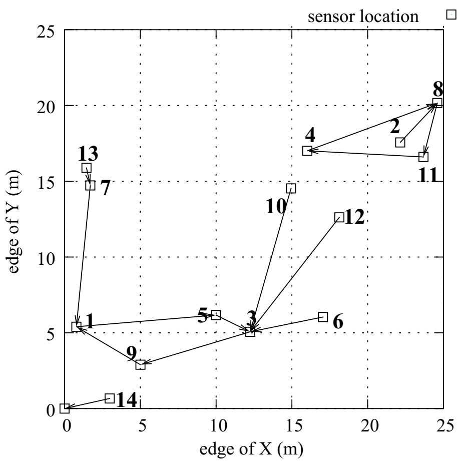
Here T^* represents the maximum lifetime obtained via the flow-based routing scheme (refer to the LP model in Figure 5.3) after the corresponding FH aggrega-



(a) MinDistFH ($T^* = 403.6s$)



(b) $\alpha -$ FH ($T^* = 553.6s$)

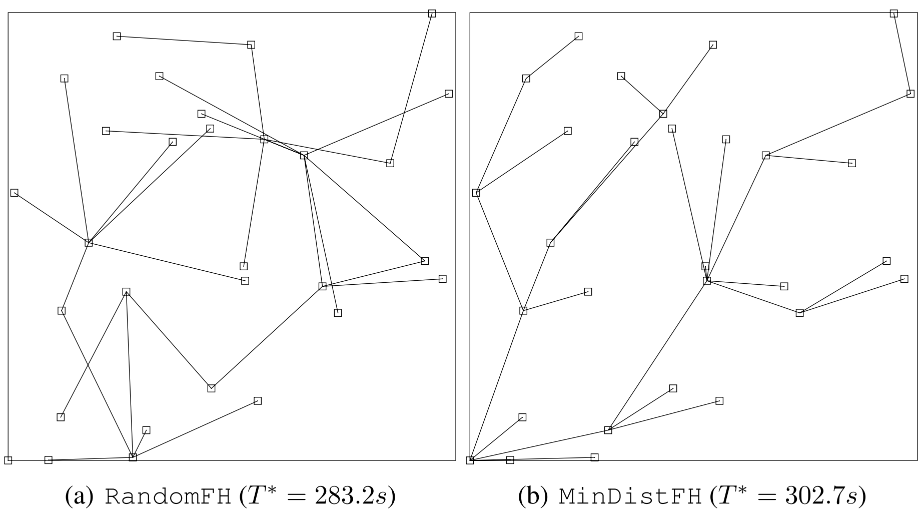


(c) RandomFH ($T^* = 389.3s$)

Figure 5.4: MinDistFH, $\alpha - \texttt{FH}(\alpha = 0.1)$ and RandomFH parent assignment.

tion. Figure 5.5(a) shows the FH parent tree obtained from MMST-FH; while Figure 5.5(b) shows the optimal FH parent tree. Note that the optimal FH tree chooses node 14 as 5's FH parent instead of 9 (in MMST-FH) so as to relax the traffic burden on 9, and the corresponding system lifetime (computed by the LP) is prolonged as well. We observed that our heuristic algorithm MMST-FH and $\alpha -$ FH can provide a near optimal FH parent selection for small networks.

To gain further insight into the behavior of the heuristics, the FH parent trees for the different aggregation heuristics are shown for an example of 30 nodes in Figures 5.6. From these figures, we find that our heuristics show more structure than this particular random FH parent assignment. MinDistFH tends to have higher in-degree at nodes causing it to overload some nodes thereby reducing lifetime; while


(a) RandomFH ($T^* = 283.2s$) (b) MinDistFH ($T^* = 302.7s$)

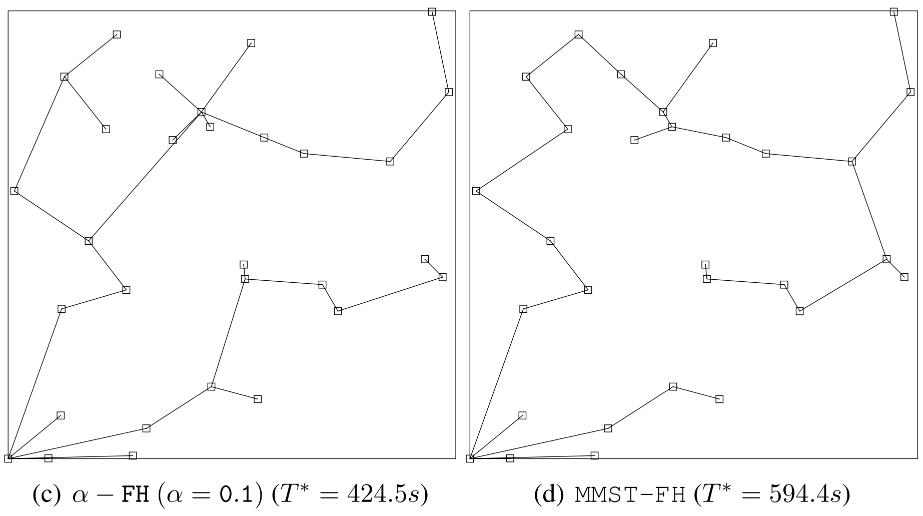

(c) $\alpha -$ FH ($\alpha =$ 0.1) ($T^* = 424.5s$) (d) MMST-FH ($T^* = 594.4s$)

Figure 5.6: FH parent assignments for 30 node networks.

the wider choice of parents of $\alpha -$ FH helps reduce load. MMST-FH tends to cause more chain-like behavior due to its focus on distance. Both $\alpha -$ FH and MMST-FH increase the gains from aggregation by choosing closer parents as demonstrated by the existence of shorter line segments in the graphs compared to MinDistFH and this example of RandomFH.

Then, we simulate the sensor network under different multi-hop data gathering schemes with/without FH data aggregation. We assume N nodes are distributed within a $25*25(m^2)$ plane area. The sink (0) is placed at the origin of the coordinate system. All the other $N-1$ sensor nodes are uniformly randomly distributed within the same area and assumed to have the same initial energy (10J), and each of them generates sensor data at a constant rate (1Kb/5min), which is same to Chapter 4. We adopt a simple energy consumption model [34]. We assume each sensor node's sensing range is the area surrounding it up to 5 meters ($R_s = 5m$). Its wireless transmission range is 10 meters ($R_t = 10m$). The results are averaged over 60 runs of different sensor network configurations and the error bars are two sided 95% confidence intervals.

For small devices such as sensor nodes, energy consumption on receiving and transmitting a unit of data is close. For the convenience of analysis, we introduce a ratio factor of e_r/e_t, γ, to describe the difference between them, where $0 < \gamma \leq 1$. For almost all the on-path nodes of a data flow, their transmitting and receiving data volume is same. Different γ only change the amount of their energy consumption with a unified factor. On the data aggregation node, incoming data volume will be reduced according to the corresponding flow loss multiplier. Compared with the entire network, the flow changes on those aggregation nodes are minor. Therefore, the substance of the simulation result is insensitive to γ. In our experiments, the energy spent on sending and receiving sensor data are set to 100nJ/bit and 30nJ/bit, respectively. The major parameters set for the simulations are listed in Table 5.2.

Table 5.2: Simulation Parameters

Parameter	Value
Unit of lifetime	second
D (flow data demand)	$1Kb/300second$
E (initial energy)	$10J$
γ	0.3
size of area of interest	$25 * 25(m^2)$
R_s (sensing range)	$5m$ and $10m$
R_t (transmitting range)	10m
e^t	100nJ/bit
e^r	30nJ/bit
α	0.1 and 0.5

Message Overhead for FH Parent Selection Heuristics

We first evaluate the message overhead of choosing parents for aggregation. This is defined as the total number of control messages used during FH parent tree formation. MMST-FH starts from a star structure in which the sink is the center and all the neighbors of the sink are connected to it. Then, the unexplored nodes are selected and connected to this star one by one according to the principle of building an MST. This process continues until all nodes are connected. Since building the initial star is no more complicated than building the MST, we use the number of control messages used during building the MST as the system overhead of MMST-FH. A distributed version of Minimum-Weight Spanning Trees [28] requires at most $5N \log_2 N + 2E$ message transmissions in a given graph with N nodes and E edges. Each message consists of enough bits to represent the length of an edge, $log_2 N$ bits to represent the node's ID and 3 additional bits to mark the message type.

To build a MinDistFH tree, the sink first broadcasts a message to claim its level is 0. After receiving that message, each neighbor node will choose the sink as their parent and broadcast to inform their neighbors that their levels are 1. This process will continue until every node sets up its level. During this process, most of the sensors may receive multiple messages and its level is determined by the message with the smallest level value. Each node only needs to send out one message. As we have discussed in Chapter 4, during the process of building a hop-based SPT, tie-breaking could be done by comparing the signal strength received at each sensor. So MinDistFH only requires N message transmissions with the same message size as MMST-FH. α-FH is a modified version of MinDistFH and assuming each node obtains all its neighbor distance information based on signal-strength measurements, its message complexity and its message size are similar to those of MinDistFH.

Performance of Flow-Based Mechanisms

We evaluate the flow-based heuristics for different network configurations based on the maximum lifetime obtained from the LP and the maximum pathlength (with respect to hop-count). In the first experiment, we compare the lifetime of the flow-based data delivery schemes including our aggregation heuristics with a non-aggregation scheme called General. The results are shown in Figure 5.7. General

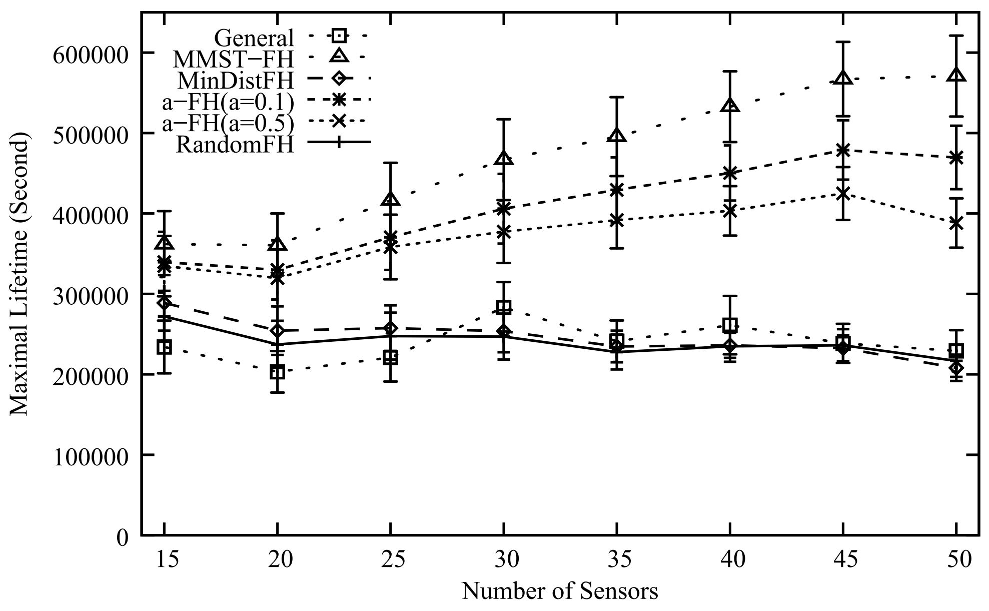


Figure 5.7: Maximum lifetime for flow-based mechanisms

represents the maximum lifetime achieved from the general multicommodity network flow-based LP where no data aggregation is performed. In this approach, data flow can be split at every source node and each fraction is routed along a different path. The most balanced energy consumption can be reached in the `General` LP because there is no constraint on flow splitting due to FH aggregation. However, without considering in-network data aggregation, lots of energy is wasted on transmitting and receiving redundant data and system lifetime is shorter than other schemes. Both `RandomFH` and `MinDistFH` have poor performance in terms of maximum lifetime. `RandomFH` performs similar to `MinDistFH` since most neighbors have some correlation. The $\alpha - \texttt{FH}$ schemes perform better than `MinDistFH` since they have larger candidate sets of parents. A smaller value of α corresponds to a greater improvement since it causes the system to more often consider parents that are close to the node and the same hop-distance away. However, a smaller α also requires accurate distance measuring capabilities. From Figure 5.7, we also observe that with an increase of sensor deployment density, the gap of maximum system lifetime between different α's becomes wider. In sparse WSNs, the cardinality of each node's neighbor set is small. As the distribution density increases, each source node has a larger candidate set, and the value of α has a more notable impact. `MMST-FH` performs the best among all FH parent selection heuristics in terms of system lifetime. This algorithm makes the most effort to choose nearby nodes as parents and therefore gains the most from aggregation. This demonstrates that an intelligent selection for FH parent assignment will result in significantly improved system lifetime particularly for denser networks. This observation is also an incentive to propose intelligent heuristic FH parent selection algorithms and evaluate their performance.

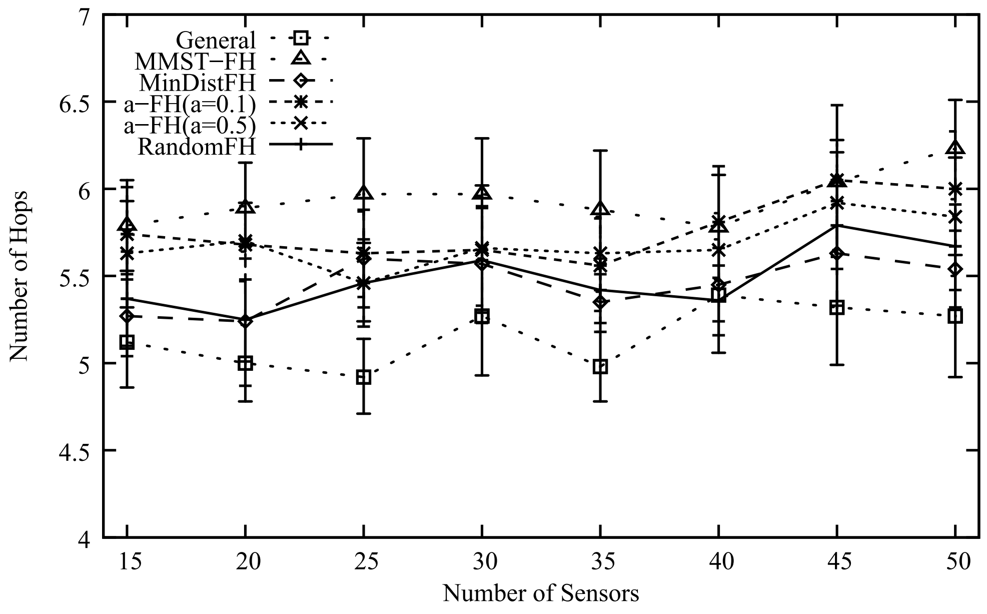


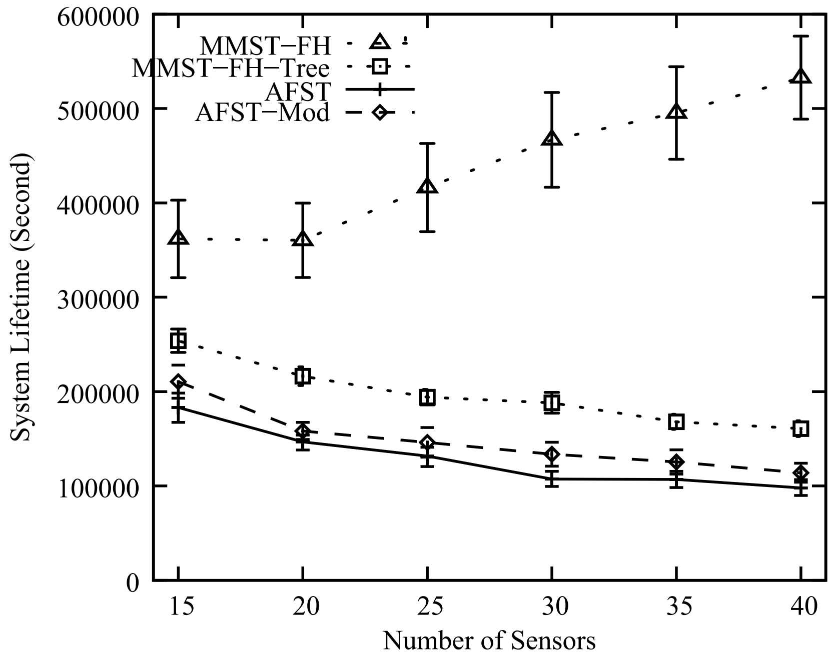
Figure 5.8: Maximum pathlength for flow-based mechanisms.

In Figure 5.8, we compare the maximum pathlength with respect to hop-count of the flow-based delivery schemes `General`, `MMST-FH`, `MinDistFH`, α`-FH(0.1)`. Maximum pathlength is defined as the maximum number of hops from any data source to the sink along the data gathering structure. The results are quite similar, since the LP decides the efficient delivery paths. The choice of a good FH parent can cause somewhat larger pathlength, however. In all cases, the maximum pathlength increases only slightly for more dense networks. In summary, aggregation does improve lifetime with a slight increase in pathlength. `MMST-FH` has the best lifetime; while α`-FH` also provides significantly longer lifetime than the mechanism with no aggregation.

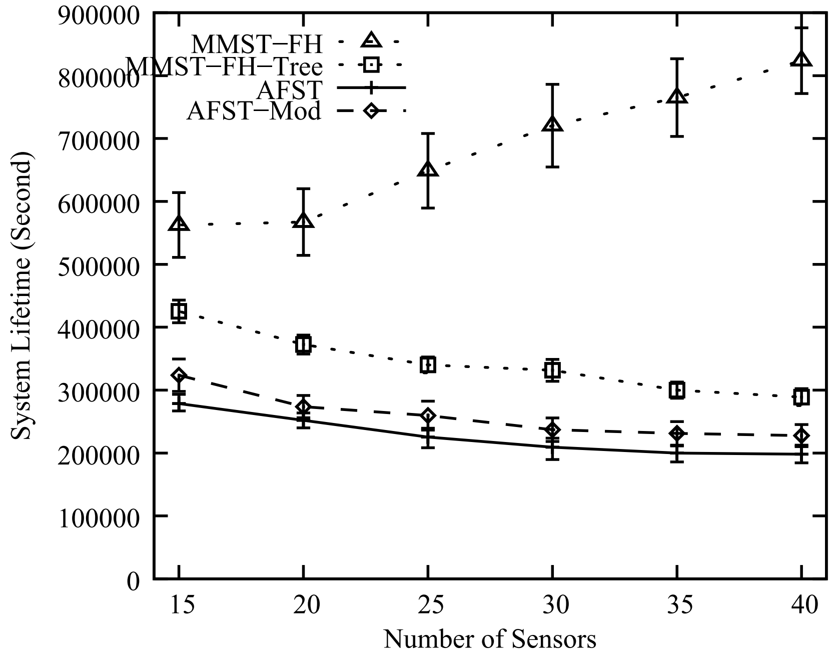
Comparison with AFST

To further explore the performance of our heuristics, we compare with another energy-efficient data gathering algorithm, Adaptive Fusion Steiner Tree (`AFST`) [54]. It provides a near optimal solution to the problem of correlated data collection in terms of minimized energy consumption both on data transmission and aggregation. To build an data collection structure via `AFST`, minimal cost perfect matching is iteratively performed to pair source nodes for aggregation. Within each iteration, nodes are matched based on the edge weight determined by jointly considering energy consumption on data aggregation and transmission. For each non-sink matched pair, an "aggregation benefit" is computed to decide if it is worth performing aggregation. For each aggregation pair, one node is randomly chosen as an aggregation node and the other transmits its data to the aggregation node. After data aggregation, only the aggregation node is left for the next round of matching. For those pairs that do not fuse, both nodes send data to the sink along the shortest path. The process continues until all aggregated data converge at the sink node. In this way, energy can be intelligently spent for data aggregation and routing, therefore extending the operation time of the network. The reason behind choosing `AFST` is two fold. First, we want to compare the lifetime obtained by our algorithms to an aggregation-enabled data gathering scheme. Second, since `AFST` performs data aggregation in an energy efficient fashion, we also use it as a benchmark to evaluate the data aggregation efficiency integrated with our algorithms.

In `AFST`, energy consumption of data aggregation and wireless transmission are assumed to be of a similar level. Data redundancy is assumed to linearly degrade with the distance between two sensors until the distance exceeds a threshold (data correlation range). To fairly compare against `AFST`, we assume data aggregation is energy free. Therefore, for each matched pair of nodes, data aggregation occurs with much higher probability than before. We examine the impact of data correlation to the performance of the algorithms by varying the data correlation range (R_c) from 10m to 20m.



(a) Case 1 ($R_c = 10m$)

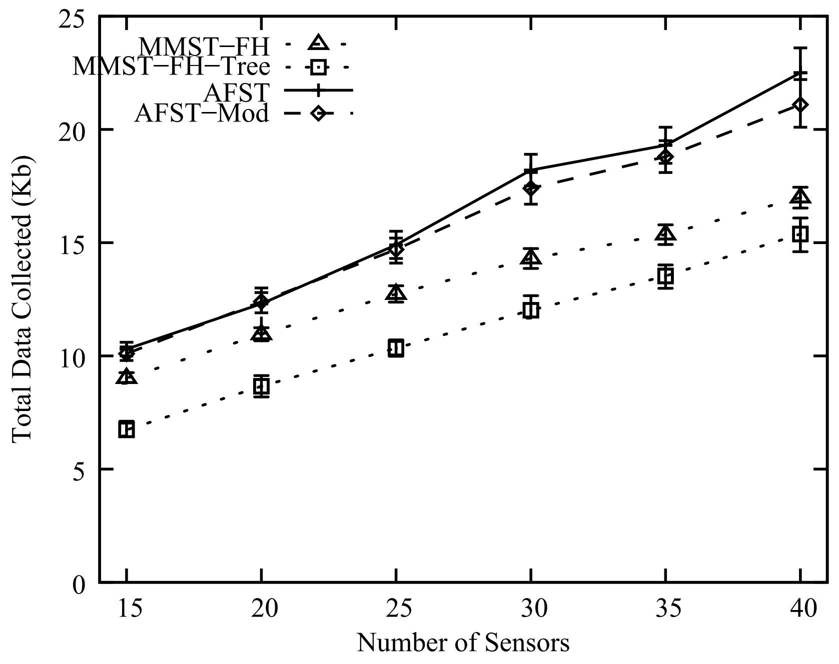


(b) Case 2 ($R_c = 20m$)

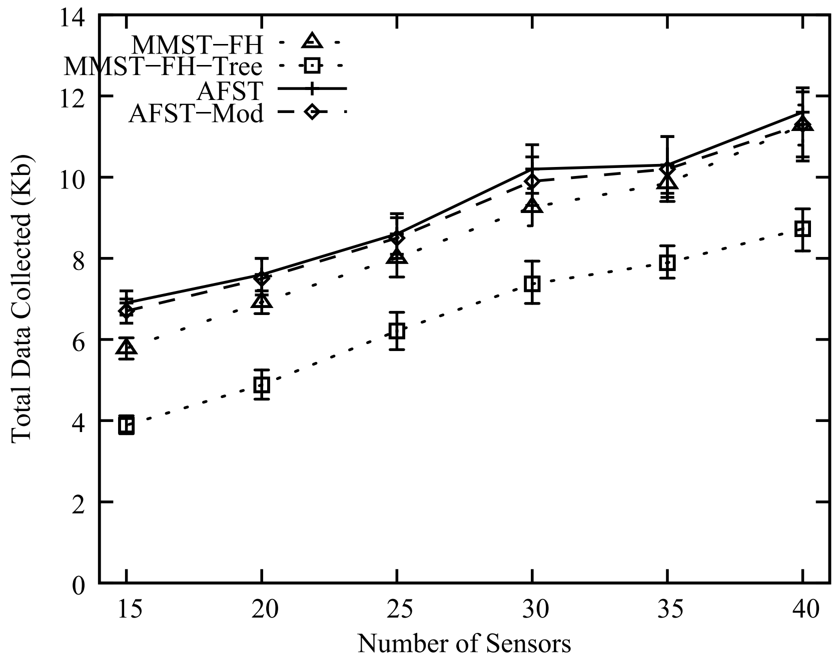
Figure 5.9: System lifetime comparison.

Since system lifetime is the performance metric of interest, we first show the system lifetime achieved by tree-based `MMST` (`MMST-FH-Tree`), flow-based `MMST` (`MMST-FH`) and `AFST` under different parameters in Figure 5.9(a) and 5.9(b). We find that even ignoring the energy consumption on aggregation, the flow splitting data gathering scheme (MMST-FH) outperforms the other tree-based schemes (MMST-FH-Tree and AFST) because MMST-FH balances the load as well as reduces wireless transmission via first hop data aggregation. For the tree-based schemes, MMST-FH-Tree performs better even though AFST is capable of doing further aggregation after FH in a large correlation range (20m). The reason behind this behavior is twofold. First, AFST does not take full advantage of data correlation existing between closely located sensors, as will be discussed later. Second, due to the random choice of data aggregation node for each matched node pair, AFST wastes valuable energy on transmitting data to some nodes which are unfortunately further away from the sink. To remove the effect of the latter problem, we also consider a modified version AFST-Mod that chooses as parent the node that is closer (in hops) to the sink. The results come closer to the lifetime of the MMST-FH-Tree mechanism.

We also show the total data collected at the sink in Figure 5.10(a) and 5.10(b) to analyze the aggregation behavior of MMST-FH, MMST-FH-Tree and AFST. For a N-node WSN, our first hop aggregation schemes (MMST-FH and MMST-FH-Tree) perform aggregation over $N-1$ edges (FH transmission) while AFST only utilizes $N/2$ edges (a side-effect of its matching logic). Since the most significant aggregation exists within the FH transmission and inner node distance will be set apart after first round matching, AFST can not make up the lost aggregation by using a large correlation range to aggregate data further. So, AFST collects the largest amount of data at the sink. In addition, MMST-FH-Tree aggregates data more effectively than MMST-FH.



(a) Case 1 ($R_c = 10m$)



(b) Case 2 ($R_c = 20m$)

Figure 5.10: Total volume of data collected.

5.3 Summary

In this chapter, we combine multi-path (flow-based and load balancing) data delivery and data aggregation together to extend system lifetime. A mixed IP/LP model is utilized to formulate the problem. A flexible and application-specific *flow loss multiplier* is also introduced representing the impact from data aggregation. Due to the computational complexity, we also presented several FH parent selection heuristics to obtain significant redundancy removal. Our simulation results demonstrate that, after appropriate FH data aggregation, the system lifetime can be greatly extended.

There are many other lessons that can be drawn from our simulation results. First, no matter whether we perform FH aggregation or not, splitting the data flow can balance the traffic and result in prolonged lifetime, but it requires extra energy for exchanging local or even global information and leads to a longer data collection delay. Secondly, without data flow splitting, transmitting aggregated data through a data gathering tree will also achieve a fairly satisfactory lifetime. Moreover, a tree-based data gathering scheme simplifies data collection saving energy consumption when it comes to system overhead. If the hardware complexity of the sensor device is low, it is still possible to effectively perform data aggregation at a FH parent node and prolong the system lifetime, because α-FH only requires local information of correlated distance which can be detected based on signal strength. For some applications that require prompt reaction, data collection latency becomes the critical issue. In this type of scenario, MMST-SPT provides a method to balance between extended lifetime and reduced latency in data collection. Finally, when compared with a complex perfect matching–based aggregation scheme (AFST), our FH aggregation scheme is more efficient at obtaining a larger amount of data aggregation in our studied scenarios.

When considering sensor data collection algorithms applied over a non-ideal MAC protocol, the impacts of this practical MAC protocol need to be taken into account during the design of all the tree-based and flow-based schemes presented before. In tree-based schemes, dynamic tree adjustment is provided to deal with non-balanced energy consumption across the network. It can also handle the broken links or links with unacceptable latency, etc. In flow-based schemes, multiple paths are utilized to route sensor data for each pair of source and destination nodes. They are more robust to link failure or similar issues from un-ideal MAC protocols. However, to implement flow-based schemes, a more complicated MAC protocol has to be provided, which may cost much more energy on control packet overhead.

Chapter 6

Contour Map Reconstruction

First hop (FH) aggregation as described earlier in the thesis exploits spatial data correlation that exists between data sources within communication range. We note that spatial correlation is not restricted to contiguous areas. In this chapter we will consider contour maps. Different to the previous data aggregation model, for WSN applications of contour map reconstruction, measurements obtained at sensors spaced far apart can still be strongly correlated (they may locate around the same contour line and possess similar sensor readings). Thus, while typically measurements by one sensor are expected to be highly correlated with those of nearby sensors, there exists potential that they are also highly correlated with measurements of sensors far away, i.e., multiple hops away in terms of routing. It is interesting to develop other aggregation schemes for exploiting data correlation between data sources more than one-hop apart to further eliminate redundant transmissions and extend system lifetime.

Compared with FH data aggregation, it becomes more complicated to intelligently perform data aggregation in WSNs where the most efficient ("best") aggregation nodes for a particular data source may be located several hops away. However, chasing maximum data aggregation by delivering raw sensor data several hops away may not necessarily lead to lifetime extension, because the energy spent on the path (to the aggregation node) could be larger than in FH aggregation. If the "best" aggregation node is not closer to the sink than the source node itself, we may lose the benefit of better aggregation by having to travel longer paths to the sink.

In this chapter, we first investigate the impact of performing greedily data aggregation at the "best" aggregation site regardless where it is. The results of our analysis indicate that, in general, to obtain the maximized lifetime by performing data aggregation at the "best" site requires that the gain out of the resulting data volume reduction has to be extremely high, and in inverse relation to the size of the network. Motivated by this conclusion, we then provide an aggressive data aggregation scheme for a particular type of WSN applications, contour map reconstruction [80, 46, 11, 92, 29], taking advantage of data correlation between more than two source nodes. We use simulations to study its performance.

6.1 The Limits of Pairwise Aggregation Techniques

It is well known that in wireless sensor networks employing multi-hop forwarding, the energy of nodes closer to the sink is depleted first [66]. Solutions that cluster nodes into "neighborhoods" and perform intra–cluster aggregation do not fundamentally change this observation (think of the clusters as becoming "nodes"). Similarly, performing pairwise aggregation between two neighboring nodes is also not changing the nature of the observation (think of each pair becoming the equivalent of a single "node"). An option is to consider performing aggregation not locally, i.e., within a "neighborhood", but globally, at a node, possibly distant, someplace in the network, assuming the most beneficial site for performing aggregation is at that distant node. Such an approach trades additional load to forward to the distant aggregation site, with a potentially better reduction of the traffic that eventually marches from the aggregation site to the sink.

Assume all sensor nodes have the same transmission range, R_t, and are uniformly distributed within a 2-D area with a unique sink located at the origin. Without loss of generality, we assume the area of interest is a sector of angle $\pi/2$ (Figure 6.1). We divide the area into k sections $A_1, A_2, ..A_k$ where the radii of each section A_i is $i * R_t$ $(i = 1, 2, ..., k)$. The area of each section A_i $(i = 1, ..., k)$ is $A_i = \frac{\pi}{4}(2i - 1)R_t^2 = (2i - 1)A_1$. For convenience we set A_1 to be the area unit (A_1=1). The total area of the k sections is $A_T = \sum_{j=1}^{k} A_j = A_1(1+3+5+...+2k-1) = k^2$. Assume now that for each data unit generated at a sensor, there is a *best* aggregation

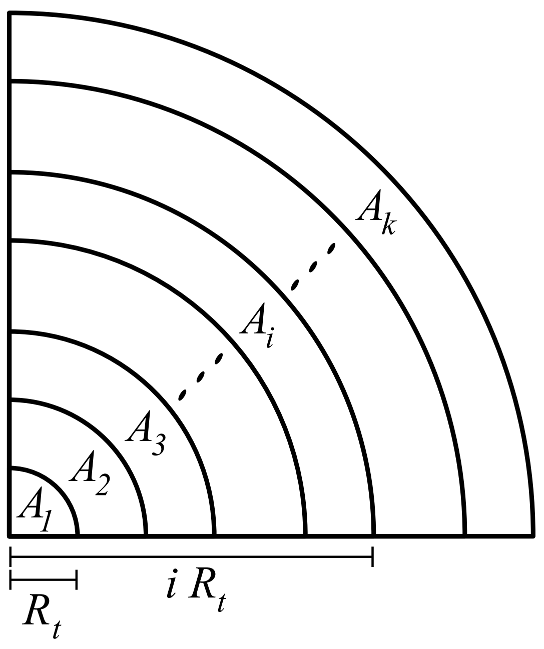


Figure 6.1: Deployment model.

node/site for it and the location of the best aggregation site is uniformly distributed across the whole network. We model the impact of the best aggregation by a factor, m $(0 < m < 1)$, which expresses traffic reduction caused by aggregation. Specifically, for every unit of traffic (before aggregation) from two sources, we produce $1 + m$ units as a result of aggregation instead of 2. We assume, optimistically, that this "best" m applies to the aggregation of traffic from any sensor (realistically, m is not the same for all nodes). In addition, to capture the energy difference between transmitting and receiving, we define β as the ratio of the amount of energy consumption for receiving one bit over that of transmitting a bit. We assume that, typically, $0 < \beta \leq 1$.

Nodes within A_1 do not gain from aggregation because a single transmission delivers their data to the sink. Nodes in A_i $(i > 1)$ impose *energy cost* on nodes in A_1, which depends on where the traffic from the node in A_i gets aggregated. For example, for a node in A_2, if its best aggregation node is in A_1, then the energy consumption it adds to nodes of A_1 is $\beta + m$ (the cost to receive it, β, and to transmit, m, the resulting aggregation outcome). If its best aggregation site is in A_j $(j = 2, 3, ..., k)$ then the cost to nodes in A_1 is $(1 + \beta)m$ (since they receive and relay the already aggregated traffic). Assuming uniform distribution for the location of the best aggregation site, the expected relay load for a node in A_1 is:

$$\begin{aligned} L_1 &= \left(\frac{A_T - A_1}{A_T} m(1+\beta) + \frac{A_1}{A_T}(\beta + m)\right) \frac{(A_T - A_1)}{A_1} \\ &= ((1+\beta)mk^2 + \beta(1-m))(1 - \frac{1}{k^2}) \end{aligned} \tag{6.1}$$

Next we generalize the definition of the energy cost to nodes in A_i ($i > 1$). Besides transmitting their own sensed data[1] they also need to relay "raw" data (data before aggregation) and aggregated data for other sensors. We classify the other nodes with respect to a node in A_i into into 3 categories: nodes in A_1, A_2, ... A_{i-1}; nodes in A_i and in A_{i+1}, ... A_k, denoted respectively by $A_1 \cup I$, II, and III. Within each category the best aggregation site could be in the same or in a different category. If the aggregation site is in A_i, we assume the ideal (optimistic) case that a single transmission delivers the data within A_i and costs the aggregating node β to receive it and m to transmit the resulting aggregated traffic, for a total of $\beta + m$. If the aggregation is performed in (A_1, A_2, ... A_{i-1}), then no node in A_i serves as relay, and the load to nodes in A_i is 0. Finally, if the aggregation is performed in (A_{i+1}, ... A_k) the data, after aggregation, has to travel to the sink and hence it imposes an energy cost $m\beta$ to receive and m to transmit on a node in A_i, for a total of $m(1 + \beta)$. The summary of all the load seen by nodes in A_i for the various categories to which the originator and the aggregator node reside is shown in Figure 6.2. Invoking the uniform distribution of the nodes in the 2-D space we can produce the weighted sum of costs, and we can determine the expected energy cost for a node in A_i:

[1]We will not account for the load of data sourced/sensed at a node, because it is an inescapable cost and it is equal to all sensors throughout the network.

Type	Source in	Aggr. in	Energy Cost	Prob.
I	$A_2, ..., A_{i-1}$	$A_1 \cup I$	0	$\frac{\sum_{j=1}^{i-1} A_j}{A_T}$
		II	$\beta + m$	$\frac{A_i}{A_T}$
		III	$(1+m)(1+\beta)$	$\frac{\sum_{j=i+1}^{k} A_j}{A_T}$
II	A_i	$A_1 \cup I$	0	$\frac{\sum_{j=1}^{i-1} A_j}{A_T}$
		II	$\beta + m$	$\frac{A_i}{A_T}$
		III	$m(1+\beta)$	$\frac{\sum_{j=i+1}^{k} A_j}{A_T}$
III	$A_{i+1}, ..., A_k$	$A_1 \cup I$	$1+\beta$	$\frac{\sum_{j=1}^{i-1} A_j}{A_T}$
		II	$\beta + m$	$\frac{A_i}{A_T}$
		III	$m(1+\beta)$	$\frac{\sum_{j=i+1}^{k} A_j}{A_T}$

Figure 6.2: Relay load vs. aggregation site.

$$\begin{aligned} L_i &= \left(\frac{A_i}{A_T}(\beta+m) + \frac{\sum_{j=i+1}^{k} A_j}{A_T}(1+m)(1+\beta)\right) \frac{\sum_{j=2}^{i-1} A_j}{A_i} \\ &\quad + \left(\frac{A_i}{A_T}(\beta+m) + \frac{\sum_{j=i+1}^{k} A_j}{A_T} m(1+\beta)\right) \\ &\quad + \left(\frac{\sum_{j=1}^{i-1} A_j}{A_T}(1+\beta) + \frac{A_i}{A_T}(\beta+m)\right. \\ &\qquad \left. + \frac{\sum_{j=i+1}^{k} A_j}{A_T} m(1+\beta)\right) \frac{\sum_{j=i+1}^{k} A_j}{A_i} \qquad (6.2) \end{aligned}$$

Clearly, the closer to the sink, the larger the energy cost. To achieve maximum system lifetime nodes close to the sink should not deplete their energy any faster than any other nodes that rely on them for relaying their traffic, and since only nodes in A_2 can directly feed the nodes in A_1, the following must hold: $L_1 = L_2$. Solving $L_1 = L_2$ for m, we get:

$$m = \frac{(1+\beta)k^2 - (4+4\beta)}{(2+4\beta)k^4 - (1+9\beta)k^2 - (1+\beta)} \qquad (k \geq 2) \qquad (6.3)$$

Hence, pairwise aggregation, performed at the best possible site that would result in the most reduction of data, is *not scalable*, because for increasing k (i.e., network scale expressed as multiples of R_t) and assuming uniform sensor deployment density, the cost is justified only for an extremely aggressive data aggregation

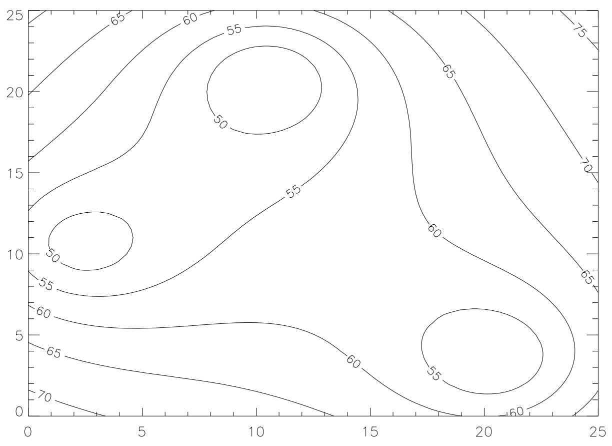


Figure 6.3: Contour map example.

scheme (tiny m), and m must be decreasing at least as fast as k^{-2}.

6.2 Contour Map Aggregation

Given the limitations of pairwise aggregation, we opt for an aggregation approach which allows, in principle, aggressive n-way ($n \geq 3$) aggregation. We also consider how clusters could be "stringed" to facilitate further aggregation, i.e., without restricting the application of aggregation to within individual "neighborhoods". To do so we define aggregation applicable to contour maps. We assume the reader is already familiar with the data representation form of a contour map, which is composed of "contour lines" (also called "isolines"), an example of which is shown in Figure 6.3. The points and, by extension, the sensors residing between two successive contour lines are said to belong to the same level set. Contour maps are a widely utilized data representation method [80, 46, 11, 92, 29]. Between two successive contour lines, the attribute values are considered to be the same or similar, hence a contour map introduces an *approximation* which is bounded by the step size between successive contour lines.

Contour map construction from sensor data has been already addressed in [46, 47], but whereas the interest in previous studies was toward discerning contour lines in the presence of noise, we are interested in the interaction between routing and aggregation for contour map data with the intent to determining the most energy efficient strategy. In summary, what we try to exploit is that given a step value (or "tolerance") that separates contour lines, sensors of the same level set could report to the sink that they possess the exact same measurement. We assume that the position of the sensors is known, and hence, based on their positions and the attribute value they represent, the sink can reconstruct the complete contour map.

The reason for applying our devised approaches to contour data collection are two fold: (1) Contour data extraction and contour map reconstruction (at the sink) provides a simple but very efficient approach to obtain the global view of the entire sensor field regarding the phenomenon of interest. A lot of environmental phenomena can be well analyzed and understood by a spatial summary of an attribute over the region of interest, for instance, the distribution of temperature, humidity, radiation or pollutant levels. Example applications include event detecting [92, 82] and environmental monitoring [63, 29]. (2) The space between two successive contour lines represents points where the same attribute readings are observed. With a given tolerance, sensors within two successive contour lines can report to the sink that they possess the same measurement. The contour map reconstructed at the sink from the data sent from the sensors possesses enough fidelity to the original data. With the dedicated contour data aggregation techniques which will be discussed later on, it is possible to get a larger volume of redundancy removed between nodes many hops away.

6.2.1 Network Architecture

We use a hierarchical architecture for aggregation purposes. Nodes are grouped into clusters, each with a cluster-head (CH) placed in charge of aggregating data of all its members (plus itself) and with sending the aggregated data to the sink, relaying the aggregated data through other CHs. As we will see, after intra–cluster aggregation, there is potential for further, inter–cluster, aggregation to be performed as well. Because of the burden to CHs, we assume a similar approach as in [34], thus "rotating" the role of CH, to even out the energy depletion within clusters. Namely, each sensor node (i) selects itself as a cluster-head (CH) with a probability

$P_{ch}(i)$, which is between 0 and 1, and is set as:

$$P_{ch}(i) \quad = \begin{cases} \frac{p_{ch}}{1-p_{ch}*(r\ mod\ \frac{1}{p_{ch}})} & i \in G \\ 0 & otherwise \end{cases}$$

where r denotes the number of the current cluster formation round, p_{ch} is the desired percentage of CHs, and G is the set of nodes that have not been CHs in the last $(r\ mod\ \frac{1}{p_{ch}})$ cluster formation rounds. Cluster formation is triggered periodically. During round 0, every sensor has the same probability p_{ch} of becoming a CH. After that, each round each node has a chance to be CH except those nodes who have already been CHs in the last $(r\ mod\ \frac{1}{p_{ch}})$ rounds. We mark those nodes (take them out of G) who used to be CHs, to avoid making them CHs again. After $\frac{1}{p_{ch}}$ cluster formation rounds, all the marks are cleared and every node has the same probability (p_{ch}) to select itself as a CH again. In this way, the heavy workload of being a CH can be distributed across the entire network. The interval between two successive cluster formations is determined by the data accuracy requirement from the given application. This is because the data reporting is part of the cluster formation process.

Once a node becomes a CH, it broadcasts an advertisement message announcing its status of CH as well as its current sensor reading. Based on a given contour step-value (which we assume is globally known) and the received CH advertisement messages, each non-CH node selects as its cluster-head a CH whose sensor reading is within the same range ($[n * step, (n + 1) * step]$) of its own reading. That is clusters are formed from nodes that belong to the *same level set* and which are, of course, within communication range of each other. If there are two or more such CHs within its transmission range, the node selects the CH which is closer to it in terms of Euclidean distance. A node may fail to join any cluster because its sensor reading is not within the same range of any neighboring CH. In this case, it will elect itself the CH of a single-node cluster. After cluster construction, each cluster member reports its reading to the sink using the shortest path from the CH to the sink. Figure 6.4 shows an example of the resulting logical topology.

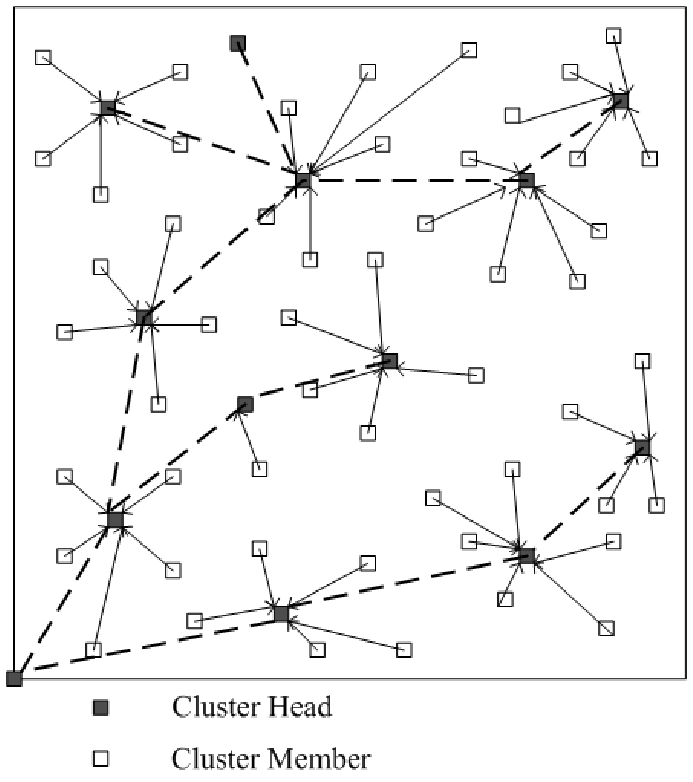


Figure 6.4: Cluster-based hierarchical framework for contour data collection.

6.2.2 Intra–Cluster Aggregation

Intra–cluster aggregation is performed in two steps:

Reading Suppression

We assume sensor locations are known. Each data unit is composed of the sensor that produced it and the sensed attribute value. The objective is to collect the readings from nodes belonging to the same level set, i.e., within the same value range ($[n * step, (n+1) * step]$). These readings can then be suppressed except for sending out a representative one. To do so, we exploit the cluster structure. By construction, all the sensor readings within a cluster are from the same level set. The aggregation performed by the CH is to report only its own value and the IDs of all its cluster members (plus its own), thus essentially approximating the data sensed at the cluster members with its own. For example, in Figure 6.5, the CH, d, which has sensed value 58, will aggregate the cluster's measurements as $< 58, C_d >$ where $C_d = \{a, c, d, e\}$.

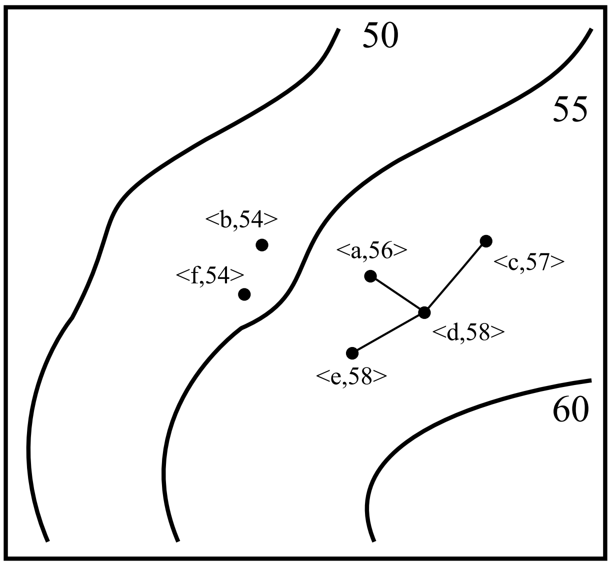


Figure 6.5: Reading suppression.

Subset Construction

The readings can be further aggregated by choosing from the reporting set C_i a *representative* subset $C_i' \in C_i$. The selection of C_i' can be performed by making sure that all nodes in C_i are at most ϵ distance away from the set of nodes in C_i'. This is the Dominating Set (DS) [36] problem on the graph built with the vertices of C_i and with edges between any two vertices whose distance are ϵ or less. That is, we seek the DS of the graph $G(C_i, E(\epsilon))$ where $E(\epsilon) = \{(x, y)|d_{x,y} \leq \epsilon, x \neq y, x, y \in C_i\}$, and $d_{x,y}$ denotes the Euclidean distance between node x and y. Because the minimum cardinality DS is NP–Hard we employ a greedy approximation [36] which can be executed easily and locally, at the CH. The final outcome of intra–cluster aggregation is $< s_i, C_i' \cup \{i\} >$, which includes the CH i (if not already in the DS), as it is the CH's value s_i that represents all the sensors in the cluster. Figure 6.6 provides an example where all nine nodes are within range of each other and form a cluster with CH the node a. The intra–cluster aggregation produces $< s_a, \{a, c, d, f, h\} >$. Note that edges in Figure 6.6 represent the set $E(\epsilon)$.

Note that, based on the $C_i' \cup \{i\}$ received from each CH, and because the sink is aware of ϵ and the locations of all nodes, the sink can determine which nodes are dominated by each C_i' assuming that nodes that belong to different level sets are at

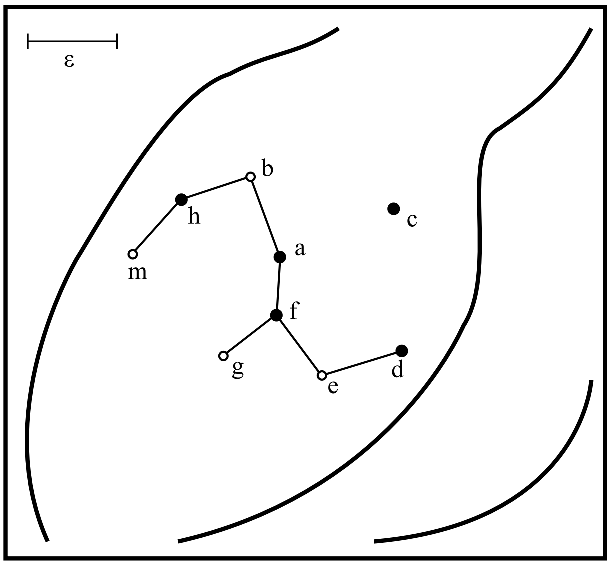


Figure 6.6: Subset construction.

least ϵ units away. If they are any closer, then the subset construction essentially introduces a spatial approximation on top of the value approximation that resulted from the reading suppression step. Moreover, if two CH, say i and g, formed by nodes that are on the *same* level set, could coordinate their subset construction, they could do so by determining the DS of their *joint* graph, i.e., the union of their two ϵ–based graphs, plus edges of distance ϵ or less between vertices of the two clusters. Naturally, the construction of the DS of the combined graph makes sense when the clusters are close to each other, i.e., they have at least two nodes at a distance ϵ or less. We exploit this possibility in the second part, that of inter–cluster aggregation.

6.2.3 Inter–Cluster Aggregation

In inter–cluster aggregation the task is to determine pairs of clusters, such that their combined aggregated data can be aggregated further. In order for this to happen, the two clusters must be formed by nodes that observed measurements that fall within the same level set. In addition, if the clusters are close to each other, then the subset construction outlined in the previous paragraph can also be applied, but its impact is expected to be, in general, rather limited. Hence, for every two clusters, one can define the *potential* of the aggregation outcome numerically. Namely, in order to evaluate the amount of data reduction between a pair of CHs, we define a *joint flow loss multiplier* (m_{xy}) between CHs x and y, as the data reduction is possible if the two node sets of the two CHs are combined and collectively represented by the dominating set of the graph defined by their union. (For well–separated clusters, this will be the union of their DS.) Thus, if the cluster with CH i sends its aggregated data $< s_i, C_i >$ to CH g which possesses the aggregated data $< s_g, C_g >$, and if D_i represents the "size" of the aggregated data from i, i.e., $D_i = | < s_i, C_i > |$ then the post–aggregation data at g will be of size $D'g = (D_g + D_i) * m_{ig}$.

At this point, the problem becomes an optimization problem: which clusters to pair so that their aggregate traffic is the least possible. Furthermore, after aggregating the traffic of two clusters, we can split it to multiple flows and forward it to the sink, thus attaining a degree of load balance. Essentially, the problem becomes one that we have already discussed in Chapter 5 where first the aggregation is determined and then the resulting aggregated data are split using a flow model. To reduce the computational complexity for solving the flow problem and make our approach scalable, we consider CHs as nodes representing the collected aggregated data. The particular flow fragments and the paths taken to the sink can be computed via our IP/LP model outlined in Chapter 5. However, we first need to determine the pairs of clusters, and to do so we employ a variety of heuristics:

Maximum Aggregation Perfect Matching (MAPM)

MAPM attempts to maximize the total inter–cluster data reduction across the entire network. The joint flow loss multiplier is utilized to measure the benefit from inter-cluster aggregation over each pair of CHs. We use $(D_i + D_g) * (1 - m_{ig})$ to denote the total joint volume reduction via aggregation and place it as edge weight on a graph including as vertices only the CHs. We apply maximum aggregation perfect matching [51] across the network. After perfect matching and inter-cluster aggregation, each pair of matched CHs (i and g) is treated as a single data flow whose data demand is $(D_i + D_g) * m_{ig}$ generated at g. If the number of clusters is odd then we are left with single CH that does not employ inter–cluster aggregation.

Local Maximum Aggregation Perfect Matching (`Local MAPM`)

`Local MAPM` is a variation of `MAPM` where pairs have to be neighbors, on the assumption that traveling a longer path to a better aggregation opportunity is expensive in terms of energy. Figure 6.7 and 6.8 denote the paired CHs obtained via `MAPM` and `Local MAPM` over the same sensor location. In these two figures, each node denotes a CH. In Figure 6.8, the two CHs of each matched pair are only one hop away from each other, while in Figure 6.7, the distance between the two CHs of each matched pair may be more than 1 hop apart.

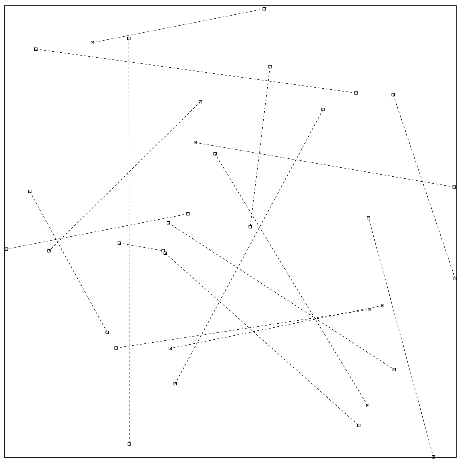


Figure 6.7: Result of inter-cluster `MAPM`

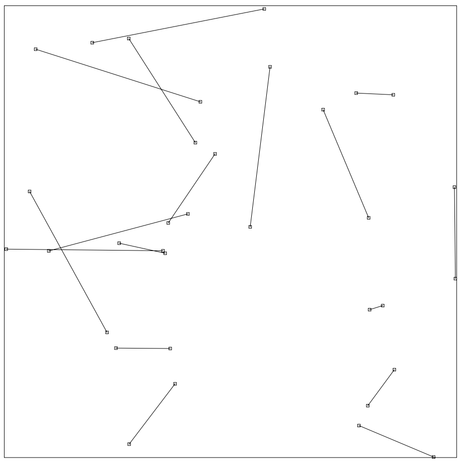


Figure 6.8: Result of inter-cluster `Local MAPM`

ENergy Critical node Aware Spanning Tree (`ENCAST`)

`ENCAST` has been introduced in detail in Chapter 4, and contrary to the previous two schemes it does not split aggregated data into flows. Instead it builds a single tree. The tree is reconstructed, i.e., a new "version" of the tree is built, as nodes become critically low in terms of energy. Critical nodes become leaf nodes of the tree. A parameter, α, is used to trigger tree reconstructions depending on the energy depletion situation in the network. Specifically reconstruction occurs when the residual energy of any node drops below the threshold given by $\alpha^k E$ where E is the initial energy and $k = 1, 2, ...$ indicates the "version" of the tree. Thus the longer the network operates (and the more times the tree is reconstructed) the lower the threshold is pushed. Because the routing to the sink is restricted to the tree, the inter–cluster aggregation is not performed as in the previous two schemes, but progressively as the aggregated data travel to the sink from one CH to the next, along the tree. When the aggregated data arrive at a CH of a cluster from the same level set, they are aggregated and sent out. This form of aggregation can result in more than two CHs aggregating their data together.

Finally, we also consider a benchmark scheme (we will call it `General`) which represents intra–cluster aggregation *alone* without any inter–cluster aggregation performed. Subsequent to intra–cluster aggregation, the data are split into multiple flows and routed to the sink as per the solution of an LP flow optimization.

6.3 Performance Evaluation

We simulate a sensor network with anywhere from 60 to 140 nodes placed in a $25 \times 25m^2$. The location of the 140 nodes is shown as dots in Figure 6.9. The sink is placed at the origin (0,0). Nodes start with energy of 10 Joules and each of them generates sensor data at a constant rate of 1 Kb/sec. We adopt the energy consumption model of [26]. Energy spent on receiving and sending data packet is determined by the wireless communication environment and the packet size. We assume each sensor node's wireless transmission range is 10 m. The contour data are dynamically generated by the diffusion model presented in [63] which ensures the

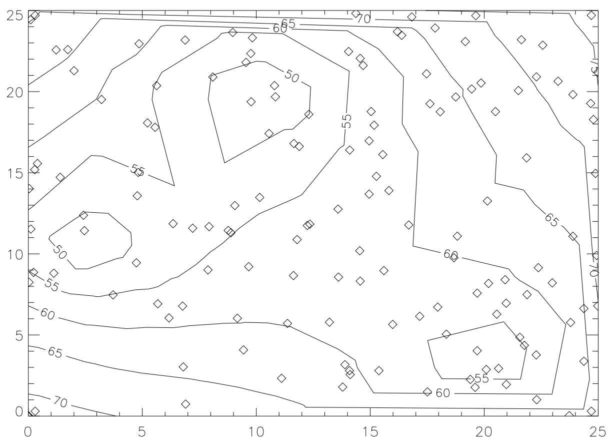


Figure 6.9: Without aggregation.

slow but continuous change of the contours. To show the quality of the contour data collected (using 140 nodes) we present reconstructed contour maps (Figures 6.9 to 6.12) the way the sink would generated them. Namely, at the sink, the per-sensor data are reconstructed from the received aggregated data, interpolated, smoothed, and presented as a contour map using a data visualization tool [1]. The step-value is 5 and $\epsilon = 2m$. The major parameters set for the simulations are listed in Table 6.1.

Table 6.1: Simulation Parameters

Parameter	Value
Unit of lifetime	second
D (flow data demand)	$1Kb/second$
E (initial energy)	$10J$
size of area of interest	$25 * 25(m^2)$
grid network	$51 * 51 sensors with 0.5 m apart$
R_t (transmitting range)	10m
ϵ (Euclidian distance tolerance)	2m
Step-value	5
e^t	100nJ/bit
e^r	30nJ/bit
α	0.5 and 0.66

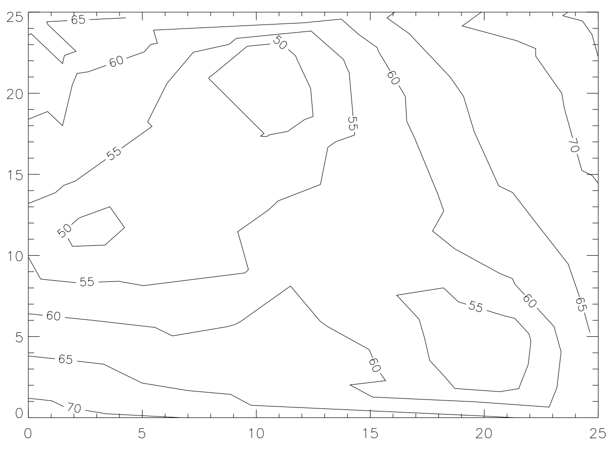


Figure 6.10: Intra–cluster aggregation only.

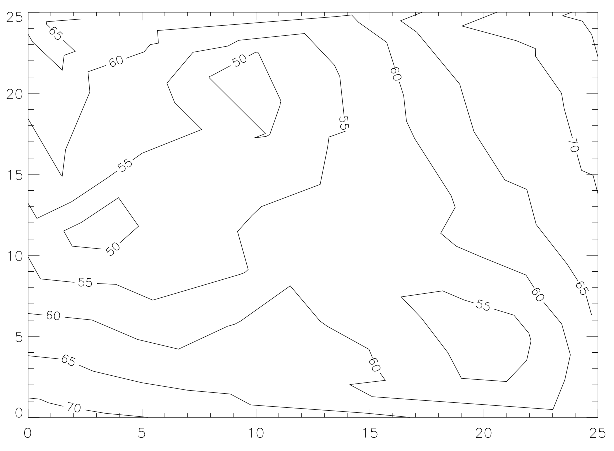


Figure 6.11: Inter–cluster aggregation.

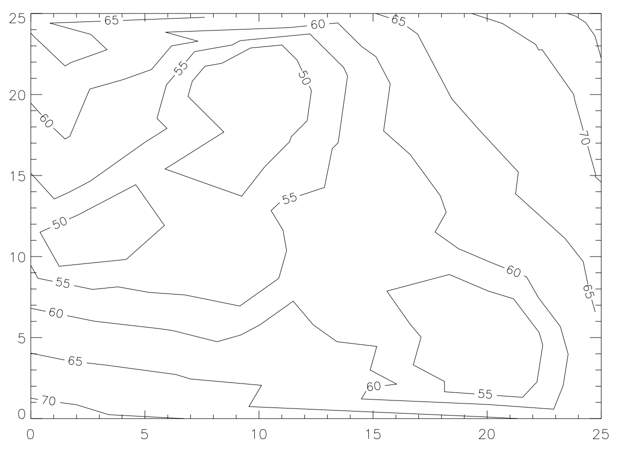


Figure 6.12: Inter–cluster ENCAST ($\alpha = 0.5$).

6.3.1 Contour Map Example

First we look into the quality of the contour map approximation. Comparing Figure 6.9 to the benchmark of Figure 6.3, the degradation mainly comes from less deployed sensors and random sensor placement (meaning poor sensing coverage and coarse granularity). However, from Figure 6.10 and 6.11, we observe that our aggregation schemes are visually similar to the non-aggregation schemes of Figure 6.9 despite their contour lines are not as smooth as the ones in Figure 6.9 because of the data approximation introduced by aggregation. Compared to other reconstructed contour maps, Figure 6.12 shows more accuracy loss because in ENCAST, aggregation is potentially performed at each node on the path to the sink instead of just pairwise.

From the Figures 6.9-6.12, we observe that random sensor node locations and sensor data aggregation degrade the quality of the reconstructed contour map. However, with enough sensors deployed and intelligent data aggregation schemes, a large amount of wireless traffic can be reduced and the reconstructed contour maps remain sufficient data accuracy.

6.3.2 System Lifetime

In our experiments, we generate dynamic contour data via periodically changing the contour map. To benefit from data aggregation as much as possible, the data collection structure for each algorithm has to be reconstructed/adjusted correspondingly, thus introducing energy consumption for transmitting control messages. The contour data obtained from the area of interest may evolve very slowly, thus the current data collection structure can be utilized for a long time without losing large data aggregation opportunities. If this is the case, energy consumption for building data collection structures is one time cost and negligible.

In order to balance unevenly distributed traffic load, `ENCAST` adjusts its data collection tree in a proactive fashion. The cluster head whose residual energy is below the predetermined threshold (*energy critical*) triggers a tree reconstruction which reassigns the children of the current energy critical CH to other CHs. `ENCAST` with two different tree adjustment triggering thresholds ($\alpha = 0.5\, and\, 0.66$) are implemented in our simulations. The larger α indicates more frequent tree adjustment and more balanced workload distribution. In order to demonstrate the benefit from the tree reconstruction, we also show the results for `ENCAST` without any tree adjustments, which uses only one tree (the initial) throughout the lifetime of the network.

The network lifetime results are shown in Figure 6.13. `General` lacks inter–cluster aggregation and performs the second worst among all the schemes. From Figure 6.13, we also observe flow-based schemes with data aggregation outperform other heuristics. `Local MAPM` obtains the longest lifetime because it benefits from balanced traffic load via flow-based traffic delivery and reduced traffic volume via the perfect matching. In addition, the local perfect matching limits the energy consumed on transmitting raw data from the source node to its aggregation site which may be far away from the source. This is also the reason the lifetime achieved by `Local MAPM` is longer than `MAPM`. `MAPM` performs perfect matching without any constraints. It reduces larger traffic volume than `Local MAPM` with more energy consumed along potentially longer paths.

`ENCAST` with various α's attains lifetime similar to the flow-based schemes

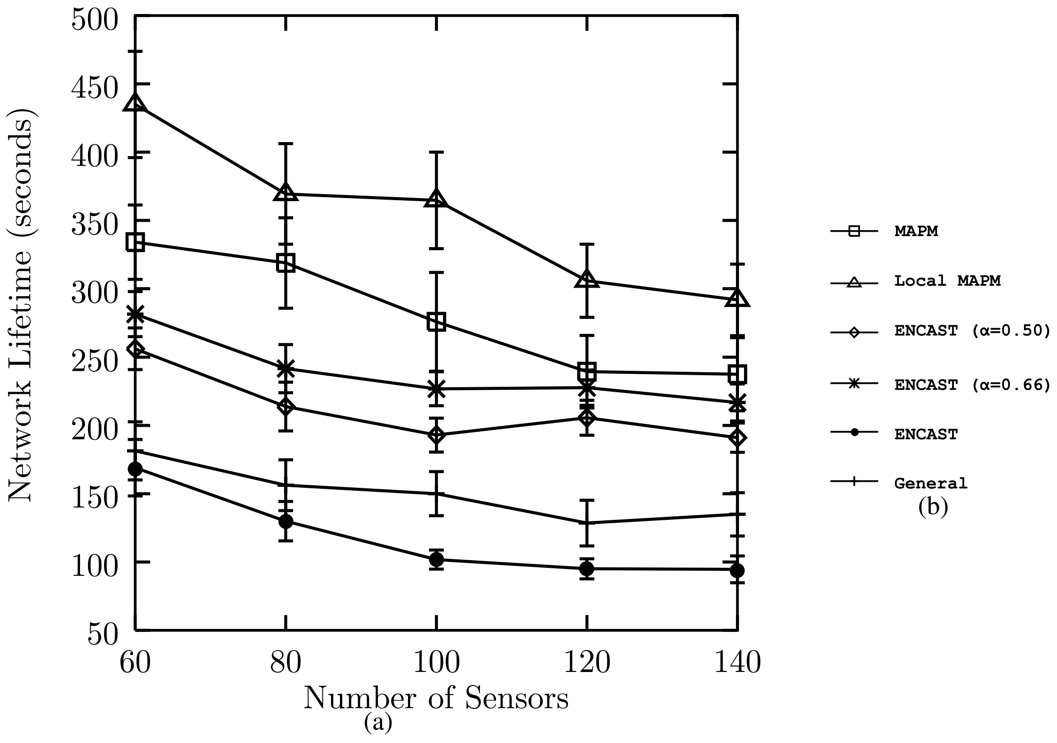


Figure 6.13: System lifetime.

despite not splitting the traffic flow. Tree reconstruction removes the heavy burden of data routing from energy critical CHs, essentially balancing energy consumption among all the CHs. Due to data aggregation, `ENCAST` $(\alpha = 0.66)$ and `ENCAST` $(\alpha = 0.5)$ perform better than `General` in terms of lifetime. Among all the schemes, `ENCAST` without tree adjustment achieves the minimum lifetime. This confirms that `ENCAST` with tree adjustment is a very effective technique to extend system lifetime. We also show the total data volume collected by each heuristic in Figure 6.14. Aggressive on-tree aggregation, `ENCAST` $(\alpha = 0.66)$ and `ENCAST` $(\alpha = 0.5)$ reduces the volume of data delivered to the sink compared to flow–based approaches.

Since `ENCAST` starts with a hop-based shortest path tree the tree reconstruction tends to increase the depth of the tree and therefore data traveling from its CH to the sink have more chances to be aggregated at another CH along the way. The collected data volume decreases (worsens) as α increases. `MAPM` and `Local MAPM` perform perfect matching to pair CHs for aggregation and gather similar amount

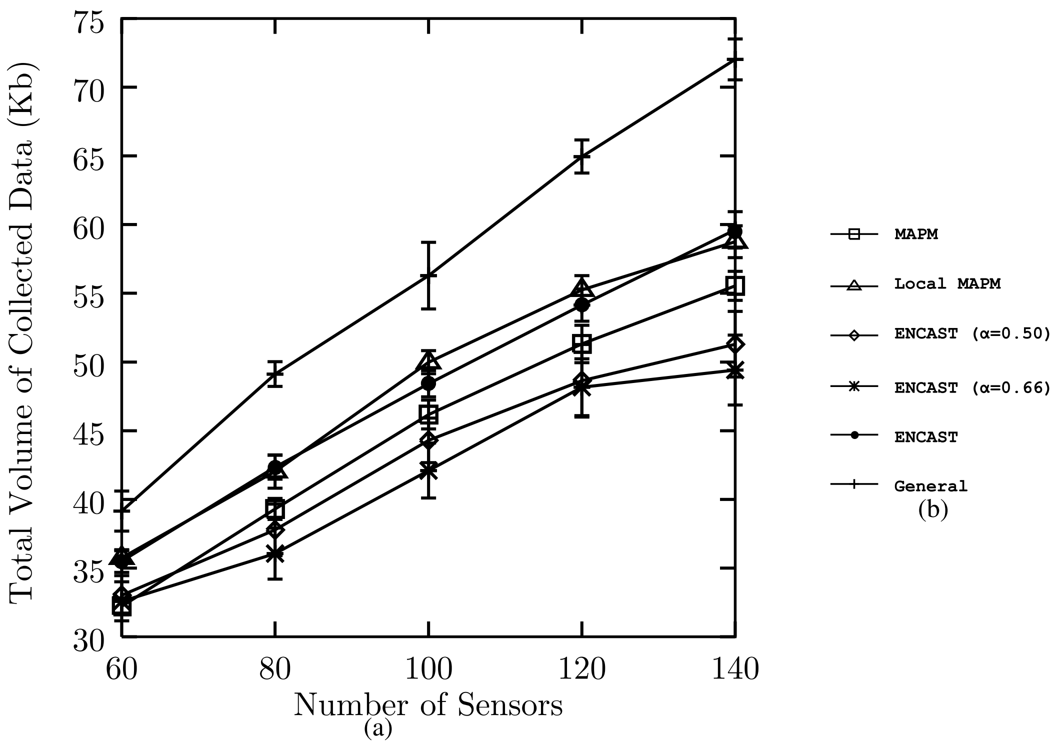


Figure 6.14: Data collected at the sink.

of data to ENCAST's with tree adjustment ($\alpha > 0$). Naturally, the total amount of collected data increases as the sensor density increases.

6.3.3 Contour Discrepancy

To evaluate the collected data accuracy, we introduce another metric similar to [81]: *contour discrepancy*. We examine the reconstructed (at the sink) data, s'_i, for each sensor, i, compared to the actual data, s_i, and formulate the average contour discrepancy as: $(\sum_{i=1}^{N} |s'_i - s_i|)/N$. Data from certain sensors that were eliminated during the aggregation are replaced by the data values from sensors that dominate them. If the missing sensor's value is dominated by more than one node sensor, then the value is chosen randomly from one of the dominators.

We present contour discrepancy for different aggregation strategies in Figure 6.15 where the units on the Y-axis are the same as those of the step-value. General does not apply inter–cluster aggregation, so it maintains the largest data accuracy. MAPM performs a more aggressive aggregation than Local MAPM and results in

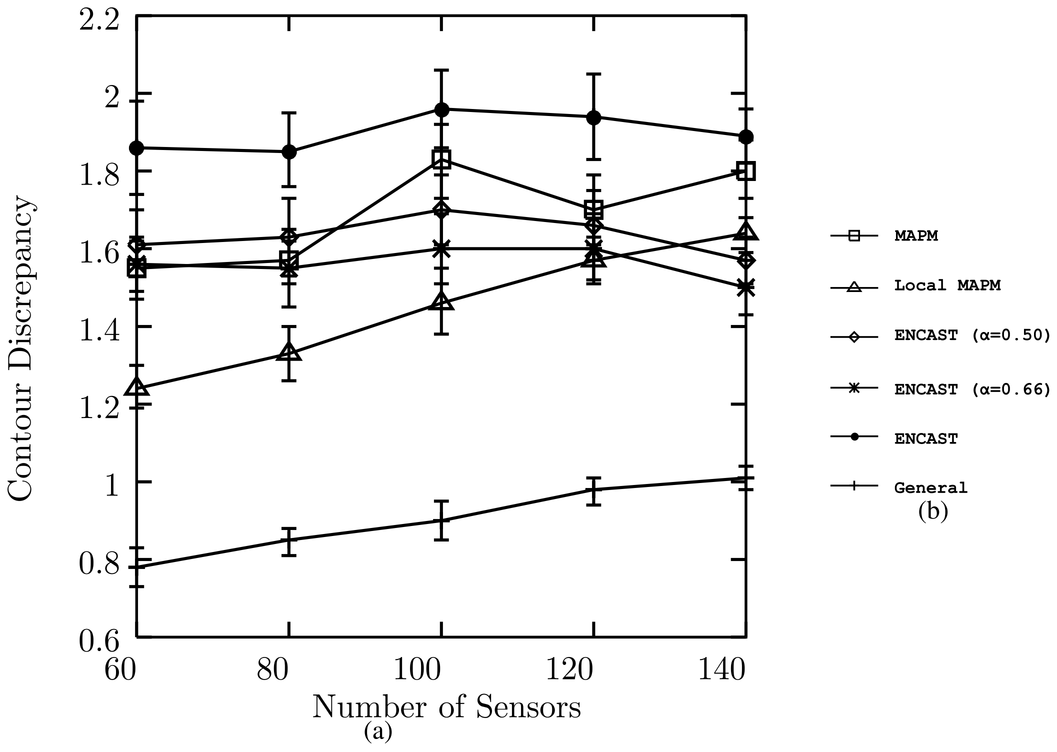


Figure 6.15: Contour discrepancy.

more accuracy loss. For tree–based schemes, they all perform aggressive data aggregation and have large contour discrepancy. However, we observe that contour data discrepancies for all aggregation schemes are below 2 or about 2, which is far less than the step-value of 5.

6.4 Summary

We reviewed the limitations inherent in aggregation techniques where data is routed to the best sensor in terms of aggregation potential. We first established that there are limited opportunities for efficiently aggregating if data aggregation needs to be performed far from the originating sensor. Instead, we argue that in many applications, the presentation of data as a contour map might be sufficient, and that such a model opens the possibility to apply n-way ($n \geq 3$) data suppression and contour vector reduction techniques. This, in turn, helps us construct simple cluster-based routing heuristics. We find evidence that the combination of contour-oriented aggregation models, with the simple cluster-based routing heuristics, leads to scalable data collection.

There are many lessons that can be drawn from our simulation results. First, both the flow-based data collecting schemes and the tree-based `ENCAST`'s (with tree adjustment) extended system lifetime by balancing traffic workload. The other lesson is that, flow-based schemes (with inter–cluster aggregation) outperform other heuristics in terms of lifetime. `ENCAST` benefits both from aggressive data aggregation and the tree's reconstruction, therefore obtaining similar lifetime to flow-based approaches.

Chapter 7

Conclusions & Future Work

7.1 Conclusions

The research detailed in this thesis addressed the topic of maximizing system lifetime in resources constrained wireless sensor networks. Focus has been placed on the development of different data collection algorithms and data aggregation strategies by which energy can be saved on transmitting less data volume. During the process of data collection, traffic load balancing is also considered to avoid exhausting some sensors' energy which usually causes the early termination of the system's functionality.

An extensive literature review of sensor data routing schemes, as well as data aggregation techniques applied during data collection, have been provided. Three widely applied data collection structures are reviewed. Tree-based data collecting infrastructures are easy to construct and maintain because sensor data of each source node are only routed upstream along a single path to the sink (root) node. During the process of data collection, there is no flow splitting for each data along its path, therefore data aggregation can be performed at any intermediate node along the path. Since all the data (from different sources) have to be gathered at the sink, large amount of data will converge at the nodes close to the sink. This inherent uneven traffic load in tree-based structures induces unbalanced energy consumption across the network and early network lifetime termination. Flow-based approaches are proposed to split a data flow (originating at an individual sensor) into several fragments and route each of them along a different path to the sink. Although workload balancing is well handled, data flow splitting destroys the integrity of the data and make data aggregation challenging. Cluster-based data collection protocols have also been intensively studied. They provide scalability and the opportunity to perform many-to-one aggregation at cluster heads over the data from cluster members.

Inspired by the aggregation strategies reviewed, a simple proximity based data aggregation model is raised and utilized to examine the impact of data aggregation on different tree-based collecting schemes. To address the unbalanced workload in tree-based schemes, a proactive tree adjusting algorithm has been designed to release traffic load from some node which relays large volume of data for its children. Simulation results show that with intelligent adjustment triggering threshold, this method significantly extends the network lifetime with small system overhead.

In order to reduce wireless transmissions, data aggregation has been also introduced into flow-based collection frameworks. Data flow is delivered in its integrity to one of its first hop neighbors where data aggregation is applied, then the reduced data flow is free to be spread in order to increase traffic load balancing. To achieve maximum network lifetime and find optimal aggregation nodes, a mixed integer/linear programming model is utilized to formulate the problem. Due to the computational complexity, heuristics are proposed to obtain significant aggregation and accordingly prolong system lifetime. Simulation results show that flow-based mechanisms provide longer network lifetime than tree-based but at the cost of extra complexity and overhead. In addition, our first hop aggregation approaches significantly increase network lifetime for both flow-based and tree-based routing schemes.

The limitations inherent in pairwise data aggregation has been studied. Transmitting data to and performing data aggregation at the most correlated source far away the originating sensor node shows statically limited opportunities in obtaining efficiency in terms of maximizing lifetime. More aggressive n-way aggregation techniques are proposed for contour map reconstruction. Simple cluster-based routing heuristics are also proposed. Simulation results show that our schemes collect largely reduced volume of contour data at the sink and retain reasonable data accuracy as well.

7.2 Future Work

Although this thesis has successfully addressed the challenges for extending system lifetime in WSNs, several questions remain. This section will present a few directions for further research.

As we discussed before, some WSNs are based on complete aggregation, for instance, applying query-based aggregate operators (`min, max, avg`) over sensor readings in [57, 58]. In this case, the data volume to be transmitted by a single sensor node does not increase with the number of other nodes which require the current one relay data for them, because relay data does not accumulate along the path to the sink. However, traffic load imbalance still occurs but not necessarily near the sink node. Nodes with large degree (large number of neighbors) may receive raw data packets from them and run out of energy faster than other small degree nodes. Our algorithms still can be applied for this case but with a simpler version, for example, the tree adjusting strategy `ENCAST` only need assign a critical node's one-hop children to other nodes for data relaying.

In this thesis, we focus on data routing and aggregating schemes by assuming ideal MAC layer. However, sensor nodes and links are prone to failure due to harsh environment where the wireless sensor network is deployed. Once considering the sensor mobility, the network topology becomes even more dynamic. Additional strategies for detecting and recovering node/link failures have to be provide especially to our static tree-based data collecting schemes to handle these unpredictable topology changes. Otherwise, a link may cause a large amount of sensor data missing during transmission or even terminate the operation of entire network.

As the concrete contour map reconstruction example shown in Chapter 6, data aggregation usually results in data accuracy loss. Flow loss multiplier is mainly denoted as a function of Euclidean distance between correlated sensor nodes in our previous simulations. However, it can also be determined by other features of the application or an externally given parameter that quantifies the maximum data accuracy loss. Since we abstract data aggregation in different WSN data collecting scenarios as flow loss multipliers, all the algorithms we provide can operate on flow loss multipliers serving for different WSN applications.

For the same number of sensors within the same area of interest, deploying more than one sinks for data collection is an promising approach to release heavy traffic load on a single sink and its direct neighbors, and prolong the system lifetime. However, solid research work has to be done for determining the number and locations of these sink nodes in order to optimize the overall system performance. Another issue that needs to be carefully considered, regarding multi-sink data collection, is data synchronization among multiple sink nodes. To ensure data consistency across all the sinks, efficient communication strategies are required for sharing data among the sinks. Otherwise, the sinks cannot cooperate to finish a unique monitoring task.

Bibliography

[1] http://www.ittvis.com/productservices/idl.aspx.

[2] Khaled Arisha, Moustafa Youssef, and Mohamed Younis. Energy-aware tdma-based mac for snesor networks. In *Proceedings of the IEEE Integrated Management of Power Aware Communications, Computing and Networking(IMPACCT)*, New York City, New York, USA, May 2002.

[3] Franz Aurenhammer. Voronoi diagrams — a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, 1991.

[4] Baruch Awerbuch and Robert G. Gallager. A new distributed algorithm to find breadth first search trees. *IEEE Transactions on Information Theory*, 33(3):315–322, May 1987.

[5] A. P. Azad and A. Chockalingam. Bounds on the lifetime of wireless sensor networks employing multiple mobile data sinks. In *Proceedings of the 49th annual IEEE Global Telecommunications Conference (GLOBECOM'06)*, San Francisco, California, USA, 2006.

[6] Seung Jun Baek and Gustavo de Veciana. A scalable model for energy load balancing in large-scale sensor networks. In *Proceedings of 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt'06)*, Boston, Massachusetts, USA, 2006.

[7] Seema Bandyopadhyay and Edward J. Coyle. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *Proceedings of the 22nd Conference of the IEEE Communications Society (INFOCOM'03)*, San Francisco, CA, USA, 2003.

[8] Kenneth Barr and Krste Asanovic. Energy aware lossless data compression. In *Proceedings of The First International Conference on Mobile Systems, Applications, and Services (MOBISYS'03)*, San Francisco, California, USA, May 2003.

[9] Dimitri Bertsekas and Robert Gallager. *Data Networks*. Prentice-Hall, Inc., 1992.

[10] Edoado S. Biagioni and K.W.Bridges. The application of remote sensor technology to assist the recovery of rare and endangered species. *International Journal of High Performance Computing Applications*, 16(3):112–121, 2002.

[11] Chiranjeeb Buragohain, Sorabh Gandhi, John Hershberger, and Subhash Suri. Contour approximation in sensor networks. In *Proceedings of the 2006 International Conference on Distributed Computing in Sensor Systems (DCOSS'06)*, San Francisco, California, USA, 2006.

[12] Jae-Hwan Chang and Leandros Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *Proceedings of the 19th Conference of the IEEE Communications Society (INFOCOM'00)*, pages 22–31, Tel Aviv, Israel, 2000.

[13] Jae-Hwan Chang and Leandros Tassiulas. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Transaction On Networking*, 12(4):609–619, 2004.

[14] Shigang Chen and Zhan Zhang. Localized algorithm for aggregate fairness in wireless sensor networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking (MobiCom'06)*, Los Angeles, California, USA, 2006.

[15] Krishna Chintalapudi, Jeongyeup Paek, Omprakash Gnawali, Tat Fu, Karthik Dantu, John Caffrey, Ramesh Govindan, and Erik Johnson. Structural damage detection and localization using netshm. In *Proceedings of The Fifth International Symposium on Information Processing in Sensor Networks (IPSN'06)*, pages 475–482, Nashville, Tennessee, USA, 2006.

[16] David Chu, Amol Deshpande, Joseph M. Hellerstein, and Wei Hong. Approximate data collection in sensor networks using probabilistic models. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*, Atlanta, GA, USA, 2006.

[17] Erik H. Clayton, Bong hwan Koh, Guoliang Xing, Chien liang Fok, Shirley J. Dyke, and Chenyang Lu. Damage detection and correlation-based localization using wireless mote sensors. In *Proceedings of the 13th Mediterranean Conference on Control and Automation*, Limassol, Cyprus, 2005.

[18] Erik H. Clayton, Bong-Hwan Koh, Guoliang Xing, Chien-Liang Fok, Shirley J. Dyke, and Chenyang Lu. Damage detection and correlation-based localization using wireless mote sensors. In *Proceedings of the 2005 IEEE International Symposium on Mediterranean Conference on Control and Automation*, Cyprus, Hawaii, USA, 2005.

[19] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Second Edition*. The MIT Press, 2001.

[20] Razvan Cristescu, Baltasar Beferull-Lozano, and Martin Vetterli. On network correlated data gathering. In *Proceedings of the 23rd Conference of the IEEE Communications Society (INFOCOM'04)*, Hong Kong, China, 2004.

[21] Rita Cucchiara, Andrea Prati, Roberto Vezzani, Luca Benini, Elisabetta Farella, and Piero Zappi. Using a wireless sensor network to enhance video surveillance. *Journal of Ubiquitous Computing and Intelligence*, pages 1–9, 2005.

[22] Amol Deshpande, Carlos Guestrin, Samuel R. Madden, Joseph M. Hellerstein, and Wei Hong. Model-driven data acquisition in sensor networks. In *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04)*, Toronto, Canada, 2004.

[23] Enrique J. Duarte-Melo and Mingyan Liu. Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks. In *Proceedings of IEEE Globecom 2002*, Taibei, Taiwan, 2002.

[24] Kai-Wei Fan, Sha Liu, and Prasun Sinha. On the potential of structure-free data aggregation in sensor networks. In *Proceedings of the 25th Conference of the IEEE Communications Society (INFOCOM'06)*, Barcelona, Catalunya, SPAIN, 2006.

[25] Kai-Wei Fan, Sha Liu, and Prasun Sinha. Scalable data aggregation for dynamic events in sensor networks. In *Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys'06)*, Boulder, Colorado, USA, 2006.

[26] Laura Marie Feeney and Martin Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *Proceedings of the 20th Conference of the IEEE Communications Society (INFOCOM'01)*, Anchorage, Alaska, USA, 2001.

[27] Markus Flierl and Bernd Girod. Coding of multi-view image sequences with video sensors. In *Proceedings of IEEE International Conference on Image Processing*, Atlanta, Georgia , USA, 2006.

[28] R. G. Gallager, P. A. Humblet, and P. M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Transactions on Programming Language and Systems*, 5(1):66–77, 1983.

[29] Sorabh Gandhi, John Hershberger, and Subhash Suri. Approximate iso-contours and spatial summaries in sensor networks. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN'07)*, Cambridge, Massachusetts, USA, 2007.

[30] Ashish Goel and Deborah Estrin. Simultaneous optimization for concave costs: Single sink aggregation or single source buy-at-bulk. In *Proceedings of the 14th annual ACM-SIAM symposium on Discrete algorithms*, pages 499–505, Baltimore, Maryland, USA, 2003.

[31] Carlos Guestrin, Peter Bodi, Romain Thibaux, Mark Paskin, and Samuel Madden. Distributed regression:an efficient framework for modeling sensor network data. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks (IPSN'04)*, Berkeley, California, USA, 2004.

[32] Himanshu Gupta, Vishnu Navda, Samir R. Das, and Vishal Chowdhary. Efficient gathering of correlated data in sensor networks. In *Proceedings of MobiHoc'05*, pages 402–413, Urbana-Champaign, Illinois, USA, May 2005.

[33] Tian He, Lin Gu, Liqian Luo, Ting Yan, John A. Stankovic, and Sang H. Son. An overview of data aggregation architecture for real-time tracking with sensor networks. In *Proceedings of the 20th International Parallel and Distributed Processing Symposium (IPDPS'06)*, Rhodes Island, GREECE, 2006.

[34] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Hawaii International Conference on System Science(HICSS'00)*, pages 3005–3014, Maui, Hawaii, USA, 2000.

[35] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. In *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems, Cambridge*, pages 93–104, Cambridge, Massachussetts, USA, 2000.

[36] Dorit S. Hochbaum. *Approximation Algorithms for NP-Hard Problems*. Operations Research, Etcheverry Hall, 1997.

[37] I.F.Akylidiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38:393–422, 2002.

[38] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom'00)*, pages 56–67, Boston, Massachussetts, USA, 2000.

[39] Rajagopal Iyengar and Biplab Sikdar. Scalable and distributed gps free positioning for sensor networks. In *Proceedings of the IEEE International Conference on Communications (ICC03)*, pages 338–342, Anchorage, Alaska, USA, May 2003.

[40] Dragan Petrovic Jim Chou and Kannan Ramchandran. A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks. In *Proceedings of The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 13–24, San Francisco, California, USA, 2003.

[41] Apoorva Jindal and Konstantinos Psounis. Modeling spatially correlated data in sensor networks. *ACM Transactions on Sensor Networks*, 2(4):466–499, 2006.

[42] Konstantinos Kalpakis, Koustuv Dasgupta, and Parag Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Networks*, 42(6):697–716, 2003.

[43] Li-Wei Kang and Chun-Shien Lu. Low-complexity wyner-ziv video coding based on robust media hashing. In *Proceedings of the 8th IEEE Workshop on Multimedia Signal Processing*, Victoria, BC, Canada, 2006.

[44] Wooyoung Kim, Kirill Mechitov, Jeung-Yoon Choi, and Soo Ham. On target tracking with binary proximity sensors. In *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks (IPSN'05)*, Los Angeles, California, USA, 2005.

[45] Bhaskar Krishnamachari, Deborah Estrin, and Stephen B. Wicker. The impact of data aggregation in wireless sensor networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems*, pages 575–578, 2002.

[46] Pei-Kai Liao, Min-Kuan Chang, and C.-C.Jay Kuo. Contour line extraction with wireless sensor networks. In *Proceedings of the 2005 IEEE International Conference on the Communications (ICC'05)*, Seoul, Korea, May 2005.

[47] Pei-Kai Liao, Min-Kuan Chang, and C.-C.Jay Kuo. A distributed approach to contour line extraction using sensor networks. In *Proceedings of the 2005 IEEE Vehicular Technology Conference (VTC'05)*, Dallas, Texas, USA, 2005.

[48] Hock Beng Lim, Keck Voon Ling, Wenqiang Wang, Yuxia Yao, Mudasser Iqbal, Boyang Li, Xiaonan Yin, and Tarun Sharma. The national weather sensor grid. In *Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems (SenSys07)*, pages 369–370, Sydney, Australia, 2007.

[49] Stephanie Lindsey and Cauligi Raghavendra. Pegasis: Power-efficient gathering in sensor information systems. In *Proceedings of IEEE Aerospace Conference*, volume 3, pages 1125–1130, Big Sky, Montana, USA, 2002.

[50] Chong Liu, Kui Wu, and J. Pei. An energy efficient data collection framework for wireless sensor networks by exploiting spatiotemporal correlation. *IEEE Transactions on Parallel and Distributed Systems*, 18(7):1010–1023, 2007.

[51] Hai Liu, Xiaohua Jia, Pengjun Wan, Chih-Wei Yi, S. Kami Makki, and Niki Pissinou. Maximizing lifetime of sensor surveillance systems. *IEEE Transactions on Networking*, 15(2):334–345, 2007.

[52] Juan Liu, James Reich, and Feng Zhao. Collaborative in-network processing for target tracking. *IEURASIP Journal of Applied Signal Processing*, 2003(4):378–391, 2003.

[53] Junning Liu, Micah Adler, Don Towsley, and Chun Zhang. On optimal communication cost for gathering correlated data through wireless sensor networks. In *Proceedings of The the 12th annual international conference on Mobile computing and networking (MobiCom'06)*, pages 310–321, Los Angeles, California, USA, 2006.

[54] Hong Luo, Jun Luo, Yonghe Liu, and Sajal K. Das. Energy efficient routing with adaptive data fusion in sensor networks. In *Proceedings of the 2005 joint workshop on Foundations of mobile computing Workshop on Discrete Algothrithms and Methods for MOBILE Computing and Communications (DIALM-POMC'05)*, Cologne, Germany, 2005.

[55] Huadong Ma and Yonghe Liu. Correlation based video processing in video sensor networks. In *Proceedings of 2005 International Conference on Wireless Networks, Communications and Mobile Computing*, Maui, Hawaii, USA, 2005.

[56] Ritesh Madan and Sanjay Lall. Distributed algorithms for maximum lifetime routing in wireless sensor networks. In *Proceedings of the IEEE Global Telecommunications Conference(GLOBECOM)*, pages 748–753, Dallas, Texas, USA, 2004.

[57] Samuel Madden. *The Design and Evaluation of a Query Processing Architecture for Sensor Networks*. PhD thesis, UC Berkeley.

[58] Samuel Madden, Michael J. Franklin, Joseph Hellerstein, and Wei Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. In *Proceedings of the 5th Synposium on Opetrating Systems Design and Implementation(OSDI'02)*, pages 131–146, Boston, Massachussetts, USA, 2002.

[59] Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler, and John Anderson3. Wireless sensor networks for habitat monitoring. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, Atlanta, Georgia, USA, 2002.

[60] Amit Manjhi, Suman Nath, and Phillip B. Gibbons. Tributaries and deltas: Efficient and robust aggregation in sensor network streams. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, Baltimore, Maryland, USA, 2005.

[61] Miklos Maroti. Directed flood-routing framework for wireless sensor networks. In *Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*, Toronto, Ontario, Canada, 2004.

[62] Tommaso Melodia, Dario Pompili, Vehbi C. Gungor, and Ian F. Akyildiz. A distributed coordination framework for wireless sensor and actor networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'05)*, pages 99–110, Urbana-Champaign, Illinois, USA, May 2005.

[63] Xiaoqiao Meng, Thyaga Nandagopal, Li Li, and Songwu Lu. Contour maps: Monitoring and diagnosis in sensor networks. *Computer Networks*, 50:2820–2838, 2006.

[64] Tomonori Nagayama, B. F. Spencer, Gul Agha, and Kirill Mechitov. Model-based data aggregation for structural monitoring employing smart sensors. In *Proceedings of the 3rd International Conference on Networked Sensing Systems (INSS'06)*, Chicago, Illinois, USA.

[65] Tomonori Nagayama, B. F. Spencer, Gul Agha, and Kirill Mechitov. Vehicle detection and tracking using acoustic and video sensors. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'04)*, Montreal, Canada, May 2004.

[66] Stephan Olariu and Ivan Stojmenovic. Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting. In *Proceedings of the 25th Conference of the IEEE Communications Society (INFOCOM'06)*, Barcelona, Catalunya, SPAIN, 2006.

[67] Mark Perillo, Zhao Cheng, and Wendi Heinzelman. On the problem of unbalanced load distribution in wireless sensor networks. In *Proceedings of the 47th annual IEEE Global Telecommunications Conference (GLOBECOM'04)*, Dallas, Texas, USA, 2004.

[68] S. Sandeep Pradhan, Julius Kusuma, and Kannan Ramchandran. Distributed compression in a dense microsensor network. *IEEE Signal Processing Magazine*, 19(2):51–60, 2002.

[69] S. Sandeep Pradhan and Kannan Ramchandran. Distributed source coding using syndromes (discus): Design and construction. *IEEE Transaction on Information Theory*, 49:626–643, 2003.

[70] Wanzhi Qiu, Efstratios Skafidas, and Peng Hao. Enhanced tree routing for wireless sensor networks. *Ad Hoc Networks*, 7(3):638–650, 2009.

[71] Narayanan Sadagopan and Bhaskar Krishnamachari. Maximizing data extraction in energy-limited sensor networks. In *Proceedings of the 23rd Conference of the IEEE Communications Society (INFOCOM'04)*, Hong Kong, China, 2004.

[72] Arvind Sankar and Zhen Liu. Maximum lifetime routing in wireless ad-hoc networks. In *Proceedings of the 23rd Conference of the IEEE Communications Society (INFOCOM'04)*, Hong Kong, China.

[73] Anna Scaglione and Sergio D. Servetto. On the interdependence of routing and data compression in multihop sensor networks. In *Proceedings of the 8th Annual International conference on Mobile Computing and Networking (MobiCom'02)*, pages 140–147, Atlanta, Georgia, USA, 2002.

[74] Akiyoshi Shioura and Akihisa Tamura. Efficiently scanning all spanning trees of an undirected graph. *Journal of the Operations Research Society of Japan*, 38:331–344, 1995.

[75] Akiyoshi Shioura and Akihisa Tamura. An optimal algorithm for scanning all spanning trees of undirected graphs. *SIAM Journal on Computing*, 26(3):678–692, 1997.

[76] N. Shrivastava, R. Mudumbai U. Madhow, and S. Suri. Target tracking with binary proximity sensors: fundamental limits, minimal descriptions, and algorithms. In *Proceedings of the 4th international conference on Embedded networked sensor systems (SenSys'06)*, Boulder, Colorado, USA, 2006.

[77] Gyula Simon, Miklos Maroti, Akos Ledeczi, Gyorgy Balogh, Branislav Kusy, Andras Nadas, Gabor Pap, Janos Sallai, and Ken Frampton. Sensor network-based countersniper system. In *Proceedings of the 2th international conference on Embedded networked sensor systems (SenSys'04)*, Baltimore, Maryland, USA, 2004.

[78] Suresh Singh, Mike Woo, and C. S. Raghavendra. Power-aware routing in mobile ad hoc networks. In *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM'98)*, Dallas, Texas, USA, 1998.

[79] Duncan Smith and Sameer Singh. Approaches to multisensor data fusion in target tracking: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 18(12):1696–1710, 2006.

[80] Ignacio Solis and Katia Obraczka. Efficient continuous mapping in sensor networks using isolines. In *Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05)*, San Diego, California, USA, 2005.

[81] Ignacio Solis and Katia Obraczka. Isolines: efficient spatio-temporal data aggregation in sensor networks. *Wireless Communications and Mobile Computing*, 2007.

[82] Sumana Srinivasan and Krithi Ramamritham. Contour estimation using collaborating mobile sensors. In *Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*, Los Angeles, CA, USA, 2006.

[83] Huseyin Ozgur Tan, I. Korpeoglu, and Ivan Stojmenovic. A distributed and dynamic data gathering protocol for sensor networks. In *Proceedings of the 21st International Conference on Advanced Networking and Applications (AINA'07)*, pages 220–227, Niagara Falls, ON, Canada, May 2007.

[84] Xueyan Tang and Jianliang Xu. Extending network lifetime for precision-constrained data aggregation in wireless sensor networks. In *Proceedings of the 25th Conference of the IEEE Communications Society (INFOCOM'06)*, Barcelona, Catalunya, SPAIN, 2006.

[85] Xueyan Tang and Jianliang Xu. Extending network lifetime for precision-constrained data aggregation in wireless sensor networks. In *Proceedings of the 25th Conference of the IEEE Communications Society (INFOCOM'06)*, Barcelona, Catalunya, SPAIN, 2006.

[86] Preetha Thulasiraman, Srinivasan Ramasubramanian, and Marwan Krunz. Disjoint multipath routing to two distinct drains in a multi-drain sensor network. In *Proceedings of the 26th Conference of the IEEE Communications Society (INFOCOM'07)*, Anchorage, Alaska, USA, May 2007.

[87] S. Tilak, N.B. Abu-ghazaleh, and W. Heinzelman. A taxonomy of wireless micro-sensor network models. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6:28–36, 2002.

[88] M. Tubaishat and S. Madria. Sensor networks: an overview. *IEEE Potentials*, 22:20–23, 2003.

[89] Mehmet C. Vuran, Ozgr B. Akan, and Ian F. Akyildiz. Spatio-temporal correlation: theory and applications for wireless sensor networks. *Computer Networks*, 45(3):245–259, 2004.

[90] Qixin Wang, Wei-Peng Chen, Rong Zheng, Kihwal Lee, and Lui Sha. Acoustic target tracking using tiny wireless sensor devices. In *Proceedings of The Second International Symposium on Information Processing in Sensor Networks (IPSN'03)*, Palo Alto, CA, USA, 2003.

[91] Ning Xu, Sumit Rangwala, Krishna Kant Chintalapudi, Deepak Ganesan, Alan Broad, Ramesh Govindan, and Deborah Estrin. A wireless sensor network for structural monitoring. In *Proceedings of The Second International Conference on Embedded Networked Sensor Systems*, pages 13–24, Baltimore, Maryland, USA, 2004.

[92] Wenwei Xue, Qiong Luo, Lei Chen, and Yunhao Liu. Contour map matching for event detection in sensor networks. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data(SIGMOD'06)*, Chicago, Illinois, USA, 2006.

[93] Ting Yan, Tian He, and John A. Stankovic. Differentiated surveillance for sensor networks. In *Proceedings of the 1st ACM Conference on Embedded Networked Sensor Systems (SenSys03)*, pages 51–62, Los Angeles, California, USA, 2003.

[94] Wei Ye, John Heidemann, and Deborah Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM TRANSACTIONS ON NETWORKING*, 12(3):493–506, 2004.

[95] Ossama Younis and Sonia Fahmy. Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, 3(4):366–379, 2004.

[96] Liyang Yu, Neng Wang, and Xiaoqiao Meng. Real-time forest fire detection with wireless sensor networks. In *Proceedings of the 1st IEEE International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM'05)*, pages 1214–1217, Wuhan, China, 2005.

[97] Kevin Yuen, Baochun Li, and Ben Liang. Distributed data gathering in multi-sink sensor networks with correlated sources. In *Proceedings of IFIP Networking 2006*, Coimbra, Portugal, May 2006.

[98] Jerry Zhao, Ramesh Govindan, and Deborah Estrin. Computing aggregates for monitoring wireless sensor networks. In *Proceedings of The First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA'03)*, Anchorage, AK, USA, May 2003.

[99] Congzhou Zhou and Bhaskar Krishnamachari. Localized topology generation mechanisms for wireless sensor network. In *Proceedings of IEEE Globecom 2003*, San Francisco, California, USA, 2003.

[100] Shoudong Zou, Ioanis Nikolaidis, and Janelle J. Harms. Efficient data collection trees in sensor networks with redundancy removal. In *Proceedings of Ad-Hoc, Mobile, and Wireless Networks: Third International Conference(ADHOC-NOW'04)*, pages 252–265, Vancouver, Canada, 2004.