Role Assignment in Wireless Sensor Networks: Energy-Efficient Strategies and Algorithms

by

Mark Adam Perillo

Submitted in Partial Fulfillment
of the
Requirements for the Degree
Doctor of Philosophy

Supervised by
Dr. Wendi B. Heinzelman
Department of Electrical and Computer Engineering
The College
School of Engineering and Applied Sciences

University of Rochester
Rochester, New York

2007

Curriculum Vitae

The author was born in Buffalo, New York, on July 22, 1978. He attended the University of Rochester from 1996 to 2000, and graduated with a Bachelor of Science in Electrical and Computer Engineering (High Distinction) in 2000. He began graduate studies at the University of Rochester in 2000 and was awarded an NDSEG fellowship. He has pursued his research in wireless communications and ad hoc and sensors network under the supervision of Dr. Wendi Heinzelman and received his Masters of Science in Electrical and Computer Engineering in 2002. Since May 2005, he has worked at Syracuse Research Corporation as a research engineer.

Acknowledgements

I would like to thank Wendi Heinzelman for the opportunity to work with her over the past seven years. Her encouraging words and positive reinforcement helped me to remain confident in myself and my work and motivated to keep working toward my doctorate degree during times when I felt lost. Wendi has been a terrific role model and a pleasure to work with during my tenure as a graduate student at the University of Rochester.

I have had the pleasure of working with many great faculty members over the years: Mark Bocko, who advised me during the early stages of graduate school, provided me with an internship opportunity, and served on my proposal and thesis committees; Zeljko Ignjatovic, who collaborated on the work in this dissertation and was a great friend to me over the years; Ed Titlebaum, who was a mentor to me as a teaching assistant; Amy Murphy, who collaborated on the work in this dissertation; Gaurav Sharma and Shanchieh Yang, who served on my proposal committee; and Chen Ding, Henry Kautz, and Azadeh Vosoughi, who served on my thesis committee.

I would like to thank to all my colleagues at the University of Rochester in the Wireless Communications and Networking Group for their help. Specifically, I would like to thank Hervaldo Carvalho, Lei Chen, Zhao Cheng, Chris Merlin, Tolga Numanoglu, Stanislava Soro, and Bulent Tavli. It has been great to work in an environment in which the atmosphere is much more collaborative than competitive. My colleagues have always been willing to take some time out of their day to explain a topic of their expertise. Most importantly, each of them has been a good friend and I hope to maintain these friendships over the coming years.

I also would like to thank all of my friends and family, most of all my wife Meghann, for offering words of encouragement and support in times when I most needed it.

This research was made possible financially in part by the Office of Naval Research through the NDSEG fellowship, the National Science Foundation through grant CNS-0448046, and Syracuse Research Corporation.

Abstract

Wireless sensor networks are expected to consist of large numbers of nodes with redundant capabilities. This redundancy may exist in the abilities of the sensor nodes to sense the environment, route data from other sensors, and aggregate and code redundant data, among other tasks. In order to allow the network to operate at a high level of fidelity, or QoS, for an extended period of time, the assignment of roles and protocol-level settings to nodes should be carefully considered and chosen so that a node's energy is used most efficiently to meet the end goal of the sensor network application. Decisions concerning role assignment should be made based on such node properties as location in the environment (both absolute and relative to other nodes), sensing capabilities, energy supply, and processing capabilities. In this dissertation, I present a general middleware framework under which these optimizations can be made and I present several ways that role assignment can be optimized to extend network lifetime, including application-aware routing cost assignment, spatial resolution management, and transmission power control.

Contents

Table of Contents **vi**

List of Tables **x**

List of Figures **xii**

1 Introduction **1**

1.1 Overview of Wireless Sensor Networks 2
1.2 Research Motivation 3
1.3 Research Contributions 5
1.4 Dissertation Structure 6

2 Related Work **7**

2.1 Middleware for Ad Hoc and Sensor Networks 7
2.2 Dynamic Sensor Selection 11
2.3 Routing in Ad Hoc and Sensor Networks 16
  2.3.1 Resource-Aware Routing 16
  2.3.2 Data-Centric Routing 18
  2.3.3 Geographic Routing 23
  2.3.4 Clustering Algorithms 25
  2.3.5 Topology Control 27
  2.3.6 Multicasting 30
  2.3.7 Distributed Multicasting 31
2.4 Transmission Range Optimization 33
2.5 Sensor Deployment Strategies 34

3 A Sensor Network Framework and Architecture **36**
3.1 Middleware Architecture . . . 37
3.1.1 Application Performance . . . 39
3.1.2 Network Control . . . 44
3.1.3 Tradeoffs . . . 46
3.2 Summary . . . 47

4 Optimizing Sensor Role Management **48**
4.1 Single Hop Wireless Sensor Networks . . . 50
4.1.1 Large-Scale Networks . . . 52
4.2 Multi-hop Wireless Sensor Networks . . . 56
4.2.1 Consideration for Networks with Non-Static Data Sinks . . . 57
4.2.2 Joint Optimization for Multi-hop Sensor Networks . . . 57
4.2.3 Simulations . . . 59
4.3 Summary . . . 60

5 Sensor Resolution Management **62**
5.1 Blue-Noise Spatial Sampling . . . 62
5.2 Application to Sensor Resolution Management . . . 65
5.3 Incorporation of an Energy Cost . . . 66
5.4 Distribution of the Algorithm . . . 67
5.5 Lifetime Optimization . . . 68
5.6 Simulations . . . 70
5.7 Summary . . . 76

6 Application-Aware Role Assignment **78**
6.1 Application-Aware Routing Costs . . . 79
6.1.1 Sensing and Network Model . . . 80
6.1.2 Worst-Coverage-Based Cost . . . 81
6.1.3 Comprehensive-Coverage-Based Cost . . . 82
6.1.4 Combining Several Cost Functions . . . 83
6.2 DAPR - Distributed Activation with Predetermined Routes . . . 84
6.2.1 Simulations and Analysis . . . 88
6.2.2 Performance of Application-Aware Routing Costs . . . 91

6.2.3 Effect of Sensor Selection Criteria 94
6.2.4 Combining Routing Costs 95
6.2.5 Comparison With Optimizations 97
6.3 Summary 100

7 Applying Application-Aware Costs to Multicast Traffic Patterns 101
7.1 Multicasting in a Wireless Environment 101
7.1.1 Simulations 104
7.2 Distributing the Multicast Algorithm 107
7.2.1 Simulations 107
7.3 Summary 108

8 Optimal Transmission Power Control for Sensor Networks 111
8.1 Assumptions 112
8.2 Lifetime Optimization 114
8.3 Simulations 117
8.3.1 One-Dimensional Scenario 117
8.3.2 Two-Dimensional Sensor Fields 124
8.4 Summary 127

9 Cost-Efficiency of Sensor Network Deployment Strategies 128
9.1 Deployment Strategy Options 129
9.2 Normalized Network Lifetime 130
9.3 Cost Model 132
9.4 Comparison of Different Deployment Strategies 133
9.4.1 Deployment Strategy DS_1: Single Static Sink 133
9.4.2 Deployment Strategy DS_2: Mobile Data Sink 136
9.4.3 Deployment Strategy DS_3: Multiple Data Sinks/Clustering . . . 138
9.4.4 Deployment Strategy DS_4: Non-uniform Energy Assignment . 141
9.4.5 Deployment Strategy DS_5: Non-uniform Relay/Sensors 141
9.4.6 Deployment Strategy DS_6: Non-uniform Traffic Generation . . 143
9.4.7 Cost Comparison of Deployment Strategies 143
9.5 Summary 147

10 Conclusions and Future Work 149
10.1 Summary of Contributions 149
10.2 Future Work 150

A Publications 163

List of Tables

3.1 Feasible sets F_A for the personal health monitor application. 45

4.1 Variables measured by different sensors in a personal health monitoring application. 52

6.1 Coverage overlap statistics for the three simulated deployment scenarios. 90
6.2 Default simulation parameters for DAPR simulations. 91
6.3 Simulation results for different routing costs in the uniform deployment scenario. 92
6.4 Simulation results for different routing costs in the clustered deployment scenario. 93
6.5 Simulation results for different routing costs in the video scenario. . . . 94
6.6 Network lifetime when using the worst-coverage-based routing cost with different selection criteria. 95
6.7 Network lifetime for heterogeneous networks when using a combination of worst-coverage-based and energy-aware routing costs. 97
6.8 Network lifetime for heterogeneous networks when using a combination of comprehensive-coverage-based and energy-aware routing costs. . 98

7.1 Simulation results for different routing costs in the clustered deployment scenario (multicast). 106
7.2 Simulation results for different routing costs in the clustered deployment scenario (distributed multicast). 109

8.1 Variables used in network modeling throughout this chapter. 113

9.1 Sensor network deployment strategies, corresponding scenarios, and potential difficulty/extra costs. 130
9.2 Network parameters for the sample scenario. 134
9.3 Network parameters for the target scenario. 144
9.4 Cost evaluation for the two cost scenarios with transmission power control. 146
9.5 Cost evaluation for the two cost scenarios without transmission power control. 146

List of Figures

2.1 A sponsored sector, as defined by the node self scheduling algorithm. . 13

2.2 Schedule calculation for a single grid point, as proposed in the reference time-based scheduling scheme. 14

2.3 The CCP activation rule for K-coverage, K = 1. 15

2.4 Message exchange in the SPIN protocol. 19

2.5 Problems with blind flooding of sensor data. 20

2.6 Establishing gradients in Directed Diffusion. 21

2.7 Query handling in Rumor Routing. 22

2.8 GPSR greedy forwarding policy and perimeter routing algorithm. 24

2.9 Possible trajectories to use in TBF for robust multipath routing, spoke broadcasting, and broadcast within a remote region. 24

2.10 Adaptive clustering of the network. 25

2.11 Example of a GAF virtual grid. 29

3.1 System that employs MiLAN. 38

3.2 MiLAN components. 40

3.3 High level overview of MiLAN operation. 41

3.4 Example State-based Variable Requirements Graph for for the personal health monitor application. 43

3.5 Example Sensor QoS Graph for the personal health monitor example. . 44

4.1 Example sensor coverage scenario. 49

4.2 Garg-Könemann algorithm. 53

4.3 Garg-Könemann algorithm applied to the sensor scheduling problem. . 54

4.4 Network lifetime vs. network size for Garg-Könemann optimizations. . 55

4.5 Round length vs. ϵ for the Garg-Könemann algorithm in a network of 200 sensor nodes. 55

4.6 Comparison of approaches for schedule and routing optimization for $P_{sense} = 0$, $P_{sense} = 50\mu W$, and $P_{sense} = 10mW$. 61

5.1 Blue noise and white noise sampling patterns. 64

5.2 Distribution of the blue noise sensor selection algorithm. 69

5.3 Mean square error of reconstructed signal using the blue noise, grid-based, and random selection methods for a phenomenon bandlimited to $\frac{1}{20}m^{-1}$. 71

5.4 Mean square error of reconstructed signal using the blue noise, grid-based, and random selection methods for a phenomenon bandlimited to $\frac{1}{15}m^{-1}$. 72

5.5 Selection optimality versus load balancing tradeoff imposed by the assignment of an energy cost. 73

5.6 Mean square error of optimized sensor schedule (mean and standard deviation) compared with mean square error using the blind selection approach. 74

5.7 Initial and subsequent sets chosen through blind blue-noise sensor selection method. 74

5.8 Sampling of sensor subsets used in the optimal sensor scheduling method. 75

5.9 Mean square error of optimized sensor schedule (mean and standard deviation) compared with mean square error using the blind selection approach, for randomly distributed initial energy. 76

6.1 Example sensor network illustrating the advantages of application-aware costs. 82

6.2 DAPR protocol timeline. 85

6.3 Example sensor deployment patterns for the uniform deployment scenario, clustered deployment scenario, and video scenario. 90

6.4 Coverage degradation over time for different routing costs in the uniform deployment scenario. 92

6.5 Coverage degradation over time for different routing costs in the clustered deployment scenario. 93

6.6 Coverage degradation over time for different routing costs in the video scenario. 94
6.7 Comparison of DAPR with globally optimal solution calculated using linear programming. 99

7.1 Example of a BIP tree construction. 103
7.2 Coverage degradation over time for different routing costs in the clustered deployment scenario (multicast). 105
7.3 Coverage degradation over time for different routing costs in the clustered deployment scenario (distributed multicast). 108

8.1 A one-dimensional topology network. 117
8.2 Optimal transmission range distribution for a one-dimensional network with a $500m$ radius and $5m$ spacing between nodes. 118
8.3 Lifetime vs. percentage of total network energy consumed. 120
8.4 Optimal transmission range distribution and energy consumption as a function of distance for 100% and 90% of total network energy consumed.121
8.5 Optimal transmission range distribution as a function of distance from the base station for $5m$, $10m$, $20m$, and $100m$ spacing, $500m$ radius. . . 122
8.6 Optimal transmission range distribution as a function of distance from the base station for radius of $250m$ and $1000m$, $5m$ sensor spacing. . . . 123
8.7 Lifetime vs. energy consumption tradeoff for network radius of $250m$, $500m$, and $1000m$. 123
8.8 Two-dimensional sensor field and one-dimensional modeling. 124
8.9 Optimal transmission range distribution as a function of distance from the base station for a two-dimensional sensor field with radius of $250m$ and $1000m$. 125
8.10 Lifetime vs. percentage of the total energy consumed in the network for a two-dimensional sensor field with various network radii. 126
8.11 Optimal transmission range distribution as a function of distance from the base station for a randomly deployed two-dimensional sensor field with radius of $250m$, $500m$, and $1000m$. 126

9.1 Network lifetime as a function of network radius for the optimal power control scheme, the fixed transmission power scheme and the heuristic power control scheme in a two-dimensional scenario. 135

9.2 Data sink locations for the lifetime optimization using 1, 2, 3, and 4 data sink locations. 136

9.3 Normalized lifetime vs. number of data sinks deployed for the sample scenario with transmission power control and without transmission power control. 138

9.4 Normalized lifetime vs. number of cluster heads deployed with transmission power control and without transmission power control. 140

9.5 Normalized lifetime as network radius varies for the non-uniform energy assignment strategy. 142

9.6 Energy distribution map for the sample scenario with transmission power control and without transmission power control. 142

9.7 Number of sensors required to meet the target scenario lifetime when using transmission power control and when not using transmission power control for the multiple data sinks deployment strategy. 145

9.8 Total cost of deployment for target scenario when using transmission power control and when not using transmission power control for the multiple data sinks deployment strategy. 145

Chapter 1

Introduction

The area of wireless sensor networks has become a hot area of research in recent years. The ability to network tiny, easily deployed sensor devices to procure a useful overall description of current environmental conditions will enable the realization of applications previously considered impossible. Recently, research from the networking community as well as advances in micro-fabrication technology have brought about the realization of practical commercial wireless sensor networks. In the coming years, these networks are expected to grow to large numbers of nodes (thousands) as the cost of manufacturing the sensors continues to drop significantly. My thesis is that protocol design for wireless sensor networks should leverage the fact that while this type of massive distribution of nodes can allow networks to perform at a high level, adequate performance can be achieved when requiring only a subset of nodes to perform each of the many roles (e.g., active sensor, router, aggregator, etc.) necessary for the network to operate. In fact, due to the tight energy constraints of sensor nodes, limiting role assignment in such a way can dramatically increase the lifetime of the network while minimally reducing the quality of data collected on the network. Furthermore, these roles should be assigned in a manner such that each node assumes the optimal role to best support the end goal of the sensor network so that sensor network lifetime is maximized.

1.1 Overview of Wireless Sensor Networks

The list of potential uses for wireless sensor networks seems endless, with applications from areas such as security, medicine, industrial machinery monitoring, the military, and a plethora of others. Imagine a large building equipped with thousands of wireless sensor devices, which are able to monitor a number of phenomena occurring within the building (e.g., temperature, chemical, presence of humans, etc.). In case of an emergency such as a fire in the building, these sensors could collaborate and detect the exact locations where the fire exists, directing inhabitants to a safe exit of the building. In addition, rescue teams may be able to poll the network to determine if there are any remaining people in the building and their locations, while guiding the rescuers through a safe rescue plan.

Sensor networks can also be used in a preventative manner to reduce or avoid catastrophes altogether. Consider stress sensors that are embedded into the structural support of a building. These can be used in a proactive manner to monitor the condition of the building, which can be especially important in instable areas such as earthquake-prone regions. Other sensors may be used to detect unwanted foreign intruders, facilitating the job of security personnel working at the building.

Another example of a useful wireless sensor network may be found in the health care community. Several universities have begun projects involving intelligent homes, in which sensor networks deployed within the home can assist in the daily life of its residents [1, 2]. In addition to sensors that may be mounted directly on the residents to measure such health signals as blood pressure and ECG waveforms, other sensors distributed throughout the house may be able to track objects, allowing them to be easily found once misplaced.

A wireless sensor node typically consists of four main components: the sensor itself (including any analog-to-digital conversion), some type of processor, a radio, and an energy supply. In fact, multi-modal sensors may have multiple sensors, and depending on the sophistication, multiple processors to handle different functionalities. A protocol stack resides on the processor, where software is written to provide medium access control (MAC), routing, and transport layer services. In addition, other services such as localization, topology control, and QoS (or fidelity) management may exist and be controlled by the processor. This dissertation primarily addresses the protocols and services on this stack, and it describes ways to design them so that energy-efficiency is achieved while QoS requirements are met.

1.2 Research Motivation

Wireless sensor networks can be thought of as a class of general ad hoc networks, and in fact there are many similarities between them, including the following

- Both are constrained by the limited energy supply of the nodes in the network.
- Communication is unreliable due to the wireless medium.
- In general, both are expected to be self-configuring, requiring little or no human intervention.

However, several unique features also exist in wireless sensor networks. These features present new challenges and require modification of designs for traditional ad hoc networks.

- While traditional ad hoc networks consist of network sizes on the order of tens, sensor networks are expected to scale to sizes of thousands.
- Sensor nodes are generally immobile, so the mechanisms used in traditional ad hoc network protocols to deal with mobility may be unnecessary and overweight.
- Nodes may be deployed in harsh environmental conditions, meaning that unexpected node failure is more common.
- Sensor nodes may be much smaller than nodes in traditional ad hoc networks (e.g., PDAs, laptop computers), requiring smaller batteries that lead to shorter lifetimes, less capable processors, and less bandwidth.
- Additional services, such as location information, may be required in wireless sensor networks.
- While nodes in traditional ad hoc networks compete for resources such as bandwidth, nodes in a sensor network can be expected to behave more cooperatively, as they attempt to accomplish a common universal goal.

- Cooperation extends beyond the absence of competition for resources, and nodes should actually collaborate to reduce the amount of data generated and sent onto the network.

Perhaps the most important aspect to consider in the design of wireless sensor networks is related to the final item. Quality of Service definitions used in wired and some wireless networks have been reformulated for sensor networks so that they relate to the overall application or system performance. The goal of large-scale wireless sensor networks is to gather information to monitor the environment. The overall quality of this information in monitoring the environment is more important than the amount of information. Each application may have its own unique interpretation of data quality, but in general, the cumulative data gathered on the network should meet minimum quality (i.e., fidelity) requirements. At the same time, sensor networks are expected to last for a long time (months or even years) without recharging the limited energy supplies of the individual sensors. It is essential to manage the energy supply of these nodes to ensure that application QoS is met while achieving maximum network lifetime.

A sensor network is essentially a distributed network of data sources that provides information about environmental phenomena to an end user or multiple end users. Typically, data from the individual sensors are routed via other sensors to sink points in the network (base stations), through which the user accesses the data. As the cost of manufacturing nodes becomes cheaper, it can be expected that often times, sensors will be deployed with great enough density that activating every sensor in the network provides little more quality of service (QoS) to the sensor network application than what could have been provided with many fewer sensors (i.e., the marginal quality provided by many of the sensors is minimal). If sensors are not needed at a given time to provide data or route other sensors' data, they can save energy by shutting down (i.e., going into "sleep" mode) or halting traffic generation until they are needed at a later time. In fact, activating all of the sensors can be detrimental to the overall task of the sensor network if there is so much traffic on the network that congestion is noticeable [3]. In this case, network throughput can degrade significantly, and important data may be dropped as packet queues at the sensor nodes overflow. Among the data packets that do reach the data sink(s), high packet delays may be introduced, rendering the data useless.

In cases of unnecessarily high sensor redundancy, only a subset of the deployed sensors should gather data so that there is no unnecessary redundancy and the network can operate at a point where the cumulative sensor data quality is sufficient to meet the application's goals, network congestion is minimal, and energy-efficiency is achieved. Thus, the need for sensor network management arises. Several protocols and algorithms have been proposed in the literature in which sensors determine the roles in which they should operate through distributed, local decision making. This work continues this trend, using an integrated approach in which the selection of nodes for active sensing, routing, cluster head operation, data compression, and other special network roles is combined within a single framework.

1.3 Research Contributions

This dissertation introduces several protocols and a general framework that address several open research issues in wireless sensor networks, as described previously. Specific contributions to the wireless sensor network research community include:

- A middleware framework is designed that allows sensor networks that have been deployed in an ad hoc fashion to maintain high-level QoS goals over an extended period of time. This middleware has been implemented on top of a Bluetooth network.
- It is shown how sensor role management, transmission ranges, and deployment plans can be optimized in multihop wireless sensor networks where individual nodes have energy constraints. While these optimizations may not be practical in large-scale sensor networks, they provide a benchmark against which practical distributed algorithms can be compared.
- Several distributed algorithms for controlling spatial resolution and multihop routing are presented.
- The routing algorithms are the first presented in the research community that consider a node's importance to the application over an extended period of time when deciding whether or not other sensors' data is routed through that node. Thus, this research is the first to present an "application-aware" routing metric.

- We analyze the energy-efficiency of wireless sensor networks characterized by many-to-one traffic patterns and provide a framework for analyzing the cost-efficiency of several deployment strategies that attempt to mitigate the sensor network hot spot problem.

1.4 Dissertation Structure

Related work from the current literature is first presented in Chapter 2. Chapter 3 describes a general sensor network middleware architecture that has been developed in collaboration with fellow researchers at the University of Rochester's Center for Future Health. Chapter 4 demonstrates how to optimize the roles of nodes in a sensor network under reliability (i.e., coverage) and energy constraints for a case in which global information is available at a central processor. Chapter 5 presents a protocol for sensor network resolution control in an application that requires the mapping of a phenomenon's data image at a central processor. Chapter 6 addresses the sensor role management problem from a routing perspective and presents application-aware routing costs and a distributed protocol that assigns optimal roles (e.g., data collector or router) to individual nodes so that their energy is used most wisely to support the end goal of the application. Chapter 7 expands this work and presents a multicasting framework that uses these application-aware routing costs. In Chapter 8, transmission range distribution and network deployment planning are optimized for the many-to-one (convergecast) traffic patterns seen in many wireless sensor network applications, and different strategies for solving this "hot spot" problem in an energy-efficient and cost-efficient manner are analyzed. Chapter 9 presents a model for comparing the energy-efficiency and cost-efficiency of several sensor network deployment strategies when transmission ranges and load distribution are optimized as in Chapter 8. The dissertation is summarized in Chapter 10.

Chapter 2

Related Work

In recent years, following the seminal paper of Estrin et al. [4] and exciting advances in wireless sensor technology such as Smart Dust [5], the field of protocol design for wireless sensor networks has grown dramatically. In this section, we review some of the work from the current literature that is relevant to this dissertation. We include a review of some of the existing middleware in the current literature, as well as an overview of methods that are used for sensor role selection, including routing and topology control protocols that determine routing roles as well as sensor selection protocols for coverage applications and distributed compression techniques that determine sensing roles. For a more complete review of the literature related to the general field of wireless sensor networks, the reader is referred to [6, 7].

2.1 Middleware for Ad Hoc and Sensor Networks

Middleware has often been useful in traditional systems for bridging the gap between the operating system (a low-level component) and the application, easing the development of distributed applications. Because wireless sensor networks share many properties with traditional distributed systems, it is natural to consider distributed computing middleware for use in sensor networks. Among existing distributed computing middleware, QoS-Aware Middleware [8] provides the closest example of a middleware that can support sensor network applications. This middleware is responsible for managing local operating system resources based on application requirements specified to the middleware. The application's QoS information is compiled into a QoS profile to guide the middleware in making resource use decisions.

While low layer protocol optimization provides the means to manage the limited sensor network resources, oftentimes the protocols cannot be effectively managed by the application. For example, suppose that an application wants to increase the resolution of data being collected. There are many ways to achieve this goal (e.g., perform less in-network aggregation, reduce sleep cycles for the active nodes and require them to transmit more data, etc.). As another example, suppose that the application is interested in only a certain type of data, such as any data that indicates an intrusion into the region of interest. Using low layer management of the network, the application cannot easily manage its resources to ensure that only data of interest is transmitted. For complex data-centric and event-centric applications, it is unreasonable to expect applications to manage the data flow and network resources themselves. A middleware that provides a well-defined API can be used to provide services to manage the data flow and network resources on behalf of the application. Recently, much work has targeted the development of middleware specifically designed to meet the challenges of wireless sensor networks, focusing on the long-lived and resource-constrained aspects of these systems.

One approach to interacting with a sensor network is to access the data according to a database model. As both the organization of the data and the type of query are different than in traditional databases, however, several challenges exist. From the organizational perspective, traditional distributed databases have few data locations, each with either the entire data set replicated or with a large portion of a non-replicated data store. In sensor networks, each individual sensor is a source for its own data only, yielding a large number of data sources, each providing minimal information. Further, queries for sensor networks tend to be long lived (e.g., for the time interval dawn to dusk, return the temperature of an area every ten minutes). In contrast, traditional queries are most often performed to retrieve current information a single time. Many approaches have been taken to adapt sensor network queries to fit this paradigm.

Both the Cougar [9] and SINA [10] systems provide a distributed database query interface to the information from a sensor network with an emphasis on power management. Cougar proactively manages power resources by distributing the query among the sensor nodes to minimize the energy consumed to collect the data and calculate the query result. SINA, on the other hand, incorporates low level mechanisms for hierarchical clustering of sensors for efficient data aggregation as well as mechanisms that limit the retransmission of similar information from geographically proximate sensor nodes.

QUASAR [11] provides the application with a means to query a sensor database with a Quality aware Query (QaQ). QaQs can express quality requirements as either set-based (e.g., find at least 90% of the sensors with temperature greater than 50 degrees) or value-based (e.g., estimate the average temperature within 1 degree). After the queries are expressed by the application, they are executed to meet the quality requirements at minimum cost.

TAG is a generic aggregation service for wireless sensor networks that minimizes the amount of messages transmitted during the execution of a query [12]. In contrast to standard database query execution techniques, in which all data is gathered by a central processor where the query is executed, TAG allows the query to be executed in a distributed fashion, greatly reducing the overall amount of traffic transmitted on the network. The standard SQL query types (`COUNT`, `AVERAGE`, `SUM`, `MIN`, `MAX`), as well as more sophisticated query types, are included in the service, although certain query types allow more energy savings than others. Time is divided into epochs for queries requiring values to be returned at multiple times. When a query is sent by some node (initially the root), the receiving nodes set their parents to be the sending node and establish an interval within the epoch (intervals may be set to a length of `EPOCH_DURATION`/d, where d represents the maximum depth of the aggregating tree) during which their eventual children should send their aggregates (this interval should be immediately prior to their sending interval).

TinyDB is a processing engine that runs Acquisitional Query Processing (ACQP) [13], providing an easy-to-use generic interface to the network through an enhanced SQL-like interface and enabling the execution of queries to be optimized at several levels. ACQP allows storage points containing windows of sensor data to be created so that queries over the data streams can be executed more easily. Such storage points may be beneficial, for example, in sliding window type queries (e.g., find the average temperature in a room over the previous hour once per minute). ACQP also supports queries that should be performed upon the occurrence of specific events as well as queries that allow sensor settings such as the sensing rate to be adapted to meet a certain required lifetime. Perhaps most importantly, ACQP provides optimization of the scheduling of sensing tasks as well as at the network layer. Since the energy consumption involved in the sensing of certain types of data is not negligible compared to the transmission costs of sending such packets, the scheduling of complex queries should be optimized in order to avoid unnecessary sensing tasks. ACQP optimizes this scheduling based on sensing costs and the expected selectivity of the query so as to minimize the expected power consumption during a query. Significant power savings can also be achieved by the ACQP's batching of event-based queries in some cases. The topology of an aggregating tree can also be optimized by considering the query in its formation. TinyDB uses Semantic Routing Trees (SRTs). Rather than requiring children to choose a parent node solely based on link quality, the choice of a parent nodes during the construction of an SRT also depends on the predicates of the query for which the tree is being built (i.e., the conditions that should be met for inclusion in the query). Specifically, children nodes choose a parent either to minimize the difference between their attributes of the predicate in the query or to minimize the spread of the attributes of the children of all potential parents. When a query is processed, a parent knows the attributes of all children and can choose not to forward the message if it determines that none of its children can contribute to the query (based on the query predicate and the attributes of its children).

Another approach to support event-centric communication is employed by EnviroTrack [14], a middleware for environmental tracking applications. EnviroTrack abstracts the low level collaboration of sensors around a certain event of interest by allowing the application to view the event as a logical object. When an event is detected, sensors surrounding the event form a cluster, to which a context label is associated. A tracking object is attached to each cluster, and the cluster performs local operations on the event of interest. Using EnviroTrack, the burden on the application is eased, as for each context label (i.e., event to be sensed), the application only needs to provide the conditions for an event to be sensed, a definition of the aggregate state, and any object code to be attached with the context label.

In some networks, abstraction at a lower level for managing a sensor node's neighborhood may be more appropriate, as neighborhood-based algorithms are typically simpler to design. Several programming primitives, such as Hood [15] and Abstract Regions [16], allow this level of abstraction and provide means necessary for establishing neighborhood membership, sharing data, caching data, and messaging.

The formation of neighborhoods is the basic premise of Hood. Each node may establish several neighborhoods, one for each different element of functionality (e.g., sensing, routing, localization, etc.), with the most important nearby nodes for each functionality comprising the members of the neighborhood. Within a neighborhood, data critical to the neighborhood's function is shared. Neighborhoods are implicitly determined through filtering operations, which scan incoming packets (containing "attributes" such as location, sensor readings, etc.) that have been broadcast by nearby nodes and determine if the broadcasting node is valuable to the functionality of the specific neighborhood. Thus, neighborhood membership is determined by individual nodes and is asymmetrical, as opposed to rigid grouping or hard clustering techniques. If the filter determines that the broadcasting node should be a member of the neighborhood, a local "mirror" will be created or updated, containing "reflections" (the attributes that are broadcast by the node), as well as "scribbles" (notes about the node that are locally generated). When a node updates an attribute of its own, it sends it to a push-policy module, broadcasting it to nearby nodes or using some other application-specific mechanism for advertising.

The Abstract Regions environment provides a neighborhood abstraction that is similar to that of Hood. A similar API is provided that includes the functionality necessary for neighborhood discovery, data sharing, and multiple forms of data aggregation. The forming of neighborhoods is a bit more proactive, as a programmer explicitly sends commands to establish a neighborhood based on one of several criteria (e.g., all nodes within N radio hops, k nearest nodes, etc.). Abstract Regions also allows the application developer to tune several network parameters to meet the quality necessary for the application. For example, if the data sharing is accomplished by broadcasting packets to neighbors, the abstraction allows parameters such as the number of rebroadcasts, which can determine the reliability of data transmission, to be set explicitly.

2.2 Dynamic Sensor Selection

To exploit potential energy savings in a wireless sensor network, care should be taken in the selection of the active nodes in the network and in the determination of the best operating mode (e.g., sampling rate, resolution settings) for each active node. When using an energy-efficient MAC protocol [17, 18], the average energy consumption of each node can be heavily dominated by the amount of traffic generated. Furthermore, deactivating sensors can reduce any congestion that may arise in low bandwidth networks [3].

Among the most commonly proposed applications for wireless sensor networks are those that require coverage of the entirety or a portion of a region where the sensors are deployed [19]. Coverage preserving protocols and algorithms have many potential applications, including intruder detection, biological or chemical agent detection, and fire detection. Also, these protocols and algorithms can be used in the initial stages of many target tracking applications, where a more detailed description or location estimate of a phenomenon is required only when a "tripwire" threshold is crossed in the measurements of some of the active sensors. Several protocols have been proposed to select which sensors to activate for these coverage applications.

PEAS

PEAS [20] is a protocol that was developed to provide consistent environmental coverage and robustness to unexpected node failures. Nodes begin in a sleeping state, from which they periodically enter a probing state. In the probing state, a sensor transmits a probe packet, to which its neighbors will reply after a random backoff time if they are within the desired probing range. If no replies are received by the probing node, the probing sensor will become active; otherwise, it will return to the sleep state. The probing range is chosen to meet the more stringent of the density requirements imposed by the sensing radius and the transmission radius. The probing rate of PEAS adapts to balance energy savings and robustness. Specifically, a low probing rate may lead to long delays before the network recovers following an unexpected node failure. On the other hand, a high probing rate may lead to expensive energy waste. In general, the probing rate of individual nodes increases as more node failures arise, so that a consistent recovery time can be expected.

Node Self Scheduling Algorithm

A node self scheduling algorithm for coverage preservation in sensor networks is presented in [21]. In this algorithm, a node measures its neighborhood redundancy as the union of the sectors (i.e., central angles) covered by neighboring sensors within the

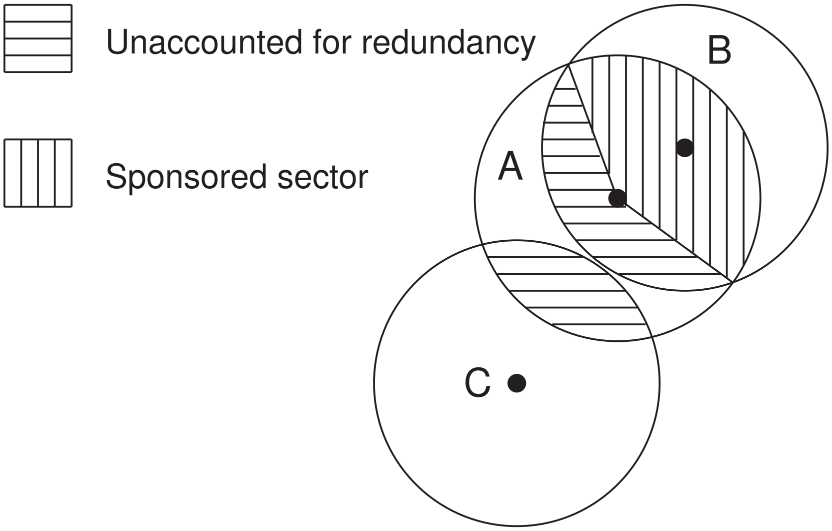


Figure 2.1: A sponsored sector, as defined by the node self scheduling algorithm [21]. Sensor A accounts for the redundant coverage of sensor B in the vertically shaded regions but not the additional redundancy of sensors B and C shown in the horizontally shaded regions.

node's sensing range. At decision time, if the union of a node's "sponsored" sectors covers the full $360°$ (see Figure 2.1), the node will decide to power off. It should be noted that additional redundancy may exist between sensors. The redundancy model is simplified at the cost of the inability to exploit this redundancy. At the beginning of each round, there is a short self-scheduling phase where nodes first exchange location information and then decide whether or not to turn off after some backoff time. Scenarios of unattended areas due to the simultaneous deactivation of nodes are avoided by requiring nodes to double check their eligibility to turn off after making the decision.

Reference Time-based Scheduling Scheme

In the reference time-based scheduling scheme presented in [22], the environment is divided into a grid and coverage is maintained continuously at every grid point while minimizing the number of active sensors. During an initialization process, each node broadcasts a randomly chosen reference time uniformly distributed on $[0, T)$, where T is the round length, to all neighboring sensors within twice its sensing radius. For each location in the grid that the sensor is capable of monitoring, a sensor sorts the reference

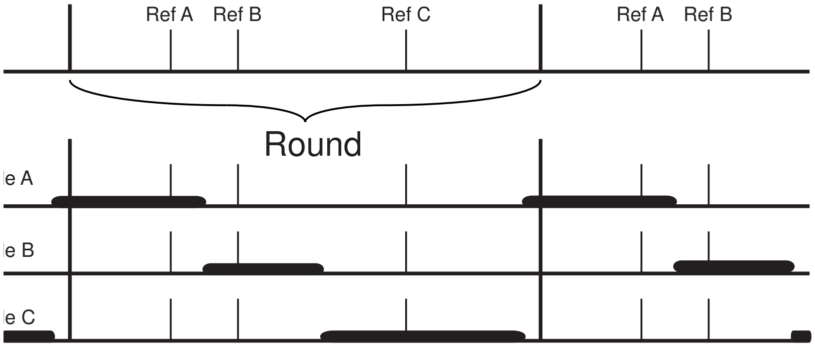
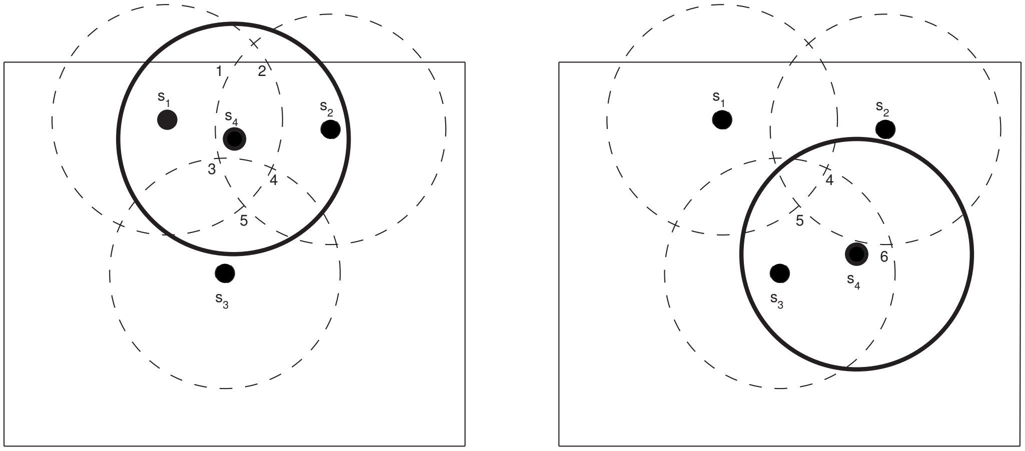


Figure 2.2: Schedule calculation for a single grid point, as proposed in the reference time-based scheduling scheme [22]. After all grid point schedules are calculated, the schedules are merged and a sensor's overall schedule is the union of all of its grid point schedules.

times of all sensors capable of monitoring that grid point. For a given grid point, the sensor schedules itself to be active beginning halfway between its reference time and the reference time of the sensor immediately preceding it in the sorted list. Similarly, its scheduled slot for the grid point ends halfway between its reference time and the reference time of the sensor immediately after it in the sorted list (see Figure 2.2). The sensor remains active during the union of the scheduled slots calculated for each grid point within its sensing range. This algorithm is also enhanced to guarantee coverage by multiple sensors in selected areas as well as provide robustness to node failures.

CCP

In CCP [23], an eligibility rule is proposed to maintain a certain degree of coverage. First, each node finds all intersection points between the borders of its neighbors' sensing radii and any edges in the desired coverage area. The CCP rule assigns a node as eligible for deactivation if each of these intersection points is K-covered, where K is the desired sensing degree. The CCP scheme assumes a Span-like protocol and state machine that can use the Span rule for network connectivity or the proposed CCP rule for K-coverage, depending on the application requirements and the relative values of the communication radius and sensing radius. An example of how the CCP rule is applied is given in Figure 2.2. In Figure 2.2(a), node s_4, whose sensing range is repre-



(a) (b)

Figure 2.3: The CCP activation rule for K-coverage, K = 1 [23]. In each scenario, node s_4 decides whether or not to activate, knowing that neighbors s_1, s_2, and s_3, are already active. In (a), node s_4 may remain inactive since all of its intersection points are K-covered. However, in (b), s_4 must become active since intersection point 6 is not covered by any of its neighbors.

sented by the bold circle, must decide whether it should become active in order to meet a coverage constraint of $K = 1$. It is assumed that D knows that s_1, s_2, and s_3, whose sensing ranges are represented by the dashed circles, are currently active. The intersection points within s_4's sensing range are found and enumerated 1-5 in the figure. Since s_2 covers points 1 and 3, s_3 covers points 2 and 4, and s_1 covers point 5, s_4 deduces that the coverage requirements have already been met and remains inactive. In the scenario illustrated in Figure 2.2(b), there is an intersection point (labeled 6 in the figure) that is not covered by any of s_4's neighbors. Thus, s_4 must become active and sense the environment.

Connected Sensor Cover

The Connected Sensor Cover algorithm [24] provides a joint solution for topology control and sensor selection. Specifically, Connected Sensor Cover finds a minimum set of sensors and additional routing nodes necessary in order to efficiently process a query over a given geographical region. In the centralized version of the algorithm, an initial sensor within the query region is randomly chosen, following which additional sensors are added by means of a greedy algorithm. At each step in this algorithm, all sensors that redundantly cover some area that is already covered by the current active subset are considered candidate sensors and calculate the shortest path to one of the sensors already included in the current active subset. For each of these candidate sensors, a heuristic is calculated based on the number of unique sections in the query region that the sensor and its routers would potentially add and the number of sensors on its calculated path. The sensor with the most desirable heuristic value and those along its path are selected for inclusion in the sensor set. This process continues until the query region is entirely covered. The algorithm has been extended to account for node weighting, so that low energy nodes can be avoided, and to be implemented through distributed means, with little loss in solution optimality compared with the centralized version.

2.3 Routing in Ad Hoc and Sensor Networks

2.3.1 Resource-Aware Routing

The field of ad hoc routing has been explored extensively. Initially, protocol design focused on efficiently finding shortest path routes in the presence of node mobility [25]. Recently, because of the scarce energy supplies available in sensor networks, a great deal of effort has been put forth in creating energy-aware routing protocols that consider the energy resources available at each sensor.

Singh et al. were among the first to develop energy-aware routing protocols and proposed several routing costs that cause nodes with scarce energy resources to be avoided [26]. They proposed that the lifetime of the network could be extended by minimizing the cumulative cost c_j of a packet j being sent from node n_1 to node n_k through intermediate nodes n_2, n_3, etc., where

$$c_j = \sum_{i=1}^{k-1} f_i(z_i) \tag{2.1}$$

$$f_i(z_i) = \frac{1}{1 - g(z_i)} \tag{2.2}$$

and $g(z_i)$ represents the normalized remaining lifetime corresponding to node n_i's battery level z_i. Further work by Chang et al. solved the problem of maximizing network lifetime by finding an optimal energy-aware routing cost that considered transmission energy as well as residual energy [27]. In their work, the routing cost of sending a packet was the sum of the routing costs of the individual links. The cost c_{ij} of a link between node i and node j was set to

$$c_{ij} = e_{ij}^{x_1} \underline{E}_i^{-x_2} E_i^{x_3} \tag{2.3}$$

where e_{ij} represents the energy necessary to transmit from node i to node j, $\underline{E}_i$ represents the residual energy of node i, and E_i represents the initial energy of node i. Brute force simulation methods were used to find the optimal values of x_1, x_2, and x_3.

Shah et al.

From the intuition that can be taken from this initial work, several energy-aware routing protocols have been developed for sensor networks, including the one proposed by Shah et al. [28]. In this protocol, interest queries are sent from an agent by way of controlled flooding toward the source node(s). The cost $Cost(N_i)$ associated with each node N_i indicates the node's reluctance to forward messages. Each upstream neighbor N_j of node N_i calculates a link cost C_{N_j,N_i} associated with N_i that depends on $Cost(N_i)$ as well as the energy e_{ij} required to transmit over this link and the normalized residual energy R_i at node N_i such that

$$C_{N_j,N_i} = Cost(N_i) + e_{ij}^{\alpha} R_i^{\beta} \tag{2.4}$$

where α and β are tunable parameters. Each node N_j builds a forwarding table FT_j consisting of its lowest cost downstream neighbors and the link cost C_{N_j,N_i} associated with those neighbors. Node N_j assigns a probability P_{N_j,N_i} to each neighbor as

$$P_{N_j,N_i} = \frac{1/C_{N_j,N_i}}{\sum_{k \in FT_j} 1/C_{N_j,N_k}} \tag{2.5}$$

Received messages are forwarded over each link with this probability. Before forwarding its message, N_j must determine its own value of $Cost(N_j)$, which is simply the weighted average of the costs in its forwarding table FT_j

$$Cost(N_j) = \sum_{i \in FT_j} P_{N_j,N_i} C_{N_j,N_i} \tag{2.6}$$

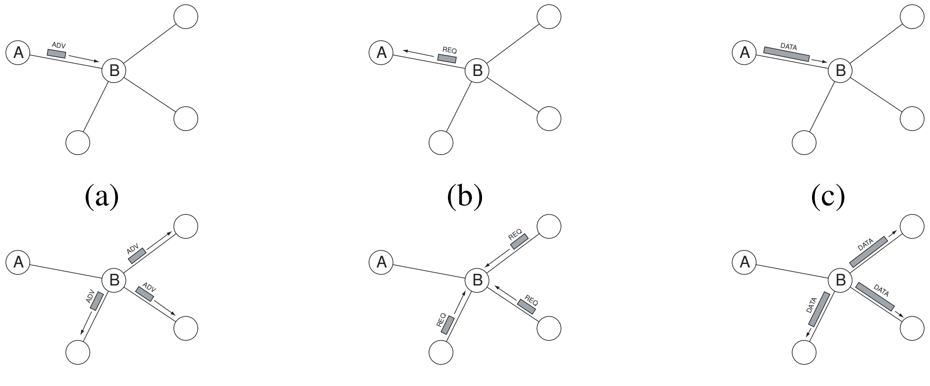
2.3.2 Data-Centric Routing

Sensor networks are fundamentally different from ad hoc networks in the data they carry. While individual data items are important in ad hoc networks, it is the aggregate data or the information carried in the data rather than the actual data itself that is important in sensor networks. This has led to a new paradigm for networking these types of devices – data-centric routing. In data-centric routing, the end nodes (i.e., the sensors themselves) are less important than the data itself. Thus, queries are posed for specific data rather than for data from a particular sensor, and routing is performed using knowledge that it is the aggregate data rather than any individual data item that is important.

SPIN

SPIN (Sensor Protocol for Information via Negotiation) is a protocol that was designed to enable data-centric information dissemination in sensor networks [29]. Rather than blindly broadcasting sensor data throughout the network, nodes receiving or generating data first advertise this data through short ADV messages. The ADV messages simply consist of an application-specific meta-data description of the data itself. This meta-data can describe such aspects as the type of data and the location of its origin. Nodes that are interested in this data request the data from the ADV sender through REQ messages. Finally, the data is disseminated to the interested nodes through DATA messages that contain the data. This procedure is illustrated in Figure 2.4.

The advantage of SPIN over blind flooding or gossiping data dissemination methods is that it avoids three costly problems: implosion, overlap and resource blindness. Implosion occurs in highly connected networks that employ flooding. In these scenarios, each sensor receives many redundant copies of the data, as illustrated in Figure 2.5(a). For large data messages, this wastes considerable energy. In SPIN, on the other hand, short ADV messages will suffer from the implosion problem, but the costly transfer of data messages is greatly reduced. The overlap problem occurs as a result of the redundant nature of sensor data. Thus, two sensors with some common data will both send



(a) (b) (c)

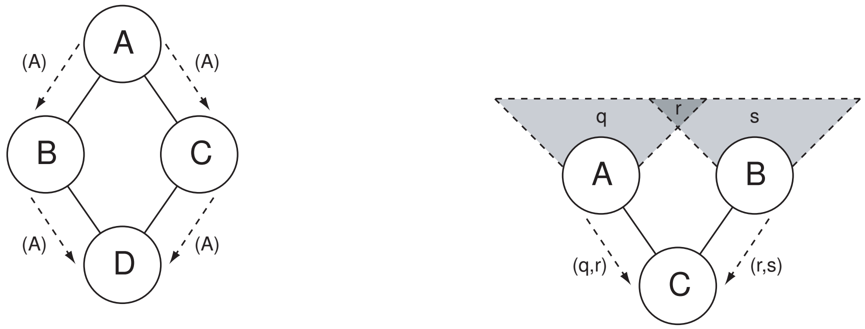
(d) (e) (f)

Figure 2.4: Message exchange in the SPIN protocol [29]. Nodes advertise their data with ADV messages (a). Any node interested in receiving the data replies with a REQ message (b), to which the source node replies with the transmission of the actual data (c). The receiving node then advertises this new data (d) and the processes continues (e,f).

their data, causing redundancy in data transmission and thus energy waste, as illustrated in Figure 2.5(b). SPIN is able to solve this problem by naming data so that sensors only request the data or parts of data that they are interested in receiving. Finally, in SPIN, there are mechanisms implemented so that a sensor that is running low on energy will not advertise its data in order to save its energy resources. Thus, SPIN solves the resource blindness problem by having sensors make decisions based on the current level of available resources.

Directed Diffusion

Directed Diffusion is a communication paradigm that has been designed to enable data-centric communication in wireless sensor networks [30]. To perform a sensing task, a querying node creates an interest, which is named according to the attributes of the data or events to be sensed. When an interest is created, the sink node injects it into the network by broadcasting an interest message containing the interest type, duration, and an initial reporting rate to all neighbors. An example of an interest is the number of people in a given area every second for the next 10 minutes. Local interest caches at each node contain entries for each interest that has been created on the network and



(a)

(b)

Figure 2.5: Problems with blind flooding of sensor data. (a) Implosion occurs in a highly connected network where nodes receive duplicate copies of data, wasting energy and bandwidth resources. As seen in this figure, node D receives two copies of node A's data. (b) Overlap occurs due to the redundant nature of sensor data. This figure shows that C receives data about region r from nodes A and B, again wasting valuable sensor resources.

reached the node. An entry in the cache contains information about the interest's type, duration, and gradient (a combination of the event rate and direction toward the data sink). Nodes receiving the interest messages find (or create) the relevant interest entry in their caches and update the gradient field toward the node from which the message was received, setting it to the rate defined in the interest message. Each gradient also has information regarding the expiration time, which must be updated upon the reception of the interest messages.

Interests are diffused throughout the network toward the sink node using one of a number of forwarding techniques. For example, Figure 2.6 shows a network in which the interest was sent to the region of interest via controlled flooding. Once the interest reaches the desired region, sensor nodes within the region process the query and begin producing data at the specified rate. If more than one entry for the same interest type exist, data is produced at the maximum rate of these entries. Data pertaining to these interests are then forwarded to each node for which a gradient exists at the rate specified for each individual gradient. After receiving low rate events from the source (recall that the initial reporting rate is set low), the data sink may reinforce higher quality paths. These reinforced paths might be chosen as those that experience low latency

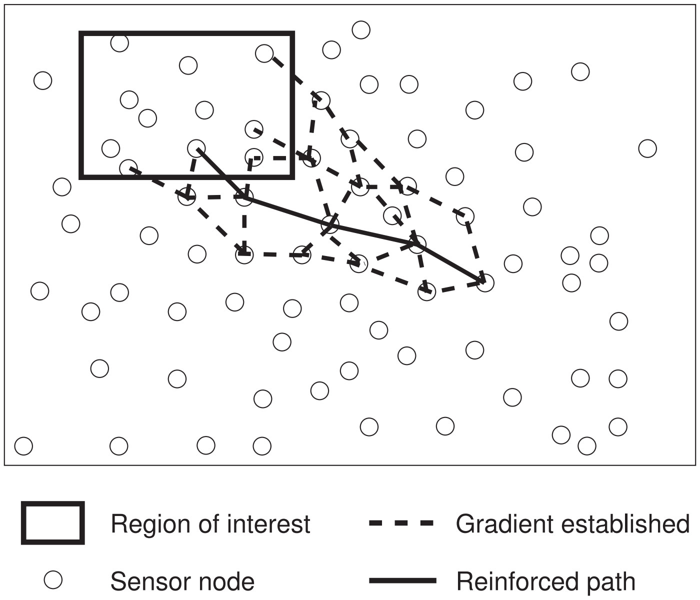


Figure 2.6: Establishing gradients in Directed Diffusion [30]. As the query is routed toward the region of interest, gradients for that interest are established in the reverse direction of the query dissemination. After data begins to arrive at the querying node, the path of highest quality is reinforced.

or those in which the confidence in the received data is deemed to be high by some application-specific measure. Reinforcement messages simply consist of the original interest messages, with the reporting rates set higher. These reinforced routes are established more conservatively than the original low rate interest messages so that only a single or few paths from the event to the sink are used.

Rumor Routing

While long-lived queries or data flows justify the overhead involved in establishing cost fields in a network, it may not be worth this effort when executing short-lived and one-shot queries. Rumor routing was designed for these types of queries [31]. When an event is detected by a sensor, it probabilistically creates an agent in the form of a data packet, and forwards it throughout the network in a random manner (illustrated by the solid line in Figure 2.7). Nodes through which the agent is forwarded maintain local state information about the direction and distance to the event. Should an agent

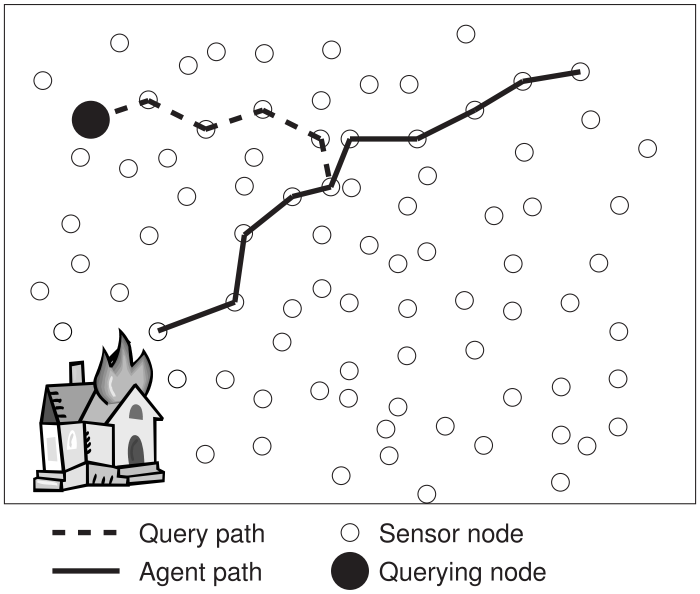


Figure 2.7: Query handling in Rumor Routing [31]. After an event is detected, and agent is initiated and sent on a random path through the network, establishing state at each node on the path. A query packet is similarly sent in a random direction and hopefully crosses paths with the agent, allowing the query to be answered and returned to the querying node.

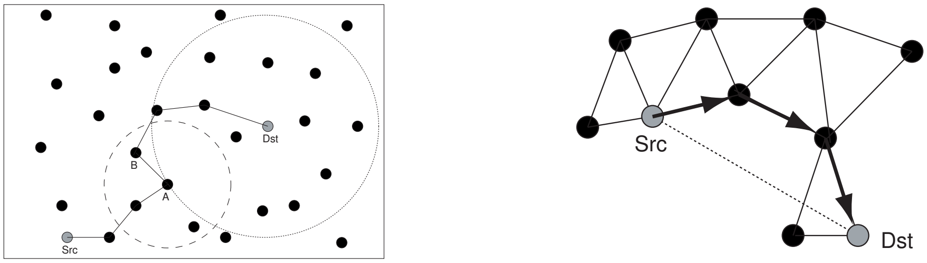
traverse a node with knowledge of a path to other events, it adds this information so that subsequent nodes through which the agent flows will maintain state information regarding these events as well. When a node wishes to perform a query related to a given event, it simply forwards a query packet in a random direction so that the query traverses a random walk throughout the network (illustrated by the dashed line in Figure 2.7). Because of the fact that two lines drawn through a given area are likely to cross, there is a high likelihood that the query will eventually reach a node with a path to the specified event, especially if multiple agents carrying that event are sent through the network. If multiple queries happen not to reach the event, the querying node may resort to flooding queries over the entire network.

2.3.3 Geographic Routing

Often times, wireless sensor networks require a query packet to be forwarded to a particular region of interest in the network. A natural approach to perform this forwarding is to utilize geographic forwarding. Geographic forwarding reduces the amount of routing overhead, which is typically dominated by route discovery, and requires less memory utilization for route caching than typical ad hoc routing protocols. Furthermore, geographic routing protocols can enable geographically distributed data storage techniques such as Geographic Hash Tables (GHT) [32].

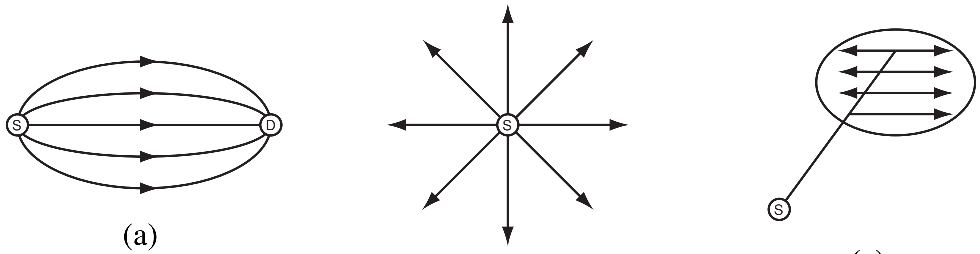
GPSR

GPSR (Greedy Perimeter Stateless Routing) is a geographic routing protocol in which nodes make local packet forwarding decisions according to a greedy algorithm [33]. Under normal circumstances, a packet that is destined for some node D is forwarded to the node's neighbor that enables the maximum progress toward D. A similar greedy forwarding scheme was originally proposed in the work of Takagi and Kleinrock [34]. However, obstacles or a lack of adequate sensor density can cause voids in the network topology so that packets reach a hole, at which point the packet cannot progress any further without first being sent backward. GPSR accounts for this by incorporating a perimeter routing mechanism. Voids can be detected by the nodes surrounding them, and routes that circumnavigate the voids can be established heuristically. When a packet reaches these voids, these routes can be used (routing by the right hand rule) until normal greedy routing can resume. This process is illustrated in Figure 2.8(a). While this approach works well, another more robust perimeter routing algorithm is also proposed. In this algorithm, the graph that can be drawn from the complete network topology is first reduced to a planar graph in which no edges cross. Once a packet reaches a void, the forwarding node N finds the face of the planar graph that is intersected by the line connecting N and the destination, as illustrated in Figure 2.8(b). N then forwards the packet to the node along the edge that borders this face. This procedure continues with each intermediate node finding the adjacent face that the line intersects and routing along an edge bordering that face until the void has been cleared.



(a) (b)

Figure 2.8: GPSR [33] greedy forwarding policy (a) and perimeter routing algorithm (b).



(a) (b) (c)

Figure 2.9: Possible trajectories to use in TBF [35] for robust multipath routing (a), spoke broadcasting (b), and broadcast within a remote region (c).

TBF

TBF (Trajectory Based Forwarding) is a useful paradigm for geographic routing in wireless sensor networks [35]. Rather than sending a packet along a straight path toward its destination, TBF allows packets to follow a source-specified trajectory, increasing the flexibility of an overall forwarding strategy. For example, multipath routing can be achieved by sending multiple copies of a single packet along separate geographic trajectories, as shown in Figure 2.9(a), increasing resilience to localized failures or congestion in certain parts of the network. Also, TBF can increase the efficiency of many different forwarding techniques, including spoke broadcasting, illustrated in Figure 2.9(b), and broadcast to a remote subregion, illustrated in Figure 2.9(c).

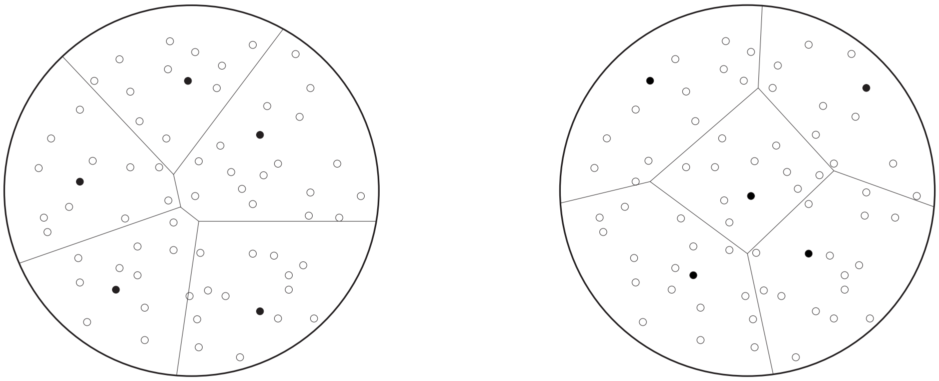


Figure 2.10: Adaptive clustering of the network.

2.3.4 Clustering Algorithms

As sensor networks are expected to scale to large numbers of nodes, protocol scalability is an important design criteria. If the sensors are managed directly by the base station, communication overhead, management delay, and management complexity become limiting factors in network performance. Clustering has been proposed by researchers to group a number of sensors, usually within a geographic neighborhood, to form a cluster that is managed by a cluster head. A fixed or adaptive approach may be used for cluster maintenance. In a fixed maintenance scheme, cluster membership does not change over time, whereas in adaptive clustering schemes, sensors may change their associations with different clusters over time, as illustrated by the two configurations in Figure 2.10.

Clustering provides a framework for resource management. It can support many important network features within a cluster, such as channel access and power control, as well as between clusters, such as routing and code separation to avoid inter-cluster interference. Moreover, clustering distributes the management responsibility from the base station to the cluster heads, and provides a convenient framework for data fusion, local decision-making and control, and energy savings.

LEACH

In-network processing can greatly reduce the overall power consumption of a sensor network when large amounts of redundancy exist between nearby nodes. Rather than requiring all sensors' data to be forwarded to a base station that is monitoring the environment, nodes within a region can collaborate and send only a single summarization packet for the region. This use of clustering was first introduced in the Low Energy Adaptive Clustering Hierarchy (LEACH) protocol [36]. In LEACH, nodes are divided into clusters, each containing a cluster head whose role is considerably more energy intensive than the rest of the nodes; for this reason, nodes rotate roles between cluster head and ordinary sensor throughout the lifetime of the network.

At the beginning of each round, each sensor node makes an independent decision through a randomized algorithm about whether or not to assume a cluster head role. Nodes that choose to be cluster heads announce their status to the rest of the network. Based on the received signal strength of these announcements, sensors join the cluster that requires the least power to communicate with the cluster head (assuming transmission power control is available). During the round, the ordinary sensors in each cluster send data to their respective cluster heads according to a time-division multiple access (TDMA) schedule. Inter-cluster interference is reduced using different spreading codes in neighboring clusters. The cluster head aggregates data from all the cluster members and sends the aggregate data to the base station. The length of each round is chosen such that each node is expected to be able to perform a cluster head role once during its lifetime.

Because there is no interaction between nodes when deciding roles, the cluster heads may be chosen such that there is no uniformity throughout the network and certain sensors are forced to join clusters located at large distances from them. To mitigate this problem, a centralized version of LEACH called LEACH-C has been developed. LEACH-C uses simulating annealing to choose the cluster heads for a given round so that the average transmission power between sensors and their cluster heads is minimized.

HEED

Nodes in LEACH independently decide to become cluster heads. While this approach requires no communication overhead, it has the drawback of not guaranteeing that the cluster head nodes are well distributed throughout the network. While the LEACH-C protocol solves this problem, it is a centralized approach that cannot scale to very large networks.

Many clustering algorithms have been proposed that create more uniform clusters at the expense of overhead in cluster formation. One approach that uses a quickly converging distributed algorithm is HEED (Hybrid Energy-Efficient Distributed Clustering) [37]. HEED uses an iterative cluster formation algorithm, where sensors assign themselves a cluster head probability that is a function of their residual energy and a communication cost that is a function of neighbor proximity. Using the cluster head probability, sensors decide whether or not to advertise that they are a candidate cluster head for the current iteration. Based on these advertisement messages, each sensor selects the candidate cluster head with the lowest communication cost (which could be the sensor itself) as its tentative cluster head. This procedure iterates, with each sensor increasing its cluster head probability at each iteration until the cluster head probability is one and the sensor declares itself a final cluster head for the round. HEED requires only local neighborhood information to form the clusters and the algorithm terminates in O(1) iterations. Furthermore, the algorithm guarantees that every sensors is part of just one cluster and that the cluster heads are well-distributed.

2.3.5 Topology Control

The purpose of traditional topology control protocols has been to balance two contradictory goals – reducing energy consumption while maintaining high network connectivity. Most early work in this area concentrated on adjusting radio settings (e.g., transmission power [38–41], beamforming patterns [42]) to maintain connectivity with an optimal set of neighbors. Because it is often more power-efficient to relay packets over several short hops than a single long hop, reducing transmission power is an effective means for reducing overall energy consumption. Reducing transmission power also allows the network to benefit from spatial reuse, possibly resulting in reduced congestion, higher throughput, and a reduction in the number of costly data packets that are unnecessarily overheard.

These methods may be very effective in sensor networks where energy consumption is dominated by the energy consumed in transmitting data packets. However, some typical power models considered for sensor networks show that receive power and idle power are comparable to transmit power [43]. Based on this observation, further savings can surely be achieved by not only reducing transmission power, but also setting the sensors' radios into a sleep state whenever possible. Below, several topology control protocols that achieve energy-efficiency through these means are described. While some of these protocols were originally designed for use in general ad hoc networks, the fact that nodes are often allowed to turn their radios off nevertheless makes them suitable protocols for sensor networks as well.

GAF

The Geographic Adaptive Fidelity (GAF) protocol [44] takes advantage of the fact that neighboring nodes are often nearly identical from the perspective of data routing. In GAF, a virtual grid is formed throughout the network, and each node is assigned to the virtual grid cell in which it resides. Only a single node from a cell in the virtual grid is chosen to be active at any given time, as illustrated in Figure 2.11. Nodes implementing GAF initially enter a discovery state, where they listen for messages from other nodes within their cell. If the node determines that a more suitable node can handle the routing responsibilities for its cell, it falls into a sleep state, from which it periodically reenters the discovery state; otherwise, it enters the active state and participates in data routing. After a predetermined active period, active nodes fall back into the discovery state. As the density of a network implementing GAF increases, the number of activated nodes per grid cell remains constant while the number of nodes per cell increases proportionally. Thus, GAF can extend lifetime approximately linearly as a function of node density.

Span

Span [45] is a topology control protocol that allows nodes that are not involved in a routing backbone to sleep for extended periods of time. In Span, certain nodes assign themselves the position of coordinator. These coordinator nodes are chosen to form a network backbone, such that the capacity of the backbone approaches the potential capacity of the complete network. Periodically, nodes that have not assigned themselves the coordinator role initiate a procedure to decide if they should become a coordinator. The criteria for this transition is if the minimum distance between any two of the node's neighbors exceeds three hops. To avoid the situation where many nodes simulta-

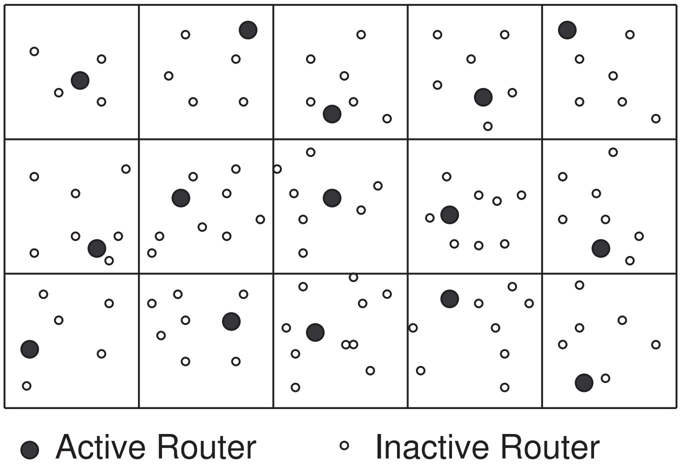


Figure 2.11: Example of a GAF virtual grid [44]. Only one node per cell is activated as a router.

neously decide to become coordinator, backoff delays are added to nodes' coordinator announcement messages. The backoff delays are chosen such that nodes with higher remaining energy and those potentially providing more connectivity in their neighborhood are more likely to become a coordinator. To ensure a balance in energy consumption among the nodes in the network, coordinator nodes may retreat from their coordinator role if neighboring nodes can make up for the lost connectivity in the region.

ASCENT

ASCENT (Adaptive Self-Configuring sEnsor Networks Topologies) [46] is similar to Span in that certain nodes are chosen to remain active as routers while others are allowed to conserve energy in a sleep state. In ASCENT, the decision to become an active router is based not only on neighborhood connectivity, but also on observed data loss rates, providing the network with the ability to trade energy consumption for communication reliability. Nodes running the ASCENT protocol initially enter a test state where they actively participate in data routing, probe the channel to discover neighboring sensors and learn about data loss rates, and send their own "Neighborhood Announcement" messages. If, based on the current number of neighbors and current data loss rates, the sensor decides that its activation would be beneficial to the network, it becomes active and remains so permanently. If the sensor decides not to become active, it falls into a passive state, where it gathers the same information as it does in the test state, as well as any "Help" messages from neighboring sensors experiencing poor communication links, but does not actively participate in data routing. From this state, the node may reenter the test state if the information gathered indicates poor neighborhood communication quality, or enter the sleep state, turning its radio off and saving energy. The node periodically leaves the sleep state to listen to the channel from the passive state.

EAD

Energy-aware data centric routing (EAD) [47] is an algorithm for constructing a minimum connected dominating set among the sensors in the network, prioritizing nodes so that those with the highest residual energy are most likely to be chosen as non-leaf (i.e., actively routing) nodes. To establish a broadcast tree, control messages, of the form {`type` (undefined, leaf node, or non-leaf node), `level` (in the broadcast tree), `parent`, `residual energy`}, are disseminated throughout the network, starting with the data sink. During the establishment of the tree, each node with an undefined status (the default starting state) listens for control messages. If an undefined node v receives a message from a leaf node, the undefined node becomes a non-leaf node and prepares to send a message announcing its non-leaf status after some backoff time T_1^v. On the other hand, if an undefined node (or a non-leaf node) receives a message from a non-leaf node, it becomes a leaf node and prepares to announce its status after a delay of T_2^v. If a node at any time receives a message from a non-leaf node indicating that it is that node's parent, it immediately becomes a non-leaf node and broadcasts its status via a control packet. To ensure that nodes with more residual energy are more likely to assume the more energy intensive non-leaf roles, T_1^v and T_2^v should be monotonically decreasing functions of the residual energy. Also, the minimum possible value of T_1^v should be larger than the maximum possible value T_2^v so that the resulting set of non-leaf nodes is of minimal size.

2.3.6 Multicasting

The establishment of optimal multicast trees is a problem that has been studied extensively in the networking community [48]. Traditionally, optimal broadcasting and multicasting have been thought of as the minimization of the edge weights consisting of the multicast trees (i.e., the formation of minimum spanning trees and Steiner trees, respectively). This model is an accurate one for wired networks; however, in wireless networks, nodes can exploit the inherent “wireless multicast advantage” to further reduce multicast tree cost. More recent algorithms for setting up minimum cost multicast trees take advantage of the fact that a single node’s transmission can be received by more than just a single intended destination. Thus, these algorithms can be thought of as node-based, rather that link-based minimization algorithms.

BIP

The Broadcast Incremental Protocol (BIP) is a broadcasting protocol that takes advantage of the aforementioned “wireless multicast advantage” to generate low-cost multicast trees in wireless networks [49]. The BIP protocol is based on Prim’s algorithm for finding minimum spanning trees (MST). However, when determining the node with the minimum cost, BIP considers the incremental cost to join the tree. In other words, if a node is already transmitting at a certain power level to reach some neighbor, its incremental cost is the additional cost required to increase its transmission power to the level necessary to reach the additional node. Following construction of the broadcast tree, a “sweeping” operation is performed on the network in order to remove unnecessary transmissions.

The MIP protocol is based on the BIP protocol and uses a pruning procedure to remove unnecessary transmissions. After establishment of the broadcast tree, links that are not necessary to reach the multicast group members are removed. Each node transmits at the minimum power level required to reach each of its neighbors on the remaining outbound links.

2.3.7 Distributed Multicasting

Following the development of the BIP and MIP protocols, attempts have been made to distribute protocols for multicasting in a wireless environment. In this section, we review several of these protocols

Dist-BIP and LBIP

Dist-BIP was developed by the original authors of BIP as a means to approximate the centralized version of their protocol while using a distributed version [50]. There are two versions of Dist-BIP – Dist-BIP-A, which is a slightly better approximation to the centralized version, and Dist-BIP-G, which requires less overhead.

In Dist-BIP-G, selected gateway nodes form local broadcast trees using knowledge of their first and second neighborhood only. Dist-BIP-A is similar to Dist-BIP-G, except that all nodes, not just those selected as gateways by their parents, compute their local broadcast tree. As in the centralized version, both versions of Dist-MIP are identical to their Dist-BIP counterparts in the early stages, followed by a pruning procedure, which is easily distributable, to reduce unnecessary transmissions.

LBIP (Localized Broadcast Incremental Protocol) is very similar to Dist-BIP-G in that gateways are selected by their parents, and nodes calculate their local broadcast tree using two-hop neighborhood knowledge. The key difference is that rather than recalculating transmission power for themselves and other nodes in their parents' broadcast trees, the gateways use the transmission power selected for them by their parents.

EWMA

Embedded Wireless Multicast Advantage (EWMA) is a heuristic algorithm for generating approximations to the minimum energy broadcast tree [51]. EWMA begins by forming the link-based minimum spanning tree (MST). Once the initial spanning tree is set up, the covered set C, transmitting set F, and excluded set E are set to the source node, null, and null, respectively. Next, nodes from $(C - F) - E$ are analyzed for potential gains that can be achieved by increasing their transmission power to the levels that will reach the children of all transmitting nodes. Basically, the gain is given as the energy that will be saved by increasing the transmission power to save power at other nodes. The node with the largest gain is selected for inclusion in the set F and the nodes that are saved from transmitting are added to E. All nodes that are covered by the increase in transmission power are added to C. This process continues until all nodes are covered (i.e., included in C). The authors propose a round-based distributed version of this protocol.

2.4 Transmission Range Optimization

The minimization of transmission energy in wireless sensor networks, and in wireless networks in general, has been studied extensively. If transmission power cannot be adjusted, power consumption can be minimized by minimizing the number of hops between the source and destination. However, when transmission power can be set according to the distance over which data is being transmitted, because received energy typically falls off with distance as $1/d^2$, it may be more energy-efficient to send data over many short hops rather than fewer long hops. Several works have noted this and shown how to minimize energy consumption by appropriately setting transmission power. Takagi and Kleinrock explored how to best set transmission power in order to minimize interference and maximize throughput [34]. The problem of setting transmission power to a minimal level that will allow a network to remain connected has been considered in several studies [39, 40]. In later work, some considered the importance of a fixed energy consumption per bit, independent of transmission distance. Because of this overhead, there exists an optimal nonzero transmission range, at which energy-efficiency is optimized [52, 53].

In the above-cited works, the goal was to minimize overall energy consumption, and a fixed network-wide transmission range was assumed. However, using such schemes may result in extremely unbalanced energy consumption among the nodes in sensor networks characterized by many-to-one traffic patterns. In addition to minimizing energy consumption, it may also be beneficial to distribute the energy among the nodes and to favor using those with greater energy resources, so that network lifetime may be maximized. Note that network lifetime may be defined a number of ways, including time until the first node dies, time when the first region of a sensor network is left unattended, etc. To accomplish this goal of lifetime maximization, load balancing through a combination of intelligent routing and transmission power control was studied in [26], where several heuristic routing costs were recommended for use in order to minimize and at the same time balance energy consumption. In [27], Chang and Tassiulas show how the optimal combination of several routing costs allows network lifetime to be extended. In [54], Efthymiou et al. show how energy consumption can be balanced by distributing packets over several paths. The problem of finding the optimal routing to achieve maximum network lifetime in a sensor network was studied as a constrained linear program optimization in [55–57]. In this work, the authors find the maximum lifetime that could be achieved by any routing cost or balancing scheme.

2.5 Sensor Deployment Strategies

Aside from transmission range optimization for balancing energy across the sensors, several other sensor deployment strategies have been proposed to extend network lifetime. For example, a mobile data sink roaming within the network can be deployed to balance the energy consumption. In [58], data mules are deployed in the network to pick up the data once they are close to the data source. Buffer requirements are the main focus of this study. In [59], Kim et al. focus on minimizing the cost for topology maintenance and communication between the mobile sinks and the data sources. In [60], the optimal sink mobility strategy is studied. Our generalized model is able to obtain the optimal assignment of communication load for the mobile sink strategy, and our study focuses on the network lifetime improvement from this strategy. Therefore, detailed design considerations such as buffer size and the overhead for network maintenance are not considered here.

Multiple data sinks can also be deployed to collect data over a certain subregion of the entire area. In [61], the optimal assignment of communication load to multiple sinks is found using a method similar to electrostatic theory. In [62], an application using multiple Crossbow Stargates as virtual data sinks is implemented. Further deployment strategies that integrate data aggregation have also been considered. In LEACH [36], each sensor can serve as a cluster head, where data from neighboring sensors is aggregated, and sensors rotate their roles to evenly distribute the energy load. This can be considered a multiple sink strategy with data aggregation.

The deployment of extra relay nodes around the data sink can also be helpful in solving energy imbalance problems. In [63], Ergen and Varaiya compare the minimum energy consumption when the relay nodes' locations are predetermined, and when they can be placed in any location. The authors provide a heuristic method to solve the latter problem. In [64], a similar mixed-integer nonlinear programming solution is provided to discover the optimal locations of relay nodes iteratively. In [65], Howitt and Wang attempt to balance energy consumption by requiring the sensors to send traffic to the next node along a chain to the base station and spacing sensors non-uniformly as a function of their distance to the data sink so that energy consumption is uniform for all nodes.

Chapter 3

A Sensor Network Framework and Architecture

Many protocols designed for wireless sensor networks have been designed to provide high quality data (in terms of low latency, high resolution, etc.) to the application to meet the application's quality of service (QoS) demands. Many of these protocols contain parameters that can be set to meet application goals. For example, depending on the spatial bandwidth of a phenomenon being measured or the distance at which a phenomenon can be measured, the nominal sensing range that is used when activating sensors may be set so that a threshold on the probability of a missed detection — a requirement imposed by the application — is not exceeded. Another example involves many MAC protocols, which can tune their duty cycle or other parameters so that packet delays are bounded. Later chapters in this thesis describe protocols that have been developed to meet these goals. Before these are presented, we present a framework in which these protocols can operate. Adapting these protocol parameters typically involves a tradeoff between application quality and energy-efficiency or network lifetime. In this chapter, we address the problem of managing data quality within an application and across applications operating on a single network from a systems perspective, and we propose a middleware architecture for wireless sensor networks to enable protocol parameters adaptation to meet application goals.

3.1 Middleware Architecture

Oftentimes, a combination of characteristics from the environment and characteristics of the application drive the design of middleware. With the advent of large-scale wireless sensor networks comes a new class of applications containing the following features.

- *Inherent distribution.* The sensors are distributed throughout a physical space, and are primarily connected via wireless links.
- *Dynamic availability of data sources.* Either mobility through space, addition of new sensors, or loss of existing sensors causes the set of available sensors to change over time.
- *Constrained application quality of service demands.* Sensor network applications require a minimum reliability, or QoS, and this level must be maintained over an extended period of time. There may be many ways to achieve the QoS (e.g., different sensors may offer data or services that meet the applications' QoS requirements). Furthermore, the required QoS and the means of meeting this QoS can change over time, as the state of the application or the availability of sensors changes.
- *Resource limitations.* Network bandwidth and sensor energy are constrained. This is especially true when considering battery powered sensors and wireless networks.
- *Cooperative applications.* Sensor network applications share available resources (e.g., sensor energy, channel bandwidth, etc.) and either cooperate to achieve a single goal, or, at the very least, do not compete for these limited resources.
- *Data-driven applications.* The applications collect and analyze data from the environment, and depending on redundancy, noise, and properties of the sensors themselves, the applications can assign a quality level to the data.
- *State-based applications.* The application's needs with respect to sensor data can change over time based on previously received data.

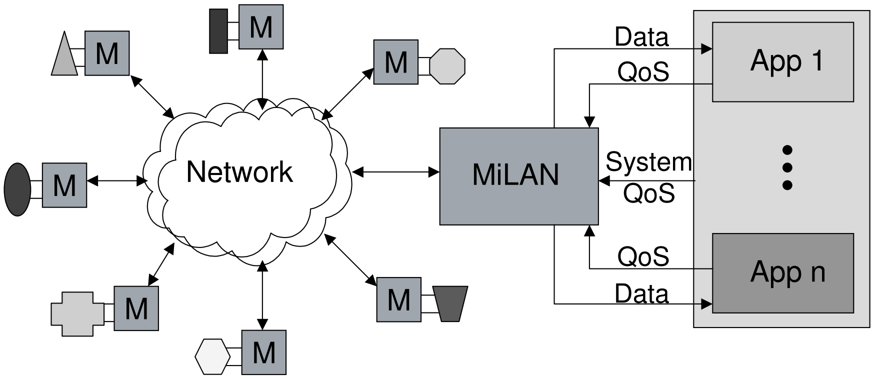


Figure 3.1: System that employs MiLAN. Each sensor runs a (possibly scaled-down) version of MiLAN. MiLAN receives information from applications about their QoS requirements, a system user about the desired interaction among the applications, and the network about available components and resources. MiLAN then decides how best to configure the network to support the applications.

Thus, wireless sensor network applications require new middleware designs to meet these unique features.

Collaboration at the University of Rochester's Center for Future Health has led to the development of a middleware architecture specifically designed for wireless sensor networks named MiLAN (Middleware Linking Applications and Networks). This middleware allows an application programmer to specify high level QoS goals through a simple application programming interface (API) and configures sensor roles according to the quality of information that the sensors can provide to a given application. Specifically, MiLAN receives information from (1) the individual applications about their QoS requirements over time and how to meet these QoS requirements using different combinations of sensors, (2) the overall system about the relative importance of the different applications, and (3) the network about available sensors and resources such as sensor energy and channel bandwidth. Combining this information, MiLAN continuously adapts the network configuration (e.g., specifying which sensors should send data, which sensors should be routers in multi-hop networks, which sensors should play special roles in the network, etc.) to meet the applications' needs while maximizing application lifetime. Figure 3.1 shows a high-level diagram of a system that employs MiLAN.

The MiLAN middleware architecture enables control of the network functionality while hiding network complexities from the application programmer. MiLAN receives a description of application requirements, monitors network conditions, and optimizes sensor and network configurations to maximize application lifetime. To accomplish these goals, applications represent their requirements to MiLAN through specialized graphs that incorporate state-based changes in application needs. Based on this information, MiLAN makes decisions about how to control the network as well as the sensors themselves to balance application QoS and energy-efficiency, lengthening the lifetime of the application.

Unlike traditional middleware that sits between the application and the operating system, MiLAN has an architecture that extends into the network protocol stack, as shown in Figure 3.2. As MiLAN is intended to sit on top of multiple physical networks, an abstraction layer is provided that allows network specific plug-ins to convert MiLAN commands to protocol-specific commands that are passed through the usual network protocol stack. Therefore, MiLAN can continuously adapt to the specific features of whichever network protocols are being used for communication (e.g., determining scatternet formations in Bluetooth networks [66], coordinator roles in Span [45], etc.) in order to best meet the applications' needs over time.

Figure 3.3 shows an overview of the interactions among MiLAN, the applications, and the sensors. This figure makes a distinction between the network plug-ins and the core of MiLAN, emphasizing the separation of computation that is specific to the selected network type versus the computation that always occurs, but the API specifies only the application and sensor level operations.

3.1.1 Application Performance

Many sensor network applications are designed to receive data input from multiple sensors and to adapt as the available sensors change over time, either as new sensors come within range or as sensors go offline when they move away or run out of energy. We assume that application performance can be described by the reliability of different variables of interest to the application, where the reliability of the different variables depends on which sensors provide data to the application. For example, in a personal health monitor application, variables such as blood pressure, respiratory rate,

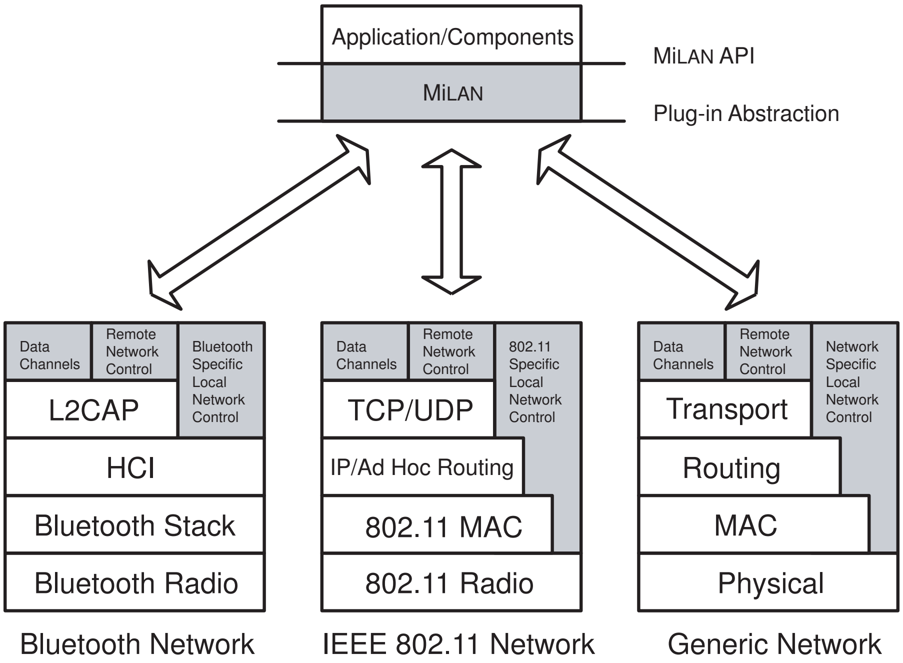


Figure 3.2: MiLAN components (shaded). MiLAN also presents an abstraction from the network-level functionality through which it issues commands to determine available sensors and configure the network. The network-specific plug-ins convert MiLAN commands to the network protocol employed. The shaded sections of the figure show where MiLAN and its plug-ins reside.

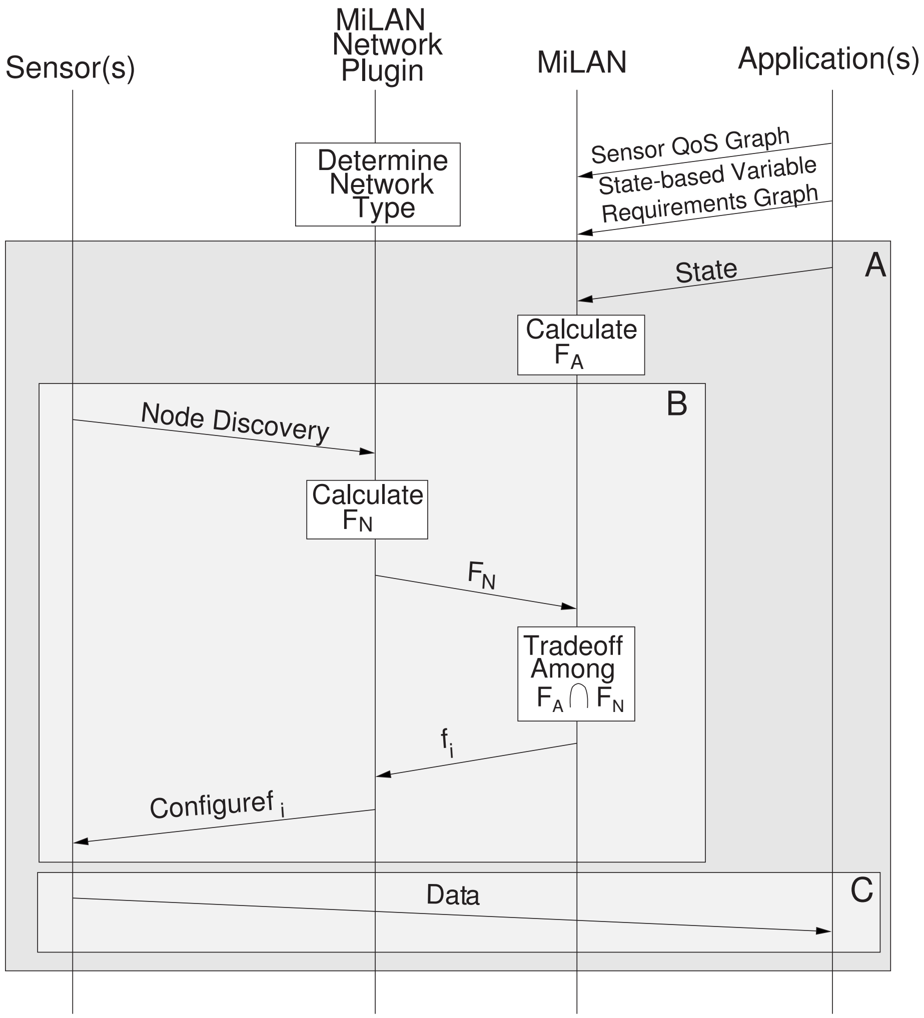


Figure 3.3: High level overview of MiLAN operation. Segment A repeats when the application changes its state based on data received from the sensors. Segment B repeats when sensors arrive in the network. Segment C repeats as data arrives from each sensor, and represents the normal operation of MiLAN conveying information from the sensors to the application. Note: arrows indicate dependencies among messages and computations. When no dependency exists, operations can be executed in parallel.

and heart rate may be determined based on measurements obtained from any of several sensors [67]. Each sensor has a certain reliability in characterizing each of the application's variables. For example, a blood pressure sensor directly measures blood pressure, so it provides a quality of 1.0 in determining this variable. In addition, the blood pressure sensor can indirectly measure other variables such as heart rate, so it provides some quality, although less than 1.0, in determining these variables. The quality of the heart rate measurement would be improved through high-level fusion of the blood pressure measurements with data from additional sensors such as a blood flow sensor.

In order to determine how to best serve the application, MiLAN must know (1) the variables of interest to the application, (2) the required QoS for each variable, and (3) the level of QoS that data from each sensor or set of sensors can provide for each variable. Note that all of these may change based on the application's current state. As shown in Figure 3.3, during initialization of the application, this information is conveyed from the application to MiLAN via "State-based Variable Requirements" and "Sensor QoS" graphs. Examples of these graphs are shown in Figures 3.4 and 3.5, respectively. Figure 3.4 shows the required QoS for each variable of interest based on the current state of the system and the variables of interest to the application, where these states are based on the application's analysis of previously received data. For a particular state (a combination of system state and variable state), the State-based Variable Requirements Graph defines the required QoS for each relevant variable. Because variables can be named in multiple variable states, MiLAN must extract the maximum QoS for each selected variable to satisfy the requirements for all variable states. The health monitor application has two state types, a system state that includes the patient's overall stress level, as well as variable states for each variable that can be monitored. The State-based Variable Requirements Graph specifies to MiLAN the application's minimum acceptable reliability for each variable (e.g., blood pressure, respiratory rate, etc.) based on the current state of the patient. For example, the figure shows that when a patient is in a medium stress state and the blood pressure is low, the blood oxygen level must be monitored with a quality level of 0.7 and the blood pressure must be monitored with a quality level of 0.8.

For a given application, the QoS for each variable can be satisfied using data from one or more sensors. The application specifies this information to MiLAN through the

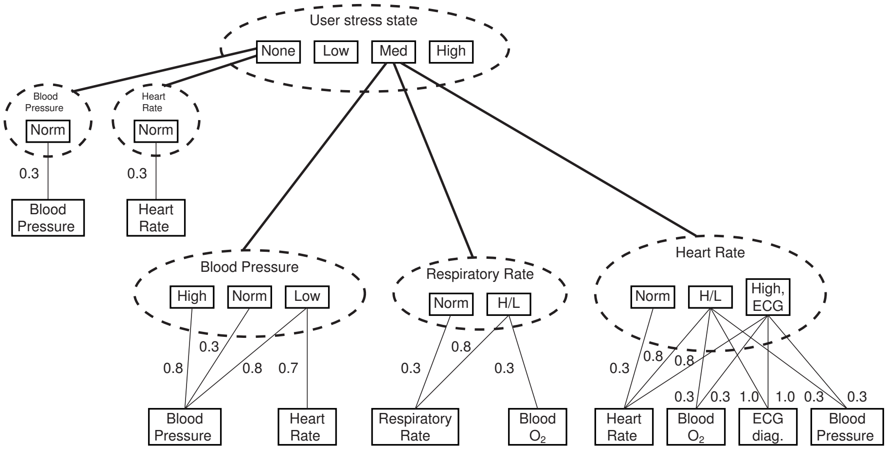


Figure 3.4: Example State-based Variable Requirements Graph for for the personal health monitor application. The graph specifies the variables and the required QoS when the application is in various states. This graph illustrates only a subset of the application's possible states.

Sensor QoS Graph. When multiple sensors are combined to provide a certain reliability level to the variable, we refer to this as a single "virtual sensor." Figure 3.5 shows the Sensor QoS Graph for the personal health monitor. This graph illustrates the important variables to monitor when determining a patient's condition and indicates the sensors that can provide at least some quality to the measurement of these variables. Each line between a sensor (or virtual sensor) and a variable is labeled with the reliability that the sensor (or virtual sensor) can provide to the measurement of that variable. For example, using data from a blood pressure sensor, the heart rate can be determined with a reliability level of 0.7, but combining this with data from a blood flow sensor increases the reliability to 1.0.

Given the information from these graphs as well as the current application state, MiLAN can determine which sets of sensors satisfy all of the application's reliability requirements for each variable. These sets of sensors define the *application feasible set* F_A, where each element in F_A is a set of sensors that provides reliability greater than or equal to the application-specified minimum acceptable reliability for each specified variable. For example, in the personal health monitor, for a patient in medium stress with a high heart rate, normal respiratory rate, and low blood pressure, the application

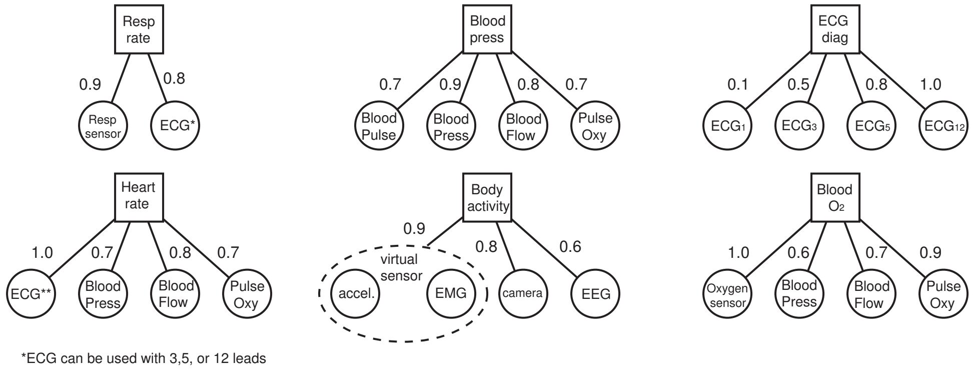


Figure 3.5: Example Sensor QoS Graph for the personal health monitor example. The graph specifies which sensors, or sets of sensors, can provide what level of QoS for each variable. This graph illustrates only a subset of the variables that should be considered by the application.

feasible sets in F_A that MiLAN should choose to meet the specified application QoS are shown in Table 3.1. MiLAN must choose which element of F_A should be provided to the application. This decision depends on network-level information.

3.1.2 Network Control

The properties of specific network types as well as the current condition of the network can constrain the set of feasible sets to a subset of those in F_A. As shown in Figure 3.3, it is the network plug-in's job to determine which sets of nodes (sensors) can be supported by the network, as well as other protocol-specific information, such as what role each node must play.

MiLAN can use a service discovery protocol provided by the system architecture (e.g., SDP in Bluetooth), or provide its own, to find new nodes and learn when nodes are no longer accessible (due to mobility or exhaustion their energy resources). The service discovery protocol must return important information about each node, such as the type of data that can be provided by that node, the modes that the node can operate in, the transmission power levels, and the current residual energy level. Using this information from each currently available node, the network plug-in must determine which sets of nodes can be supported by the network.

Set #	Sensors
1	Blood flow, Resp. rate
2	Blood flow, ECG (3 leads)
3	Pulse oxymeter, Blood pressure, ECG (1 lead), Resp. rate
4	Pulse oxymeter, Blood pressure, ECG (3 leads)
5	Oxygen measurement, Blood pressure, ECG (1 lead), Resp. rate
6	Oxygen measurement, Blood pressure, ECG (3 leads)

Table 3.1: Feasible sets F_A for the personal health monitor application for a patient in medium stress with high heart rate, normal respiratory rate, and low blood pressure.

If we assume that all nodes are on a single-hop, centralized network, bandwidth constraints place limitations on the total amount of data that can be transmitted to the application. For example, if all nodes are on a Bluetooth piconet or an 802.11 network operating in infrastructure mode, all nodes transmit data directly to the application (residing at the master in Bluetooth or the Access Point in 802.11). Therefore, the network constraint is the total rate of all data transmitted.

However, in more complex environments such as Bluetooth scatternets, 802.11 multi-hop networks, or hybrid networks, network topology plays an important role in determining network feasibility and power costs. For example, in Bluetooth it is necessary to choose a feasible scatternet topology, where nodes selected in the feasible set allow the network to be fully connected. In addition to ensuring the feasibility of a network configuration, we must also consider how the power costs of nodes are affected by their roles in the network (e.g., piconet masters or bridge nodes in Bluetooth scatternets [66], coordinators in Span [45]). The power cost of using a node is a combination of the power to run the device, the power to transmit its data, the power to forward the data of other nodes in the set, and the overhead of maintaining its role in the network. These costs can be influenced by MiLAN through techniques such as transmission power control, efficient traffic scheduling, and the setting of different sleep states. In multi-hop networks, routing data from nodes to the application also becomes an important factor. The plug-in should know all of the network's protocol-specific features that can be modified and choose how to set these features to make sets feasible and energy-efficient.

The subsets of nodes that can be supported by the network define a network feasible set F_N. As only sets in F_A provide the required application QoS, we can combine these two constraints to get an overall set of feasible sets:

$$F = F_A \cap F_N \tag{3.1}$$

For the personal health monitor, suppose that the sensors and processors communicate using an IEEE 802.11b network. As these networks can support overall throughput of nearly 11 Mbps, the network is able to support the transmission of all data from each of the sensor sets in F_A from Table 3.1 in real-time. However, if other applications are running simultaneously on the network and the personal health monitor application can only utilize 100 kbps of the throughput, the network would not be able to support the transmission of data from the ECG sensor with either 3, 5, or 12 leads. Thus, the set of network feasible sets F_N will only partially overlap with F_A. This overlap is the set of feasible sets F and consists of sets 1, 3, and 5 in Table 3.1. MiLAN must choose a set of sensors from one of the sets in F based on the tradeoffs discussed in the next section. If F is empty, MiLAN should raise an exception to the application, allowing it to decide the appropriate action.

3.1.3 Tradeoffs

Among the sensor sets in F, MiLAN chooses a set that represents the best tradeoff between performance and cost . How should “best” be defined? This depends on the application – the MiLAN framework supports any method of deciding how to choose an element of F. In most sensor network applications, we want to allow the application to last as long as possible using the limited energy of each of the sensors. Simple approaches to choosing sensor sets may yield the set from F that consumes the least power or that will run for the maximum lifetime before the first sensor dies. However, if we want to ensure that the application can run at the required QoS level as long as possible, we should instead optimize the total lifetime by intelligently choosing how long to use each feasible sensor set, as described in Chapter 4. In some cases, there are multiple ways to schedule sensors so that the same total network lifetime is achieved. In these cases, we may want to maximize the average quality of the sensor sets over time. For some applications, the goal may be to maximize some combination of lifetime and quality. MiLAN is flexible enough to incorporate any of these or other optimization criteria.

In Figure 3.3, we show this tradeoff computation occurring in the core MiLAN component. After the computation is complete and the first set of sensors is chosen, the MiLAN core informs the plug-in of the selection, and the plug-in configures the network accordingly, using information about the role each sensor should play.

3.2 Summary

In this chapter, we have presented a generalized middleware framework for sensor networks named MiLAN that allows the network to dynamically configure itself according to application QoS goals and current network conditions. A preliminary version of the MiLAN middleware has been implemented on a Linux platform with a Bluetooth plug-in using the BlueZ Bluetooth protocol stack. Using this implementation, we have demonstrated a simple application transmitting ECG data to a remote monitor with a display. We have also begun implementation of the MiLAN architecture on the TinyOS platform, and a full implementation is planned in the future. An enabling technology called X-Lisa [68], which facilitates information-sharing between different protocol layers, may ease further transition of MiLAN to TinyOS.

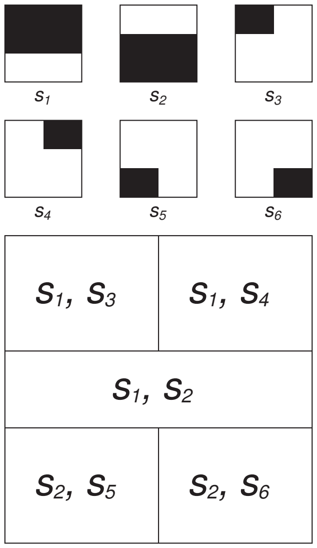
The middleware proposed in this chapter provides a framework for tuning the parameters of network protocols and policies in order to meet quality of service goals. With this framework in place, MiLAN can optimize sensor role assignment in order to maximize network lifetime for a given application requirement. In the following chapters, we show how this optimization can be accomplished and present several distributed protocols and policies for performing role assignment in real life sensor networks.

Chapter 4

Optimizing Sensor Role Management

In some situations, sensor networks may consist of sensors with overlapping coverage areas that provide redundant information, giving an application a quality level that is more than necessary. Rather than provide this unnecessary redundant data, it may be desirable to reduce power consumption and conserve energy in these sensors to lengthen the lifetime of the network or minimize the rate at which the sensors must be replenished with energy. This energy conservation can be accomplished through a number of methods. For example, sensors' reporting rate or data resolution can be adjusted, or the sensors can be turned off completely for an extended period of time. Balancing the application quality with this goal of energy-efficiency essentially provides a type of application quality of service (QoS). In this chapter, we show how the use of several strategies, including redundant sensor deactivation and energy-efficient routing, can be optimized to extend network lifetime while meeting a required level of application quality. In the optimizations presented in this chapter, the specification of time-varying roles (i.e., when individual sensors should sense data, how much of others' data each sensor should route) is integrated for maximum efficiency.

Consider an application that relies on data from a number of sensors in a network. The individual sensor data is gathered at a base station in order to provide a complete picture of the region of interest. One or more sensors may be used at any time to provide data to the application, but only certain subsets of available sensors may satisfy channel bandwidth and/or application quality of service constraints. For example, consider the network shown in Figure 4.1. Each sensor is capable of reliably monitoring some portion of the environment, illustrated in the figure. If we wish to detect the presence of



f_1	s_1, s_2
f_2	s_1, s_5, s_6
f_3	s_2, s_3, s_4

Figure 4.1: Example sensor coverage scenario.

phenomena anywhere in the environment with minimal redundancy, there are a number of sensor sets, also shown in Figure 4.1, that cover the entire environment and may be used for this purpose.

The problem that we wish to solve is to determine which sensor sets should be used and for how long so that the lifetime of the network is maximized while the necessary quality of service is always maintained at the application. Assume that the power consumption and initial energy among the sensors in this case is identical (1 μW and 1 μW-day, respectively). When deciding which sensor set to use, a naive approach may be to activate the set with the fewest required active sensors. This would result in the choice of sensor set f_1. After this set is used for one day, the energy supplies of sensors s_1 and s_2 would be exhausted and there would remain no sensor set from the table in Figure 4.1 that covers the entire region of interest. Thus, the surveillance application would obtain a lifetime of one day, during which it could achieve the desired level of QoS. However, if a more intelligent approach were used, sets f_2 and f_3 could be chosen and used subsequently, allowing the surveillance application to operate for two days at its desired level of QoS.

In this chapter, we generalize this sensor role management optimization problem for single hop sensor networks and show how it can be solved using linear programming.

The problem is later generalized to networks where multi-hop forwarding is required to deliver data to the data sink.

4.1 Single Hop Wireless Sensor Networks

We will refer to the complete set of sensors and their locations as $S = \{s_1 \dots s_{N_S}\}$ and $L = \{l_1 \dots l_{N_S}\}$, where N_S represents the number of sensors deployed in the network. In general, sensors in the network are capable of operating in multiple active modes and additionally in sleep mode, where power consumption is negligible. An example of a sensor that is capable of operating in multiple active modes is a video camera that can send data at variable resolution or with a variable frame rate. For simplicity, in this section we will assume that nodes only operate in a single active mode. All sensors can communicate directly with a single data sink s^*, which is not energy-constrained. In order to achieve the application's required QoS level, it may be possible to use a number of the sensors by themselves or in combination, constituting a feasible sensor set. A sensor set is determined to be feasible if

- the total bandwidth necessary to support the set is below the capacity of the network and the traffic can be scheduled, and
- the set provides the application with a quality that meets its minimum quality requirements.

We will refer to the set of feasible sensor sets as $F = \{f_i \dots f_{N_F}\}$. A single feasible sensor set is defined by the sensors that compose the set as well as their operational mode such that $f_i = \{s_{j_1}^{k_1} \dots s_{j_{N_{f_i}}}^{k_{N_{f_i}}}\}$, where superscripts indicate the operating mode of the sensors in the set. It should be noted that only sensor sets in which it is not possible to reduce power consumption in any one sensor while not increasing power consumption in any of the sensors or falling below QoS requirements should be considered in the optimization. In other words, the sensor sets should not provide unnecessarily redundant coverage.

We must also introduce R_k, P_k^{sense}, and $E_{tx,bit}$, which represent the bit rate of a sensor operating in mode k, the sensing power consumption of a sensor operating in mode k, and the communication energy consumption per bit, respectively. In this chapter, wherever the subscripts are dropped from R_k and P_k^{sense}, it should be assumed that we are considering homogeneous sensors capable only of operating in a single active mode. In general, $E_{tx,bit}$ is dependent on the distance of the link being transmitted over; however, in this section, we will assume a constant value with respect to link distance.

In addition to the bandwidth and QoS constraints that were considered when forming the set of feasible sensor sets F, we are constrained by the initial battery levels E_j of the sensors such that

$$\sum_{i,k:s_j^k \in f_i} (R_k E_{tx,bit} + P_k^{sense}) \times T_i \leq E_j \qquad \forall s_j \tag{4.1}$$

We wish to develop a schedule that determines the length of time that each sensor set should be used to provide data to the application. Let T_i represent the length of time that feasible sensor set f_i is being used in the schedule. The objective of the problem is to maximize

$$T_{total} = \sum_{i=1}^{N_F} T_i \tag{4.2}$$

Typically, the most difficult aspect of maximizing sensor network lifetime for large-scale sensor networks is finding the feasible sensor sets. Depending on the application, this procedure can be very simple or very difficult. A simple example can be drawn from a personal health monitor system. In this application, a number of medical variables (e.g., blood pressure, temperature, oxygen levels, etc.) need to be monitored by some combination of sensors attached to the body, some of which are capable of measuring multiple variables. An example of the sensors that may be used in the system, as well as the variables that must be monitored and the associations between the sensors and variables, is given in Table 4.1. In this case, the process of finding the feasible sensor sets is not difficult and may be carried out manually. Another class of applications that might require sensor role optimization are those requiring coverage of some region of interest. In such applications, each sensor is assumed to have some sensing range and hence a coverage region, within which the sensor can detect any events of interest with high probability. In other words, the active sensors at any point in time must form a cover set. It may be reasonable to find all possible cover sets when the size of the

○	Heart rate
◇	Blood pressure
◁	Muscle activity
○	Blood oxygen
⋆	Respiratory rate

(a) Variables

					Sensors
○	◇		○		Blood pressure
	◇		○		Blood pulse
○	◇				Blood flow
○				⋆	ECG
○	◇		○		Pulse oxymeter
		◁			EEG
		◁			EMG
		◁		⋆	Respiratory sensor

(b) Sensors

Table 4.1: Variables measured by different sensors in a personal health monitoring application.

network is on the order of tens of nodes; however, as the number of cover sets grows exponentially with the network size, this becomes impossible for large-scale networks.

4.1.1 Large-Scale Networks

Our work has shown how to optimize network lifetime when it is possible to enumerate all possible feasible sensor sets [69]. In the examples of the personal health monitoring system and the coverage application in small-scale networks, it is not computationally intensive to find all possible feasible sensor sets. However, the problem of finding all feasible sensor sets may become extremely large and computationally infeasible for networks consisting of a large number of sensors and a large amount of sensing redundancy. For example, in the coverage application, the number of cover sets grows exponentially with the number of sensors deployed, making the enumeration of all cover sets impossible for large-scale networks.

Rather than performing this enumeration, it would be ideal to find a subset of F whose optimal scheduling would yield a similar lifetime as the optimal scheduling of F. Following our work in [69], Berman showed how the calculation of the feasible sets can be accomplished simultaneously with the scheduling of the sets [70]. The work is

```
Initialize: $\delta = (1+\epsilon)((1+\epsilon)m)^{(-1/\epsilon)}$, $y = \frac{\delta}{b}$, $D = m\delta$, $j = 0$
While $D < 1$
    Find the column $A^q$ using the approximation algorithm.
    Compute $p$, the index of the row with the minimum $\frac{b(i)}{A^q(i)}$.
    $j = j + 1$, $x^j = \frac{b(p)}{A_q(p)}$, $A^j = A^q$
    $y = y(1 + \epsilon \frac{b(p)}{A^q(p)} / \frac{b}{A^q}) \qquad \forall i$
    $D = b^T y.$
Output $(A^j, \frac{x^j}{\log_{1+\epsilon} \frac{1+\epsilon}{\delta}})_{j=1}^{k}$
```

Figure 4.2: Garg-Könemann algorithm.

based on an algorithm proposed by Garg and Könemann [71].

For an arbitrary linear program (LP) packing problem

$$max\{c^T x | Ax \leq b, x \geq 0\}$$

if an approximation algorithm for finding the column j of the matrix A that minimizes the value of $length_y(j) = \frac{\sum_i A_{ij} y(j)}{c(j)}$ for any arbitrary vector y, then the Garg-Könemann algorithm yields a solution whose goal function value $c^T x$ is arbitrarily close (within some factor ϵ) to the optimal value while using only a subset of the columns of the complete matrix A. The Garg-Könemann algorithm proceeds as shown in Figure 4.1.1.

In order to use the Garg-Könemann algorithm to solve the sensor scheduling problem, the vector b is replaced by the initial energy vector E and the vector x is replaced by the time schedule vector T. The complete set of feasible sensor sets and individual nodes' power consumption during those sets can be represented in A by setting the elements A_{ij} as

$$A_{ij} = \begin{cases} P_k^{sense} & s_i^k \in f_j \\ 0 & else \end{cases} \tag{4.3}$$

Since we assume that all cover sets provide adequate QoS, all elements of $c(j)$ are set to unity. The Garg-Könemann algorithm applied to the sensor cover set scheduling

```
1   Initialize: $\delta = (1+\epsilon)((1+\epsilon)N_S)^{(-1/\epsilon)}$, $y = \frac{\delta}{E}$, $D = \delta N_S$, $j = 0$
2   While $D < 1$
3       $j = j + 1$
4       Find $f_j$, the approximate minimum length cover set.
5       Find $s_i^k$, the sensor in $f_j$ with minimum $\frac{E_i}{P_k^{sense}}$.
6       $T_j = \frac{E_i}{P_k^{sense} \log_{1+\epsilon} \frac{1+\epsilon}{\delta}}$.
7       $y_{i'} = y_{i'}(1 + \epsilon \frac{E_i}{P_k^{sense}} / \frac{E'_i}{A_{i'j}}) \qquad \forall s_{i'}^k \in f_j$
8       $D = E^T y$.
```

Figure 4.3: Garg-Könemann algorithm applied to the sensor scheduling problem.

problem proceeds as shown in Figure 4.1.1. Line 4 consists of iteratively choosing a cover set. Line 6 schedules that cover set to be used for a round length proportional to the initial energy of the most constrained sensor in the cover set.

Simple simulations illustrate the attainable lifetime as a function of the number of sensors deployed in a network. Figure 4.4 shows results from such simulations where the number of sensors is varied while the sensing range remains fixed at $150m$. We set the value of ϵ equal to 0.05 while finding the cover sets for these simulations.

Using the Garg-Könemann algorithm greatly reduces the number of sensor sets that need to be found to calculate a near optimal schedule. However, the calculation involved in this task is not insignificant. As ϵ is set smaller (i.e., the output of the algorithm becomes more optimal), the round lengths calculated in line 7 of Figure 4.1.1 become small very quickly. In other words, the number of sets that need to be calculated increases very quickly as ϵ decreases. Figure 4.5 shows the round length as a function of ϵ in a network of 200 nodes, each containing a potential lifetime of 1. As the figure shows, for the solution to be guaranteed to perform within 5% of its potential maximum lifetime, approximately 2500 cover sets must be found. Fortunately, such a large number of cover sets may not need to be found in practice. Once the cover sets have been chosen, their scheduling may be modified through a linear program optimization, and this optimized lifetime can often match or nearly match the true upper bound of the lifetime even when a very large value of ϵ is used.

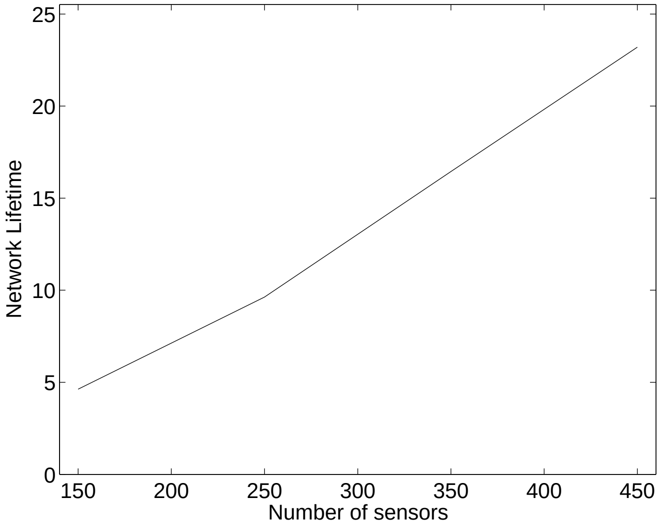


Figure 4.4: Network lifetime vs. network size. As the amount of energy distributed in the network increases and the level of coverage (i.e., average number of active sensors) remains constant, lifetime increases, as expected.

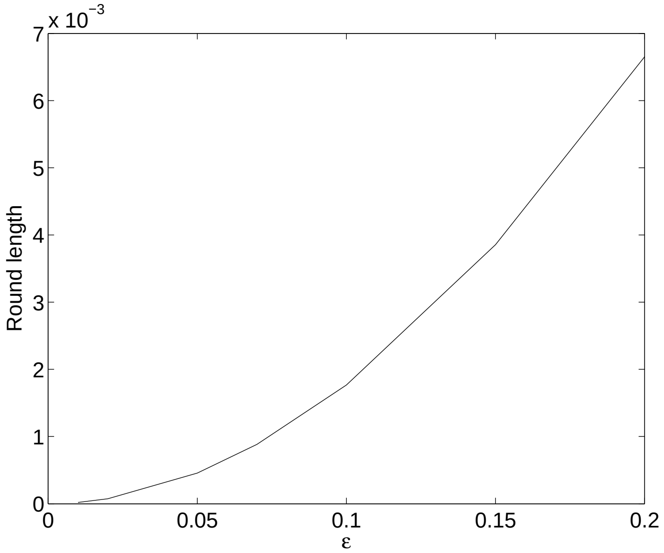


Figure 4.5: Round length vs. ϵ for the Garg-Könemann algorithm in a network of 200 sensor nodes.

4.2 Multi-hop Wireless Sensor Networks

The work in [70] has shown how the single-hop scheduling problem can be approximately solved within reasonable time constraints even for large-scale networks by considering only an important subset of the possible cover sets in the network when optimizing the sensor scheduling. In this section, we extend the sensor scheduling problem to account for multi-hop networks and present a solution based on the method used in [70].

In the previous section, we stated that in general, $E_{tx,bit}$ depends on the distance of the link being considered. In this section, we will use the notation $E_{tx,bit}(s_{i_1}, s_{i_2})$ to represent the energy necessary for s_{i_1} to transmit a bit to s_{i_2}. We must also introduce $E_{rx,bit}(s_{i_1}, s_{i_2})$, which represents the energy consumption necessary for s_{i_2} to receive a bit from s_{i_1}. Now that we are considering multi-hop networks, the free variables that need to be solved for include the routing information as well as the scheduling information that was determined by finding T. The routing information can be expressed through the variables $t_{i_1 i_2}$ and $p_{i_1 i_2}$, which represent the total traffic and the fraction of traffic, respectively, that sensor s_{i_1} forwards toward the base station using s_{i_2} as its next hop.

$$p_{i_1 i_2} = \frac{t_{i_1 i_2}}{\sum_{i' : s_{i'} \in S \cup s^*} t_{i_1 i'}} \tag{4.4}$$

Additional requirements regarding the conservation of data flow (i.e., that the sum of a node's incoming data and its generated data must equal its outgoing data) need to be introduced into this problem.

$$\sum_{i' : s_{i'} \in S \cup s^*} t_{ii'} = \sum_{i' : s_{i'} \in S} t_{i'i} + \sum_{j} \sum_{k : s_i^k \in f_j} R_k T_j \qquad \forall s_i \in S \tag{4.5}$$

Because we need to consider routing, the energy constraint originally defined in Equation 4.1 now becomes

$$\sum_{j} \sum_{k : s_i^k \in f_j} P_k^{sense} T_j + \sum_{s_i' \in S \cup s^*} t_{ii'} E_{tx,bit}(s_i, s_{i'}) + \sum_{s_i' \in S} t_{i'i} E_{rx,bit}(s_{i'}, s_i) \leq E_i \quad \forall s_i \in S \tag{4.6}$$

This constraint limits the total amount of time any node can route other nodes' data and the total amount of time any node can be an active sensor by that node's initial energy. Again, the goal of the multi-hop network optimization problem is to maximize T_{total}, the sum of the time for which individual feasible sets are used.

4.2.1 Consideration for Networks with Non-Static Data Sinks

If the data sink of the network does not remain constant with respect to node identity or location, some of the constraints of the previous section must be modified in order to perform the optimization. Consider a mobile data sink capable of residing at several locations represented by $L^* = \{l_1^* \ldots l_{N_L^*}^*\}$. In this case, the constraints that maintain the conservation of data flow must be given on a per-location basis. Thus, T_j must be modified as T_{jm} to represent the scheduled time of set f_j while the data sink resides in location l_m^*. Also, $p_{i_1 i_2}$ and $t_{i_1 i_2}$ must similarly be modified to $p_{i_1 i_2 m}$ and $t_{i_1 i_2 m}$, respectively. Incorporating these modifications, equations 4.5 and 4.6 become

$$\sum_{i':s_{i'} \in S \cup s^*} t_{ii'm} = \sum_{i':s_{i'} \in S \cup s*} t_{i'im} + \sum_j \sum_{k:s_i^k \in f_j} R_k T_{jm} \quad \forall s_i \in S, \;\; l_m^* \in L^* \tag{4.7}$$

and

$$\sum_m \left(\begin{array}{l} \sum_j \sum_{k:s_i^k \in f_j} P_k^{sense} T_{jm} + \\ \sum_{s_i' \in S \cup s^*} t_{ii'm} E_{tx,bit}(s_i, s_{i'}) + \\ \sum_{s_i' \in S} t_{i'im} E_{rx,bit}(s_{i'}, s_i) \end{array} \right) \leq E_i \quad \forall s_i \in S \tag{4.8}$$

while the goal is to maximize

$$T_{total} = \sum_j \sum_m T_{jm} \tag{4.9}$$

4.2.2 Joint Optimization for Multi-hop Sensor Networks

In order to optimize sensor network lifetime for a multi-hop sensor network, one may find a reasonable representation of F through the Garg-Könemann algorithm and proceed to a linear programming solution, using Equations 4.7 and 4.8 as constraints. This will be referred to as approach I for the remainder of this section. Alternatively, it is possible to directly apply the Garg-Könemann algorithm to the multi-hop sensor scheduling problem. This approach may be easier since the implementation is very simple and easily distributable. Recall that a single feasible sensor set in a single hop sensor network could be represented by a column of the matrix A, where each element of the column is proportional to the power consumption of a single sensor node. In the case of the single hop network, the values of the elements in a single column in A depend only on the active sensing mode for a node in the given sensor set. To model the multi-hop sensor scheduling problem as a packing LP problem, columns of the matrix A represent the power consumption of each node for a single configuration of the network (including active sensors comprising a cover set and their routes to the data sink) rather than simply the power consumption of the nodes constituting the cover set. For this modeling, $t_{i_1 i_2 m}$ must be modified to $t_{i_1 i_2 j}$, which represents the total amount of traffic forwarded by sensor s_{i_1} toward the data sink using s_{i_2} as its next hop during the operation of configuration f_j. It is assumed that the data sink location is fixed during the operation of a given configuration and is represented as l_j^*. Similarly, $p_{i_1 i_2 m}$ must also be modified to $p_{i_1 i_2 j}$, which represents the fraction of traffic forwarded by sensor s_{i_1} toward the data sink using s_{i_2} as its next hop during the operation of configuration f_j. Elements of A are set as the sum of the sensing and routing components of power consumption such that

$$A_{ij} = A_{ij}^s + A_{ij}^r \tag{4.10}$$

where

$$A_{ij}^s = \begin{cases} P_k^{sense} & s_i^k \in f_j \\ 0 & \text{else} \end{cases} \tag{4.11}$$

and

$$A_{ij}^r = \sum_{s_{i'} \in S \cup s^*} b_{ii'j} E_{tx,bit}(s_i, s_{i'}) + \sum_{s_{i'} \in S} b_{i'ij} E_{rx,bit}(s_{i'}, s_i) \tag{4.12}$$

where b represents the traffic rates induced on the network during a given configuration and can be derived from the currently used feasible sensor set and p. We can apply the Garg-Könemann algorithm to the problem

$$\max\{c^T T | AT \leq E, T \geq 0\}$$

where c^T again consists of a vector whose elements are set to some arbitrary positive constant.

The remaining implementation task is to find an approximation algorithm for finding the minimum length column of yA. For this, we turn to a greedy algorithm similar to the one used in the single hop optimizations. For a given cover set (i.e., sensing portion of a configuration), the length of yA^j can be minimized by using shortest cost paths with routing costs of y_i applied to each sensor s_i. Using these cost assignments, a link cost can be set as

$$C_{i_1 i_2} = E_{tx,bit}(s_{i_1}, s_{i_2}) y_{i_1} + E_{rx,bit}(s_{i_1}, s_{i_2}) y_{i_2} \tag{4.13}$$

and shortest cost routes can be easily found using a slightly modified version of Dijkstra's algorithm. To approximate the column A^j of matrix A that minimizes yA^j, we can use the same greedy algorithm as in the previous section to find the cover sets and subsequently find the shortest cost routes. This will be referred to as approach II in this section. Alternatively, since the selection of one node for inclusion in the cover set affects not only itself, but also its would-be routers, it may make more sense to pre-calculate the routes and add sensors to the cover set based on their cumulative route costs rather than simply their own costs y_i . This will be referred to as approach III in this section.

4.2.3 Simulations

In order to compare the performance of the approaches that were considered, we ran simulations of a coverage application in a disc network with a $500m$ radius. We also compared against approaches where scheduling and routing decisions were made blindly. In these blind approaches, routes were calculated according to a shortest cost algorithm with link costs equal to

$$C_{i_1 i_2} = \frac{E_{tx,bit}(s_{i_1}, s_{i_2}) R}{E_{res}(s_{i_1})} + \frac{E_{rx,bit}(s_{i_1}, s_{i_2}) R}{E_{res}(s_{i_2})} \tag{4.14}$$

where $E_{res}(s_i)$ represents the residual energy of sensor s_i. The active sensors were selected at the beginning of rounds (with pre-determined length) according to a weighted greedy algorithm with each node's cost assigned as $\frac{P_{sense}}{E_{res}(s_i)}$ in the approach using post-computed routes (called approach IV) and as the sum of its cumulative route cost and

$\frac{P_{sense}}{E_{res}(s_i)}$ in the approach using pre-computed routes (called approach V). We used the energy model of [36], in which the energy to transmit and receive a bit were

$$E_{tx,bit}(s_{i_1}, s_{i_2}) = E_{elec} + \epsilon_{fs} \| l_{i_1} - l_{i_2} \|^2 \tag{4.15}$$

and

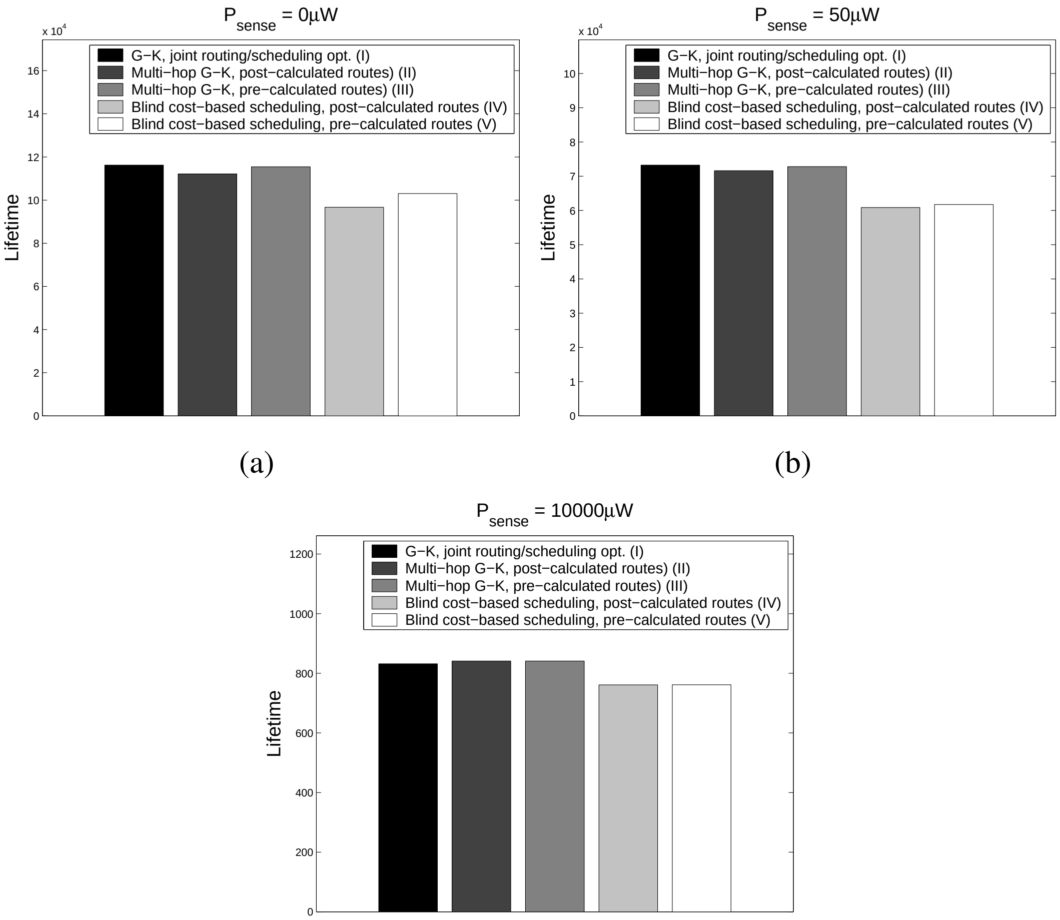
$$E_{rx,bit}(s_{i_1}, s_{i_2}) = E_{elec} \tag{4.16}$$

respectively. E_{elec} (the electronics energy) was set to $50\ nJ$ and ϵ_{fs} (the constant that characterizes the power amplifier) was set to $10\ pJ/m^2$. In our simulations, 200 sensors were randomly deployed, each containing equal initial energy. The sensing and transmission range were set to $150m$ and $250m$, respectively. The sensing power was varied to see the effect of the importance of routing on the performance of the different routing and scheduling approaches. In order to avoid hot spots in the network, the data sink was capable of moving between several locations in the network.

Figure 4.6 shows the results of our simulations. The plot shows that the approach using the Garg-Könemann algorithm to find the sets followed by joint scheduling/routing optimization, as well as the integrated multi-hop Garg-Könemann algorithms, with both pre-computed and post-computed routing, perform nearly as well as each other for all values of P_{sense}. Thus, it can be seen that the integrated multi-hop implementation of the Garg-Könemann algorithm can attain nearly optimal lifetime and outperforms the blind approaches.

4.3 Summary

In this chapter, we have reviewed the means by which the roles of sensors can be optimized so that network lifetime can reach its potential maximum for given energy constraints and application requirements. However, using these optimized approaches may not be practical, as the required computation may exceed the capabilities of realistic sensor nodes, or even certain data sink nodes, especially in large-scale networks. In subsequent chapters of this dissertation, we will present some protocols that can be used to accomplish the goals of these optimizations through distributed means.



(a) (b)

(c)

Figure 4.6: Comparison of approaches for schedule and routing optimization for $P_{sense} = 0$ (a), $P_{sense} = 50\mu W$ (b), and $P_{sense} = 10mW$ (c).

Chapter 5

Sensor Resolution Management

In Chapter 3, we presented a framework for adapting the network to current application QoS requirements. In this chapter, we consider the QoS requirements for a sensor network application whose goal is to reconstruct of a band-limited signal sensed by a subset of sensors in the network. We propose a method to decide which nodes in a wireless sensor network should be used to actively sense the environment and which should remain off in order to conserve energy. We consider the situation where a large number of sensor nodes are randomly deployed within the region to be monitored such that in all subregions of the area, it is likely that the density of the sensor nodes is more than necessary to meet the signal-to-noise ratio (SNR) requirements of the application. In such situations, sensing accuracy can be traded for energy-efficiency, meaning that a smaller subset of sensor nodes can be used to observe the area and the set of active sensors may be rotated throughout the lifetime of the network.

5.1 Blue-Noise Spatial Sampling

In this chapter, we describe a protocol for sensor resolution management that is based on the blue noise masking algorithms that are used in many image processing applications such as image half-toning. The traditional purpose for using blue noise masking is that half-toned images created from a blue noise mask appear very accurate to the human eye when compared with images created from other simpler masks. While this property is not particularly beneficial to our application of recreating a data image of some physical phenomenon to be sensed, blue noise masks have other desirable attributes. Particularly, sampling points in a blue noise sampling pattern are very well spaced for a random pattern and rarely yield large areas absent of sampling points. Thus, when stochastically sampling a bandlimited signal, the maximum spatial distance between sampling points is nearly minimized using a blue noise pattern for a given number of sampling points. In other words, the number of points necessary to stochastically sample a bandlimited signal, meeting the Nyquist rate in all subregions, is nearly minimized using a blue noise spatial sampling pattern. Since sampling points are analogous to active sensors in our application, selecting sensors through a blue noise sampling algorithm can lead to significant energy savings as the number of sensors required to be active is minimized.

A blue noise pattern is a statistical model for describing stochastic patterns with very little frequency content below a blue noise principal frequency f_{BN} [72]. A binary blue noise mask/pattern $p_{BN}(x) \longrightarrow P_{BN}(f)$ is a special case of a blue noise pattern that consists of similarly sized impulses distributed in a homogeneous manner and maintains a stochastic nature (i.e., uniform distribution of the impulses is prohibited). By distributing impulses in such a way, the resulting spectral content of the pattern is composed almost entirely of high frequency content. A binary blue noise pattern may be described by the following equations.

$$p_{BN}(x) = \sum_{x_i} \delta(x - x_i) \tag{5.1}$$

$$\int_{0^+}^{f_{BN}} P_{BN}(f)df \ll \int_{f_{BN}}^{\infty} P_{BN}(f)df \tag{5.2}$$

Several algorithms have been proposed to generate binary blue noise patterns [73–76]. The method that is of interest to us is that which was proposed in [73] and improved in [74]. In their work, the authors propose a dart throwing method, mimicking a stochastic Poisson disc method. In this method, randomly chosen new points (impulses) are added to the point set if and only if no other points are located within a specified radius centered at the location of the new points. A low pass spatial filter is then used to determine which points contribute the most low frequency content (the points with the highest value of the filtered sampling pattern). These points are subject to relocation within the regions with the least low frequency content. Our proposed sensor selection method uses a modification of this algorithm to create a binary blue noise pattern.

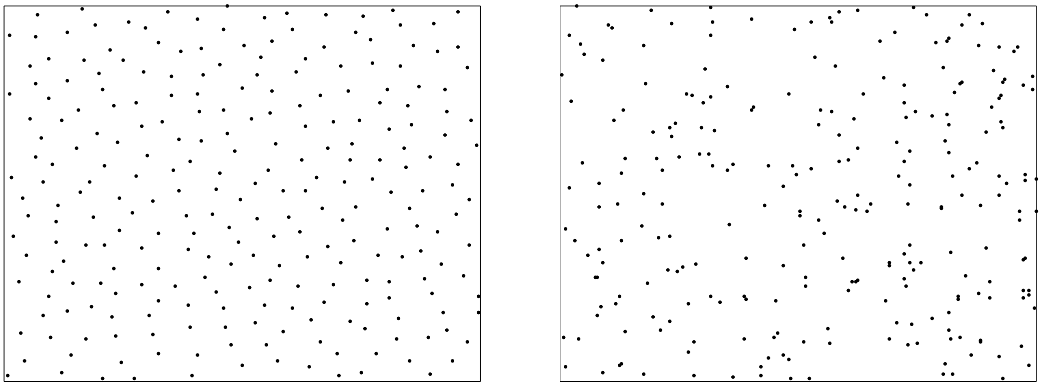

(a) (b)

Figure 5.1: Blue noise (a) and white noise (b) sampling patterns.

While the optimal sampling pattern for a bandlimited signal consists of samples at regular intervals (i.e., a grid pattern for images), other sampling patterns can be used with little performance loss. As long as the maximum spacing between sensors is small enough to meet the Nyquist sampling rate criteria, several algorithms can be used to perform ideal reconstruction, converging to a reconstructed signal with no error from the original [77–79]. In fact, when employing a sampling pattern based on a binary blue noise mask, a source signal can be reconstructed nearly as optimally as from a grid pattern. In other words, the number of sampling points necessary to perfectly reconstruct a bandlimited signal is only slightly higher when sampling with a blue noise pattern than when performing regular sampling at the Nyquist rate. Furthermore, the necessary number of sampling points is much less for blue noise sampling than random (white noise) sampling. Even if the local sampling rate falls below the Nyquist rate in some subregions, a blue noise sampling pattern will achieve higher accuracy than a white noise sampling pattern with the same number of sampling points. Figure 5.1 illustrates a typical blue noise sampling pattern and a typical random (white noise) sampling pattern. As the figure shows, the sampling points in the blue noise pattern are more evenly spread out than those in the white noise pattern.

5.2 Application to Sensor Resolution Management

Since the objective of our application is to reconstruct a spatially dependent signal such as a temperature field, we can consider the cumulative sensor data to be a stochastically sampled signal, sampled at the active sensor locations, and apply blue noise masking to our selection algorithm. Specifically, since the random deployment of the sensors limits us such that we must stochastically (rather than regularly) sample the phenomenon, we would like to sample with a blue noise pattern so that the accuracy is as high as possible for a given number of sensors. Thus, we designed the sensor selection algorithm using the intuition of the algorithm for creating blue noise patterns presented in [74].

We assume an image grid overlayed on the sensor network region with resolution high enough that each sensor's location can be precisely mapped into a single grid point and each point on the grid is associated with no more than one sensor. If a sensor is associated with a grid point, the grid point is assigned a value of 1; otherwise, it is assigned a value of 0. The resulting binary pattern $p_{initial}[i, j]$ should have white noise spectral characteristics because of our assumption about the random deployment of sensors.

In our proposed method, a low pass filter relaxation algorithm is applied to the sampling pattern $p_{current}[i, j]$, which is initially set to $p_{initial}[i, j]$. The characteristics of the low pass spatial filter $h_{BN}[i, j]$, which is a symmetric filter based on a one-dimensional impulse response $h_{BN}(d)$, should depend on the characteristics of the phenomenon that is being observed. More specifically, the coefficients of the low pass filter are determined such that the frequency content of the observed variable falls within the filter's pass band. Meanwhile, the selection of the order of the filter is essentially a tradeoff between the desired performance of the algorithm and computational cost. The algorithm filters the sampling pattern to create $F[i, j]$, such that $F = p_{current} * h_{BN}$. Next, the algorithm finds the maximum value of $F[i, j]$ at the pixels where any currently active sensors reside, deactivates the corresponding sensor, and updates $p_{current}$ by inverting the value of the pixel that the sensor is mapped to. Essentially, this results in the removal of a sensor that contributes significant low spatial frequency content. These steps are carried out iteratively until the maximum filter output drops below a predetermined threshold or the number of remaining active sensor nodes drops below a certain value. The resulting sampling pattern $p_{BN}[i, j]$ is shown to have blue noise spectral characteristics [74].

5.3 Incorporation of an Energy Cost

A challenging design goal for wireless sensor networks is to allow the networks to operate unattended for extended periods of time. In addition to reducing overall power consumption through methods such as the one described above, it is also important to avoid using nodes with little residual energy. Consider a network that chooses active sensors based on a method similar to the one described above, where chosen sensors operate until their energy supply becomes completely depleted. Toward the end of the network lifetime, the available sensors from which to choose will be much more sparse and so many more sensors may need to be chosen to achieve the same SNR as in the early stages of the network. In addition to reducing the number of active sensors, we would also like to extend the time before any of the sensors in the network die in order to avoid this situation.

To achieve this goal, we add a modification to the proposed sensor selection method in which we incorporate energy costs. The energy costs are assigned according to a monotonically decreasing function of the residual energy of the sensor nodes. In this modified selection method, the low pass filter output is combined with an energy cost, calculated from the residual energy of the sensor nodes at the pixels to which they are mapped, according to

$$Cost(s_i) = F[x(s_i), y(s_i)]^{\alpha} \times Cost_{energy}(s_i)^{(1-\alpha)} \tag{5.3}$$

where α ($0 \leq \alpha \leq 1$) represents a tuning parameter that allows the designer to balance a tradeoff between ideality of the sensor pattern and balanced energy distribution. In essence, we are combing a redundancy cost with an energy cost. A typical energy cost assignment might be simply

$$Cost_{energy}(s_i) = \frac{1}{E(s_i)} \tag{5.4}$$

where $E(s_i)$ represents the residual energy of sensor s_i. The sensor with the highest overall cost is deselected and again, these steps are repeated iteratively until the active subset consists of the desired number of sensors or no longer guarantees a sufficient

SNR. Following the selection algorithm, the selected subset of nodes is used to observe the region for a certain time interval. After this interval, the selection algorithm is repeated again with updated energy information of the sensor nodes in the network.

5.4 Distribution of the Algorithm

As wireless sensor networks are expected to scale on the order of thousands of nodes, it is desirable to distribute self-organization algorithms. Conveniently, the nature of our proposed sensor selection algorithm makes it easily distributable. The deactivation of a sensor is affected only by sensors with distances of less than $\frac{N \times T_r}{2}$ from the sensor, where N represents the order of the filter used to create the blue noise sampling pattern and T_r represents the resolution of the image grid that the sensors are mapped to. In fact, there is not necessarily a need for any concept of a discrete grid, as long as each sensor s_i is able to translate distance to its neighbors s_n into the appropriate value of the impulse response $h_{BN}(dist(s_i, s_n))$.

As long as sensors are synchronized and begin the sensor selection algorithm simultaneously, they may set a backoff timer according to

$$B(s_i) = W - K \times F[x(s_i), y(s_i)] \tag{5.5}$$

$$F[i, j] = s_{initial}[i, j] * h_{BN}[i, j] \tag{5.6}$$

where W represents the maximum backoff window value, and K is chosen according to the maximum expected node density and the specific filter used to create the blue noise sampling pattern. Using this approach, sensors whose locations correspond to high values of F will have low backoff timers, as they should have highest priority for deactivation. When a sensor s_i's timer expires, it broadcasts a beacon to its neighbors, informing them of its intended deactivation. When the neighboring nodes s_n receive these deactivation beacons from s_i, they adjust their calculated filter output by subtracting $h_{BN}(dist(s_n, s_i))$ and increase their backoff time by $K * h_{BN}(dist(s_n, s_i))$. It is convenient that sensors can only increase their backoff time as a result of receiving another node's beacon, meaning that the nodes must only synchronize once during the algorithm. The algorithm should terminate once the density of the nodes approaches a threshold based on the desired SNR. In terms of the distributed algorithm, this means that once a node resets its backoff value past a certain threshold B_{max}, it should withhold its beacon, stop listening for beacons, and assume that it will remain active.

An illustration of how the algorithm works, using sensors deployed in a single dimension for clarity, is shown in Figure 5.2. Figure 5.2a shows the initial values of F and the corresponding timer values at each of the sensor nodes. Since sensor s_2 has the highest value of F, it is the first to send a beacon. In response to this, the other sensors update the values of F and their backoff timers, as shown in Figure 5.2b. s_2's beacon causes sensors s_1 and s_3 to set their timers beyond B_{max}, and the algorithm terminates at these nodes. At this point, s_5 has the highest value of F. After its backoff timer expires, s_5 sends a beacon and again, each of the other active sensors updates F and their backoff timers accordingly (Figure 5.2c). By now, all of the remaining sensors have increased their backoff timers beyond B_{max} and remain active, sensing the environment.

The energy cost modification can easily be incorporated into this distributed version of the protocol.

5.5 Lifetime Optimization

Here, we propose another energy-efficient sensor selection approach. In this approach, active sensor subsets are created in a similar manner as in the first approach (that does not incorporate an energy cost). However, rather than deterministically deselecting nodes at locations with the highest filter output, nodes are deselected with a weighted probability proportional to this value. Having introduced this factor of randomness, many active subsets, each providing the desired blue noise sampling characteristics, can be calculated. It is possible to schedule the use of each of these subsets so that the total lifetime of the monitoring application is maximized, as described in Chapter 4. Unfortunately, such an optimized schedule is not as robust or as easily distributed as the previously described algorithm and should only be used in static topology networks where one sensor has the computational power necessary to solve such a problem.

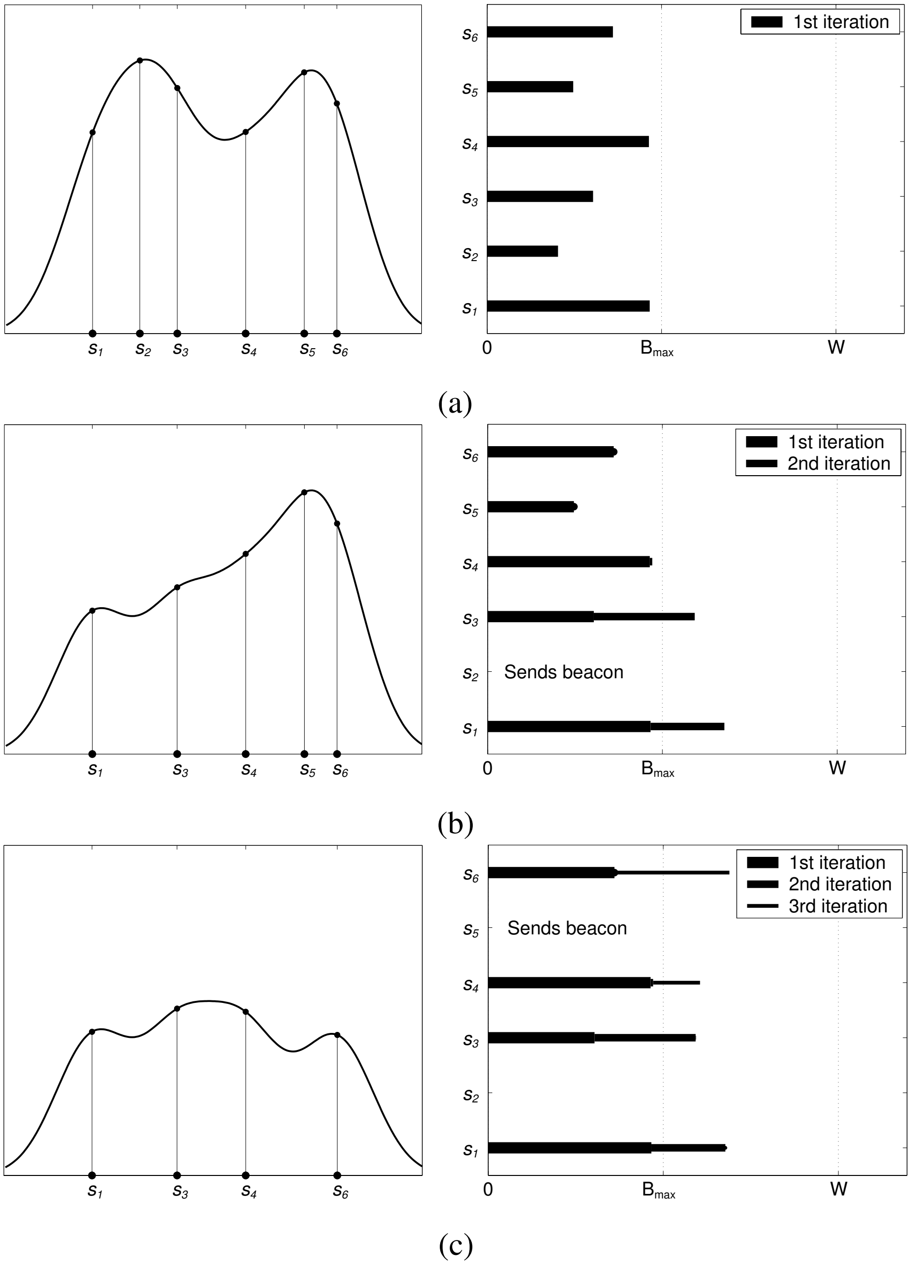


Figure 5.2: Distribution of the sensor selection algorithm. Figures (a)-(c) show the value of F and the current backoff timer values for the sensors at three time instances.

5.6 Simulations

We simulated a network of sensors randomly deployed within a $128m \times 128m$ region, of which a given number were activated to monitor a bandlimited phenomenon whose average signal power was normalized to unity. We compared our proposed blue noise sensor selection approach with the random selection of sensors (essentially, a white noise sampling approach) and a grid-based approach. In the grid-based approach, a hexagonal grid was overlaid onto the region where the sensors were deployed and the closest sensor to each grid point was selected for activation. Following the selection of nodes, the original data image (e.g., temperature field) was reconstructed from the sensor samples using the Voronoi reconstruction algorithm [77] and the mean square error was calculated.

In our first simulations, there were 250 sensors deployed to sense a signal that was bandlimited to $\frac{1}{20}m^{-1}$. We measured the mean square error of the reconstructed signal using each approach as we increased the number of sensors activated. Figure 5.3 shows that, as expected, for all methods the mean square error decreases as the number of activated sensors increases. However, the blue noise sampling method performs slightly better than the grid-based sampling approach and much better than the random (white noise) sampling approach. At first, it might seem surprising that the grid-based approach does not perform the best. However, this method is only expected to perform very well when the number of selected active sensors is much less than the number of deployed sensors. Otherwise, the sampling pattern created from the grid-based method may not resemble a grid at all, due to the random initial placement of the sensors. The blue noise selection algorithm also holds the advantage of being able to easily distribute the algorithm.

In Figure 5.3, the error goes to zero at approximately 150 sensors using the blue noise and grid-based approaches and approximately 200 sensors using the random sampling approach, meaning that at these points, the Nyquist sampling rate criteria is being met in all subregions. Even when the sensor deployment is not sufficient to meet this criteria, a subsampling approach can be beneficial if the deselected sensors are chosen correctly. In Figure 5.4, we plot the results from a scenario similar to the one above, but with a phenomenon that is bandlimited to $\frac{1}{15}m^{-1}$. In this case, even when all 250 sensors are activated, the mean square error of the reconstructed signal does not go to

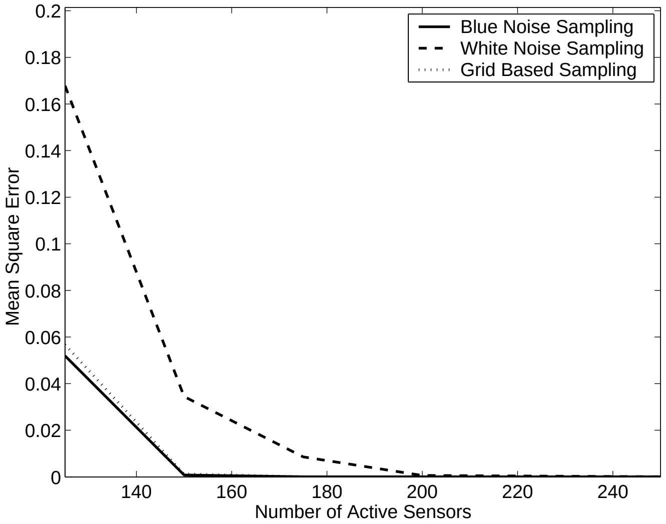


Figure 5.3: Mean square error of reconstructed signal using the blue noise, grid-based, and random selection methods for a phenomenon bandlimited to $\frac{1}{20}m^{-1}$.

zero. However, with an intelligent selection scheme such as the blue noise selection algorithm, a limited number of sensors may be deactivated without much further loss in signal quality. Figure 5.4 shows that with the blue noise selection algorithm, the deselection of 20 sensors has almost no effect on signal quality, and even the deselection of 40 sensors does not greatly affect signal quality.

Next, we ran some simple simulations to show the benefit of including an energy cost. In these simulations, 400 nodes were deployed, of which 150 were selected for activation. We used the inverse of a node's residual energy as its energy cost, so that

$$Cost(s_i) = f(x(s_i), y(s_i))^{\alpha} \times (\frac{1}{E(s_i)})^{(1-\alpha)} \tag{5.7}$$

where $E(s_i)$ represents the residual energy of sensor s_i, which was initially distributed uniformly between $2J$ and $10J$ in these simulations. We varied α to observe the tradeoff between ideality of the sampling pattern and appropriate distribution of the energy load. For a large value of α, the algorithm is essentially unchanged from the original version, while a small value of α means that good load distribution has become the more critical goal. This tradeoff is illustrated in Figure 5.5. For large values of α, a very low mean square error is achieved, but many nodes with little residual energy are selected. The

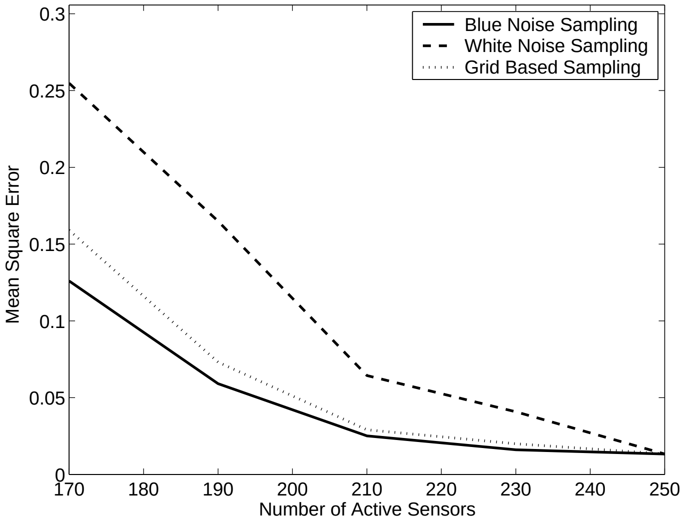


Figure 5.4: Mean square error of reconstructed signal using the blue noise, grid-based, and random selection methods for a phenomenon bandlimited to $\frac{1}{15}m^{-1}$.

average residual energy of the selected nodes is the mean of the energy distribution — $6J$. As we decrease the value of α, the average residual energy of the selected nodes increases, as desired, while there is initially a very small rise in mean square error. As the value of α decreases further, it can be seen that mean square error begins to increase more rapidly. A typical application may want to operate somewhere near the knee of the curve, simultaneously attaining the goals of accuracy of the reconstructed data image and good load balancing.

Finally, we ran simulations to observe the lifetime achievable through the optimization of the sensor schedule, as described previously. We compared the optimal schedule's lifetime with that of a blind selection method, in which we select the sensor subset based on the unaltered blue noise selection algorithm, and iteratively reselect the subset once a sensor in the subset dies. In these simulations, we deployed 100 sensors in a $128m \times 128m$ field and chose active subsets of 50 nodes to observe a phenomenon that was bandlimited to $\frac{1}{35}m^{-1}$. For the optimized schedule, we chose 100 sensor subsets of 50 nodes each according to the blue noise sampling pattern algorithm (with the necessary modifications to add a factor of randomness to the selection algorithm). Nodes were each given a lifetime of 1 (units are arbitrary).

In Figure 5.6, we show the average mean square error of the reconstructed signal

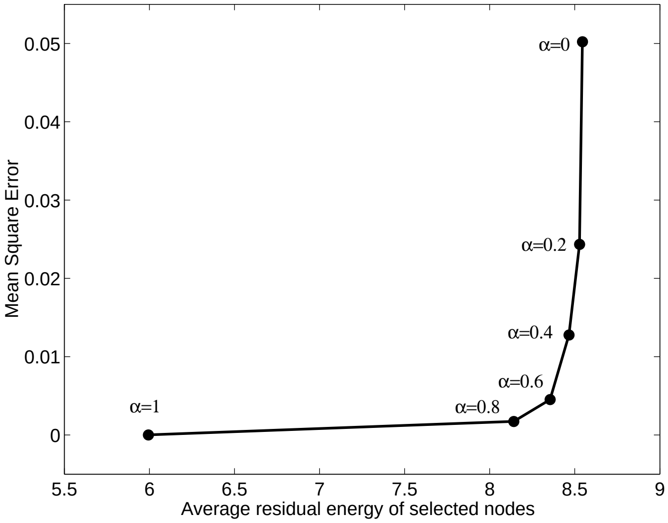


Figure 5.5: Selection optimality versus load balancing tradeoff imposed by the assignment of an energy cost.

as a function of time using the optimized sensor schedule and using the blind selection algorithm, averaged over 20 trials. Since the order in which the subsets are used in the optimized schedule is arbitrary, we show the average and standard deviation of these subsets instead of a time plot. The blind selection scheme initially chooses a sensor subset that reconstructs the data image with a low mean square error, as Figure 5.7(a) shows for a single trial. However, once the energy of the nodes in the first subset is used up (at time 1), there are only 50 sensors remaining from which to choose the next set of 50. The resulting sensor subset's sampling pattern, shown in Figure 5.7(b), does not have the desired blue noise properties. This is reflected in the high mean square error in the late stages of network operation. On the other hand, the optimization procedure uses many high quality sensor subsets, a sampling of which is shown in Figure 5.8. The optimal scheduling of these sets allows the network to perform well over a longer period of time, for an average length of 1.43.

To observe the performance of the optimization program when the energy among the deployed nodes is nonuniform, we simulated a network in which the nodes' initial energy was randomly distributed so that sensor lifetime ranged from 3 to 10 time units, taking only integer values in order to simplify the simulations. Again, we found 100 sensor subsets from which we calculated the optimal schedule and compared with the

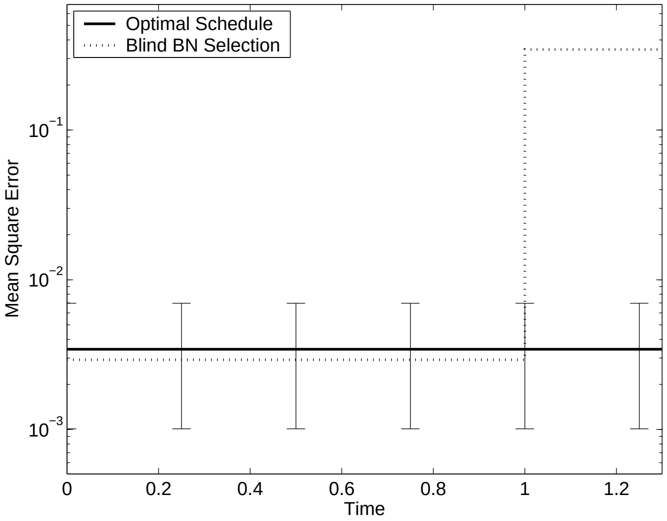
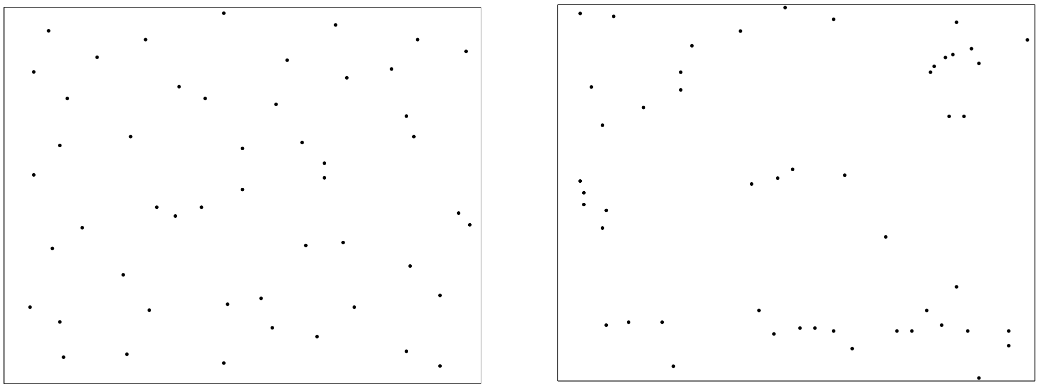


Figure 5.6: Mean square error of optimized sensor schedule (mean and standard deviation) compared with mean square error using the blind selection approach.



(a)

(b)

Figure 5.7: Initial (a) and subsequent (b) sets chosen through blind blue-noise sensor selection method.

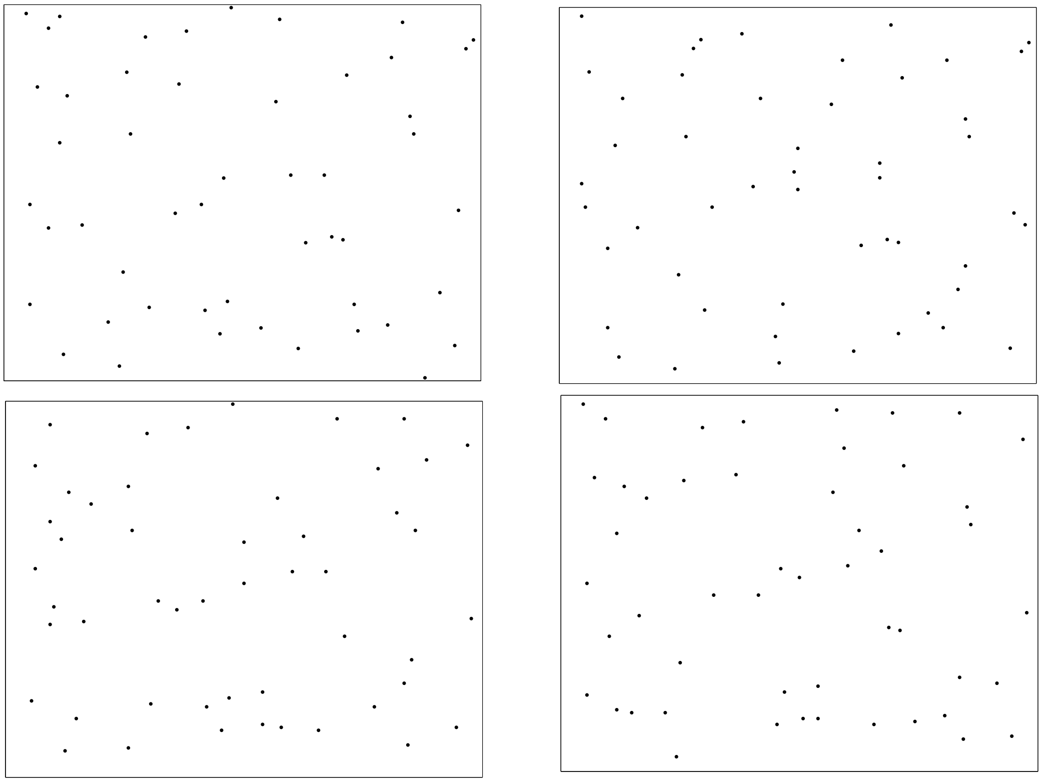


Figure 5.8: Sampling of sensor subsets used in the optimal sensor scheduling method.

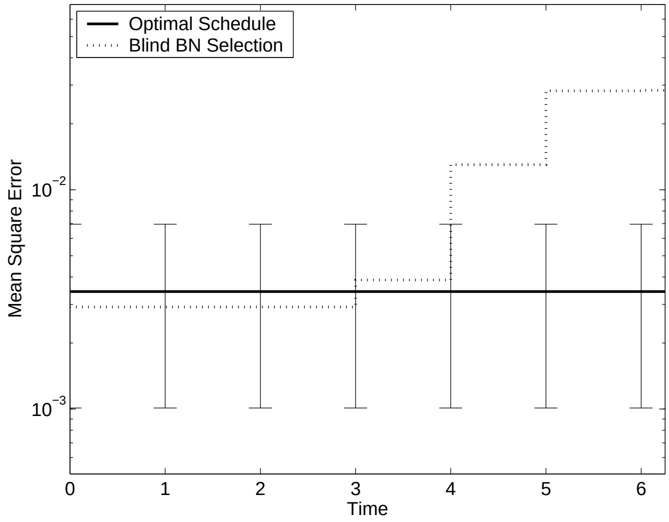


Figure 5.9: Mean square error of optimized sensor schedule (mean and standard deviation) compared with mean square error using the blind selection approach, for randomly distributed initial energy.

blind approach. The results are shown in Figure 5.9. The optimized schedule allows the network to operate with a low mean square error for an average time length of 6.3, while the average mean square error of the blind approach reaches an unacceptable level by this time. Of course, the amount by which lifetime can be extended is affected by factors such as how many subsets we run the optimization for as well as how much randomness we add to the selection algorithm, which allows sensor pattern ideality to be traded for sensor subset diversity.

5.7 Summary

In this chapter, we have presented a protocol for node selection in a wireless sensor network that uses the concept of blue-noise sampling to ensure that sensor spacing is approximately uniform throughout the network. This allows the phenomenon being sensed to be reconstructed accurately by the end user. We have shown that this protocol can be distributed, enabling its use in large-scale sensor networks. We have also shown how sensors' residual energy can be incorporated into sensor selection so that good sensor sets can be selected for a longer period if the activity of sensors is rotated throughout network lifetime. In this section, we have considered the sensor selection service for our target application. In the following chapters, we will shift our focus to the routing layer and present protocols that consider application goals when making routing and sensor selection decisions.

Chapter 6

Application-Aware Role Assignment

In the previous chapter, we considered application QoS requirements for an application whose goal is to reconstruct a phenomenon sensed by a subset of sensors in the network. Other sensor network applications may require coverage of the network by a subset of sensors that are assigned a nominal sensing range. In this chapter, we consider role assignment for such networks and focus on optimizing decisions at the routing layer as well as sensor selection.

Traditionally, the objective of routing protocols for ad hoc wireless networks has been to reduce power consumption by finding shortest path routes or routes that minimize overall energy consumption. Recently proposed routing methods have used awareness of energy resources to avoid routing through nodes with little energy so that network lifetime is extended as much as possible. In this chapter, we present several cost functions — called "application-aware" costs — for routing in wireless sensor networks that use awareness of energy resources as well as sensing capabilities of the nodes (i.e., coverage areas) so that sparsely deployed regions are avoided and the sensing application's lifetime is extended. In our work to date and throughout most of this chapter, the use of the application-aware costs is limited to routing decisions. However, this is not a fundamental limitation of the application cost's usefulness. Other work has shown the applicability of these cost metrics in sensor selection and clustering decisions [80].

6.1 Application-Aware Routing Costs

Our work is motivated by the intuition that for collaborative sensor networks, application goals should play a role in many of the network decisions, such as which sensors to activate and how to route the data. Specifically, sensors that are more important to the sensing application as data generators (e.g., those that are located in sparsely deployed areas) and those whose residual energy is least should not be chosen as routers over those that are less important to the application (e.g., those with more redundant neighbors) and those with more residual energy. Because these more important nodes are expected to consume more energy on average to sense data, they should be avoided as potential routers for the data generated by other active sensors. While traditional energy-aware routing costs will allow these important nodes to be routed around in the later stages of the network since their energy resources will be depleted faster than the less important nodes that are not used as often to sense data, care should be taken to route around these nodes in the early stages as well. Guided by this intuition, we propose the use of application-aware routing costs that consider the importance of the node to the sensor network application. Since certain nodes are more critical than others as data generators, using an application-aware routing cost allows the network to identify and avoid these sensors as routers.

The actual cost function depends on the quality of service (QoS) required by the sensor network application. Consider, for example, a sensing application that requires full coverage of a region of interest. In this case, it is important that all subregions contain at least a single active sensor for as long as possible. Thus, sensors that have several redundant neighbors covering the same subregions are less important to the application than sensors that do not have many neighbors that can provide the same data. In this case, an application-aware routing cost should be chosen such that these more important sensors have a higher cost, so they are not selected as routers. One way to achieve this goal is to consider not only the residual energy of the sensor to which the cost is being assigned, but also that of its redundant neighboring sensors, as described in this section. Thus, we propose using "application-aware" routing costs for route selection in wireless sensor networks.

6.1.1 Sensing and Network Model

In this work, we assume an application where the entirety or a portion of a region needs to be monitored by any one or multiple sensors that are within their sensing range of that location. We refer to the complete set of sensors as $S = \{s_1, \ldots, s_{N_s}\}$. If we require the network to perform at some predetermined level of QoS, or fidelity, we can assign a nominal sensing range to the sensors so that sensors can adequately monitor activity within this sensing range (e.g., the signal-to-noise ratio exceeds a given threshold at this range).

In the system model that we are considering, many sensors are deployed in the region to be monitored. A data sink periodically queries the network for data. Sensors in the network then reply to the query and send data, the entirety of which meets the querying node's QoS requirements, back to the querying node. It is often the case that long network lifetime is more important than meeting very high QoS constraints. Furthermore, it may be the case that the cumulative QoS of the data from each sensor is marginally better than what could be provided by a subset of the sensors. In these situations, many of the sensors can choose not to participate in the sensing of data, but remain active to route traffic from those that do participate in sensing data. If one were to observe over time which sensors participate most in the sensing task, it will typically be the case that the sensors deployed in sparse areas with the least coverage redundancy sense the most data, while those deployed in the denser areas with the most coverage redundancy sense the least, since they can share the task with neighboring sensors. Thus, we need to take this into account when assigning the node costs used in route selection.

A common "energy-aware" routing cost used in wireless ad hoc networks is the inverse of a node's residual energy e_i^{res}.

$$C_{ea}(s_i) = 1/e_i^{res} \tag{6.1}$$

With the use of this routing cost, nodes with little energy remaining are unlikely to be used to route the traffic of other nodes and, consequently, this increases the time before the first nodes die. In the application model that we are considering, certain nodes are expected to be used more often than others as data generators, meaning that on average, their energy consumption will exceed that of the other nodes in the network. As the network progresses into its final stages, these nodes will have the lowest remaining energy and will be avoided as routers. However, this happens too late and these nodes may die prematurely, as they are required to generate traffic very often. In order to improve network lifetime, these nodes should be avoided as routers even in the initial stages of the network.

Let $A(s_i)$ represent sensor s_i's coverage area. Because redundancy exists between the coverage of the sensors, each location x is characterized by a sensor set $S(x) \subseteq S$ that is capable of monitoring it. We will denote the total energy of all sensors that have location x within their coverage area as $E(x)$.

$$E(x) = \sum_{s_i \in S(x)} e_i^{res} \tag{6.2}$$

We can define several cost functions based on $E(x)$ that allow nodes to indicate their unwillingness to route traffic even before their residual energy drops significantly below other nodes in the network. While these application-aware costs can be used for the sensor network models considered in this work, other methods for determining application-aware costs may be used for sensor network applications that do not conform to this coverage model. In developing an application-aware routing cost, the general goal is to provide information about the importance of the individual sensors to the sensing application.

6.1.2 Worst-Coverage-Based Cost

In some applications, it may be critical that the entirety of the region being monitored is covered as long as possible. In other words, the utility of the application drops significantly as the coverage falls from 100% to just below 100%. For such situations, we define a worst-coverage-based cost $C_{wc}(s_i)$

$$C_{wc}(s_i) = \frac{1}{\min_{x \in A(s_i)} E(x)} = \max_{x \in A(s_i)} \frac{1}{\sum_{s_j \in S(x)} e_i^{res}} \tag{6.3}$$

This cost assignment method finds the least-covered subregion (in terms of energy) of each node's coverage area and sets the node's cost equal to the inverse of the sum of the energy of the individual sensors capable of monitoring that critical subregion.

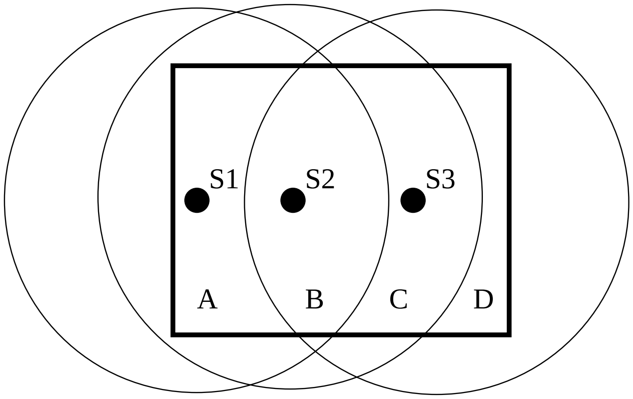


Figure 6.1: Example sensor network. Since s_3 is the only sensor that can cover region D, its worst-coverage-based cost $C_{wc}(s_i)$ is the highest in the network. The comprehensive-coverage-based cost $C_{wc}(s_i)$ gives a more complete encapsulation of a sensor's value to the sensing task and considers the area and redundant energy of each subregion.

Consider the scenario illustrated in Figure 6.1, where the rectangular area is the region to be monitored and sensors s_1, s_2, and s_3 are capable of monitoring the regions within the circles representing their respective sensing ranges. For simplicity, we assume that all sensors have a single unit of energy. Any point in region A, which we will refer to as x_A, can be covered by 2 sensors – s_1 and s_2. Thus, $E(x_A) = 2$ and similarly, $E(x_B) = 3$, $E(x_C) = 2$, and $E(x_D) = 1$. Sensor s_1 can monitor regions A and B and since the coverage in region A is the poorest in terms of total energy, s_1's cost is set to $C_{wc}(s_1) = \frac{1}{E(x_A)} = \frac{1}{2}$. Similarly, $C_{wc}(s_2) = \frac{1}{2}$ and $C_{wc}(s_3) = 1$.

Note that several sensors, whose least redundantly covered portions of the monitored region consist of overlapping portions, will have identical application costs, regardless of their individual residual energy. This follows the intuition of our design, since these sensors are equally effective at monitoring this critical region of the environment.

6.1.3 Comprehensive-Coverage-Based Cost

In some scenarios, the utility of a sensor network application may degrade gracefully with the amount of area that is covered. To account for this, we propose another routing cost $C_{cc}(s_i)$ that considers the comprehensive coverage in the regions that a sensor can monitor instead of the single least-covered region. This comprehensive coverage-based cost is set as a weighted sum of $1/E(x)$, weighted by the area of each subregion. In other words, to obtain $C_{cc}(s_i)$, we integrate the inverse of $E(x)$ over s_i's coverage region.

$$C_{cc}(s_i) = \int_{A(s_i)} \frac{dx}{E(x)} = \int_{A(s_i)} \frac{dx}{\sum_{s_j \in S(x)} e_j^{res}} \tag{6.4}$$

Again, consider the scenario illustrated in Figure 6.1. Sensor s_1 will set its cost as $C_{cc}(s_1) = \int_{A(s_1)} \frac{dx}{E(x)} = \int_A \frac{dx}{2} + \int_B \frac{dx}{3} = \frac{area(A)}{2} + \frac{area(B)}{2}$. Similarly, $C_{cc}(s_2) = \frac{area(A)}{2} + \frac{area(B)}{3} + \frac{area(C)}{2}$ and $C_{cc}(s_3) = \frac{area(B)}{3} + \frac{area(C)}{2} + \frac{area(D)}{1}$. This comprehensive-coverage-based routing cost provides a more balanced view of a node's importance to the sensing task.

6.1.4 Combining Several Cost Functions

So far, we have proposed two application-aware cost functions that capture the importance of individual nodes to the sensing of the environment. However, the usefulness of sensors is not limited to their ability to sense the environment and generate data; they are useful for routing the data of other sensors as well. While the objective of our proposed costs is to use the sensors that are not important as data generators more liberally as routers, some combination of the application-aware costs proposed in this work and a connectivity cost could ensure that these sensors are not used too liberally. Consider a network in which a number of nodes that can serve as routers but do not have any sensing capabilities are deployed in addition to the microsensors that we have considered thus far. Using $C_{wc}(s_i)$ and $C_{cc}(s_i)$ as routing costs, these nodes will be assigned a cost of 0. Large amounts of traffic will be routed through these nodes and sent as far toward the data sink as transmission ranges permit, even when large distances between these nodes and the data sink cause energy inefficient transmissions. Clearly, this routing strategy is not optimal. In an energy-efficient solution, these router-only nodes would be used more conservatively as routers and a greater portion of their energy would be saved for use in the later stages of the network.

Thus, we propose the use of a routing cost that considers a node's importance as a router as well as a data source. Here, we simply use the energy-aware routing cost

$C_{ea}(s_i)$ to help balance the importance of a node. It should be noted that this is a very coarse approximation of the importance of a node to maintaining connectivity, but it is typically a closer approximation than the application-aware routing costs. In future work, we plan to develop a connectivity cost that measures the importance of individual sensors in routing data and maintaining good network connectivity.

We considered several methods for combining the energy-aware routing cost and the comprehensive application-aware routing cost, including the weighted arithmetic mean, the weighted geometric mean, and the weighted maximum. Simulation results have shown that using the maximum value of the worst-coverage-based cost and a weighted value of the energy-aware cost is most effective in extending network lifetime with 100% coverage.

$$C(s_i) = \max(C_{wc}(s_i), \beta C_{ea}(s_i)) \tag{6.5}$$

Similarly, the use of the maximum value of the comprehensive-coverage-based cost and a weighted value of the energy-aware cost is effective in providing long network lifetimes with graceful degradation.

$$C(s_i) = \max(C_{cc}(s_i), \beta C_{ea}(s_i)) \tag{6.6}$$

In each case, the parameter β can be optimally tuned to maximize network lifetime, as we will show in Section 6.2.4.

6.2 DAPR - Distributed Activation with Predetermined Routes

We have designed a simple distributed protocol called DAPR (Distributed Activation with Predetermined Routes) that integrates the services of sensor selection and route discovery. Most architectures proposed for use in coverage-preserving wireless sensor network applications use a modular approach where sensor selection and routing are performed independently. Even in those that use an integrated approach (e.g., [24]), the integration is rather loose, as the sensor selection algorithm considers the effect of the potential routers, but the routers are not chosen with any consideration of the sensor selection algorithm. In the DAPR protocol, route discovery and sensor selection are

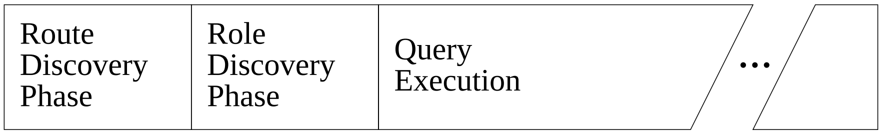


Figure 6.2: The execution of a DAPR query is triggered by a Query message, sent by the data sink. A Route Discovery Phase is followed by a Role Discovery Phase, in which sensors select and deselect themselves to participate in the query's execution. After these phases are completed, the query is executed for some predetermined query length.

performed separately, but decisions made in each process are influenced by the other. The premises for the design of DAPR are twofold — that sensors critical to the sensing applications as data generators should be avoided as routers and that the selection of a sensor for the active sensor set affects its potential routers as well as the sensor itself.

In DAPR, finite-length queries, which are triggered by the sending of Query packets, are processed for a predetermined query length by a subset of the sensors available in the network. Before the query is processed, the network undergoes a Route Discovery Phase followed by a Role Discovery Phase. Upon completion of the Role Discovery Phase, sensors process the query and provide data to the querying node for the duration of the query, as shown in Figure 6.2. In previous work [81], we considered a round-based approach where a data sink collects data for long periods of time, and sends Round Start messages periodically so that roles are updated regularly and energy is balanced throughout the lifetime of the network. The single-query approach proposed here is simply a more generic version of this protocol and can be made equivalent by requiring queries to be sent at the correct interval.

During the Route Discovery Phase, the Query packets are broadcast throughout the network, with one copy of the packet broadcast by each node, so that a spanning tree, rooted at the data sink, is formed. As the packets are flooded throughout the network, each node updates a cost field within the packet, adding the cost of the link to its parent node. Routing costs such as those proposed in the previous section are assigned to individual sensors, and the cost of a link is a weighted sum of the effort that each sensor must put forth to transfer the data. Specifically, the cost of a link is calculated as

$$C_{link}(s_i, s_j) = C(s_i)e_{ij}^{tx} + C(s_j)e_{ij}^{rx} \tag{6.7}$$

where e_{ij}^{tx} represents the energy that is required by s_i to transmit a bit to s_j and e_{ij}^{rx} represents the energy that is required by s_j to receive a bit from s_i. The cumulative cost of a sensor's route is

$$C_{route}(s_i) = \sum_{(s_j, s_k) \in p(s_i, s^*)} C_{link}(s_j, s_k) \tag{6.8}$$

where $p(s_i, s^*)$ represents the set of links along the chosen optimal path from s_i to the sink s^* that minimizes $C_{route}(s_i)$.

After the Route Discovery Phase, each sensor must decide whether or not it is necessary to become active in the Role Discovery Phase. After initially assuming that it will remain active to process the query, each sensor will attempt to deactivate itself if possible by sending a deactivation beacon. To ensure that sensors with the highest route costs are given the highest priority to deactivate, each node backs off before broadcasting its deactivation beacon, with backoff delays set according to a decreasing function of the route costs. The intuition behind prioritizing sensors based on route costs is based on the fact that a sensor's activation affects its potential routers as well as itself. If, after its backoff delay, a sensor infers that its coverage region is entirely covered by its neighbors that have not yet sent a deactivation beacon, the sensor sends its deactivation beacon, becomes inactive, and does not process the query. It should be noted that this deactivation is for sensing purposes only. A node that sends a deactivation beacon must remain available for routing purposes since routes have already been determined by this time.

Implementation Issues

The calculation of our proposed application-aware routing costs assumes that nodes have location information of neighboring nodes with redundant coverage regions. This information can be exchanged between neighbors after being obtained through GPS or any number of proposed location estimation algorithms in the current literature [82–84]. Since DAPR was designed for networks of static sensor nodes, location updates must be performed only a single time at the beginning of network operation, or very infrequently in the worst case. Note that the need for location information is not a drawback of the DAPR protocol or the proposed application-aware routing costs specifically – this information is necessary in any coverage-preserving protocol. Also, very loose time synchronization is required so that nodes can identify the beginning and end of query periods.

The application-aware routing costs also depend on information about the residual energy of neighboring nodes. This information can be conveyed within the Query messages that are forwarded. Before forwarding these messages, which each node should do once per query, a node simply fills in a field in the packet header that is reserved for residual energy information. Since a node must know its own routing cost before forwarding a Query message, it must calculate this value from information obtained during the previous query. As long as the query length is not so long that nodes may use a significant portion of their initial energy during a single query, the residual energy information should not be too stale to calculate near-optimal routes. Alternatively, two packets could be sent during the Route Discovery Phase – the first containing only residual energy information and the second containing route cost information.

We have assumed that nodes are able to begin the dissemination of a query immediately after the data sink broadcasts the initial Query packet. In practice, this means that sensor nodes must listen to the channel in an idle listening mode until receiving these packets. Since it has been shown that power consumption in the idle listening mode is typically comparable to that in the receive mode, this can severely impact network lifetime. If moderate delays are acceptable, then a low power wakeup system may be used to inform nodes about a predetermined time at which the Route Discovery Phase will start [85]. However, idle listening during the Route Discovery and Role Discovery phases is unavoidable, as sensors do not know when their neighbors will send the Query messages and deactivation beacons. Since the Route Discovery and Role Discovery phases are expected to be very short compared to the query length, this will not greatly impact the energy-efficiency of DAPR. Also, this is not a requirement of the application-aware routing costs specifically, and would be required under a similar protocol using other routing costs. For short-lived queries where the Route Discovery and Role Discovery Phases contribute significant overhead in terms of energy consumption, DAPR should not be used.

For normal network operation during the processing of the query, we assume that a schedule-based MAC protocol is used so that idle listening does not contribute significantly to overall energy consumption. The development of such a MAC protocol is beyond the scope of this work, but the reader is referred to [86, 87] for some examples.

The determination of the existence and size of overlapping coverage regions during the calculation of the proposed routing costs and decisions concerning deactivation can potentially be very computationally intensive. Our implementation uses an approximation in which sensors create a grid of locations within their sensing ranges and, point-by-point, observe the redundancy of their neighbors. However, for the deactivation decision, any of the coverage preserving rules described in the current literature could be used in place of this method (e.g., [21, 23]).

The deactivation beacons may be sent over a single hop if it is assumed that the transmission range is at least twice as great as the sensing range. If this assumption is not valid, the beacons must be forwarded through controlled flooding until they reach all sensors that redundantly cover at least some portion of the sending sensor's coverage region (i.e., those within twice the sensing range).

6.2.1 Simulations and Analysis

In this section, we present simulation results measuring the performance of the proposed application-aware routing costs and the DAPR protocol. The simulations in this section were performed using Matlab, and they focus on the routing and application layer while simplifying MAC and physical layer implications.

In these simulations, a data sink sent periodic queries, which were processed by sensors that sent constant bit rate traffic to the sink. We assumed that queries were generated from different locations in the network throughout the network lifetime. This helped to avoid rapid energy drain in the nodes surrounding the data sink.

The energy model that was used in our simulations was similar to that used in [36], in which the energy required by s_i to send a bit to s_j separated by a distance of d_{ij} was

$$e_{ij}^{tx} = E_{elec} + \varepsilon \, d_{ij}^{\alpha} \tag{6.9}$$

where E_{elec} represents the energy associated with the radio electronics, ε characterizes the power amplification component, and α represents the path loss exponent. The energy required by s_j to receive a bit from s_i was

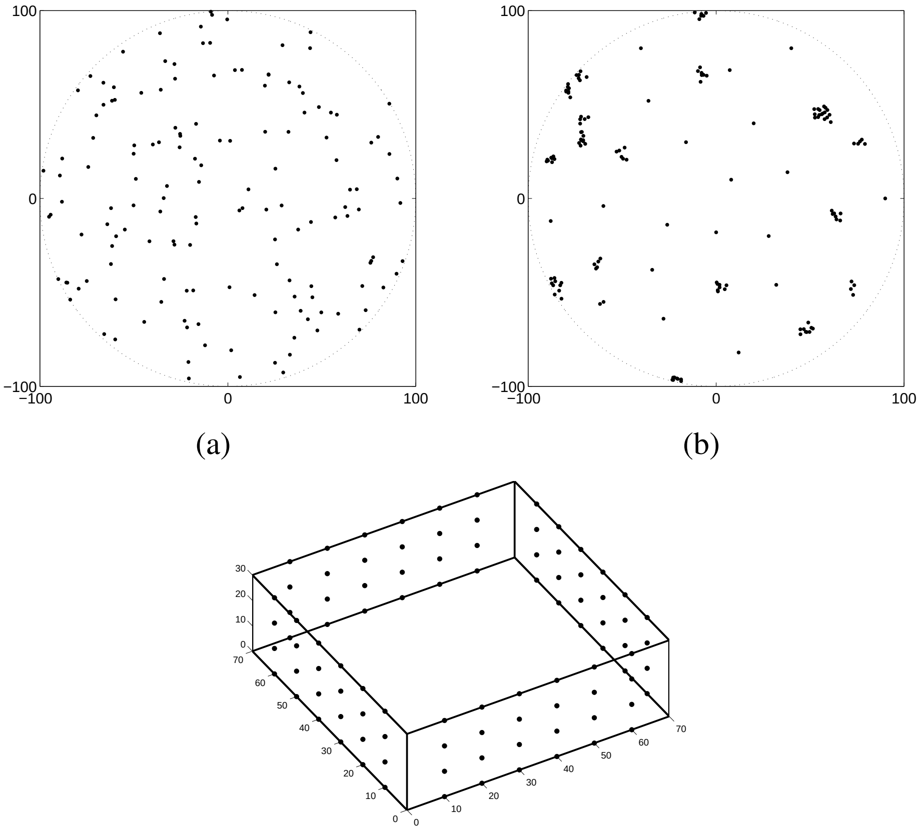
$$e_{ij}^{rx} = E_{elec} \tag{6.10}$$

Under ideal conditions (e.g., very high density), power consumption is minimized by sending packets over distances of d^* [88, 89], where

$$d^* = \sqrt[\alpha]{\frac{2E_{elec}}{(\alpha - 1)\varepsilon}} \tag{6.11}$$

Using our power consumption numbers, given in Table 6.2, the optimal transmission distance d^* was approximately $32\ m$. While ideal conditions were not seen in our simulations and sensors may send traffic along circuitous routes in order to avoid routing through critical sensors, this distance, along with the geographic size of the networks that were simulated, provides some indication of the amount of routing that must be performed on the data generated within the network.

In these simulations, we considered three deployment scenarios. The first was a uniform deployment scenario, in which sensor locations were selected uniformly from a circular deployment region. In this scenario, coverage nonuniformities were generally not very severe. While the application-aware costs were not designed for such networks, we include analysis of their performance in these types of deployments for thoroughness. The second scenario that we considered was a clustered deployment scenario, in which small groups of sensors were deployed in a normal distribution around a number of locations chosen randomly from within the network. In this scenario, more coverage nonuniformities existed as a result of deployment nonuniformity. The third scenario was a video network, in which cameras were mounted in a grid deployment on four walls, each of which was required to be monitored at all times. Each camera was randomly tilted horizontally and vertically between -45 and 45 degrees. The simulations of this scenario helped to measure the performance of the application-aware routing costs when the sensors' physical proximity to each other did not necessarily determine their coverage overlap. Examples of deployment patterns for each scenario are given in Figure 6.3. The coverage nonuniformities are summarized in Table 6.1, which shows the mean and standard deviation of the coverage overlap throughout the region to be monitored. While both the uniform and clustered scenarios have an average overlap of about 9 sensors, the standard deviation is more than twice as high in the clustered scenario.



(a) (b)

(c)

Figure 6.3: Example sensor deployment patterns for the uniform deployment scenario (a), clustered deployment scenario (b), and video scenario (c).

Scenario	Uniform	Clustered	Video
Mean coverage overlap (Number of sensors)	9.0	8.9	6.4
Standard deviation of coverage overlap (Number of sensors)	3.0	6.6	2.2

Table 6.1: Coverage overlap statistics for the three simulated deployment scenarios.

Parameter	Value
Packet Size	$20\ bytes$
Packet Rate	$1\ packet/sec$
α	2
E_{elec}	$50\ nJ/bit$
ε	$100\ pJ/bit/m^2$
Query Length	$24\ hr$
Initial Node Energy	$1000\ J$
Sensing Range (Uniform, Clustered)	$25\ m$
Deployment Radius (Uniform, Clustered)	$100\ m$
Surveillance Radius (Uniform, Clustered)	$90\ m$
Room Width (Video)	70 m
Room Height (Video)	30 m
Sensor Spacing (Video)	10 m
Sensor Field of View (Video)	30 degrees

Table 6.2: Default simulation parameters for DAPR simulations.

The rest of the parameters used in our simulations are summarized in Table 6.2. All simulation results were averaged over 25 trials. It should be noted that we did not compare our results against other coverage-preserving protocols that exist in the literature. The reason for this is that the major contribution of our work is the incorporation of coverage information into the routing protocol. In fact, the way that nodes determine whether or not they need to remain active to preserve coverage in the network is not very important. Any of the coverage-preserving decision algorithms in the literature could be used with an application-aware routing cost.

6.2.2 Performance of Application-Aware Routing Costs

In this section, we analyze the performance of our proposed application-aware routing costs as alternatives to traditional energy-aware routing, where $C(s_i) = C_{ea}(s_i)$, and minimum power routing, where $C(s_i) = 1$, using the DAPR protocol for sensor and router selection. All networks in this section consisted of 150 sensor nodes.

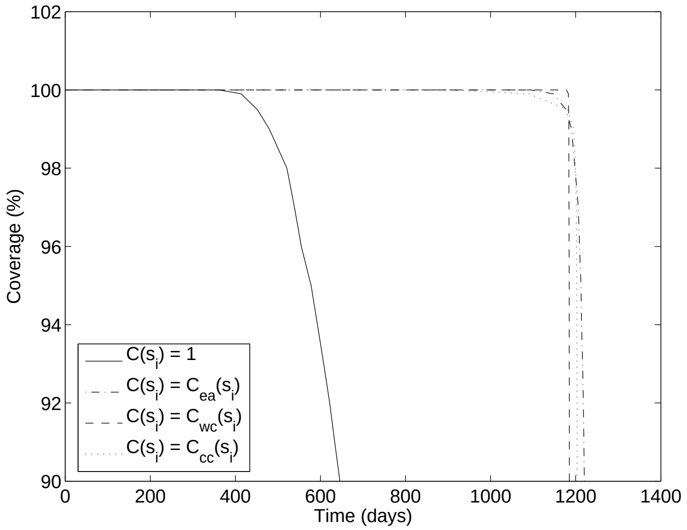


Figure 6.4: Coverage degradation over time for different routing costs in the uniform deployment scenario.

$C(s_i)$	1	$C_{ea}(s_i)$	$C_{wc}(s_i)$	$C_{cc}(s_i)$
100% coverage lifetime (days)	362	1094	1178	904
98% coverage lifetime (days)	521	1198	1184	1200

Table 6.3: Simulation results for different routing costs in the uniform deployment scenario.

Figure 6.4 shows the coverage degradation over time for the uniform deployment scenario. Although the application-aware routing costs were not designed for such networks in which node redundancy is approximately equivalent throughout the network, it can be seen that the application-aware routing costs perform very similar to the energy-aware cost, and even slightly better. From this plot and the results summarized in Table 6.3, we can see that the lifetime before the first break in coverage is highest for the worst-coverage-based routing cost, giving an improvement of 7% over the energy-aware routing cost. Networks using the comprehensive-coverage-based cost, which was designed so that coverage degrades more gracefully, were the last to drop below 98%, although the gain over the energy-aware routing cost was minimal in this scenario.

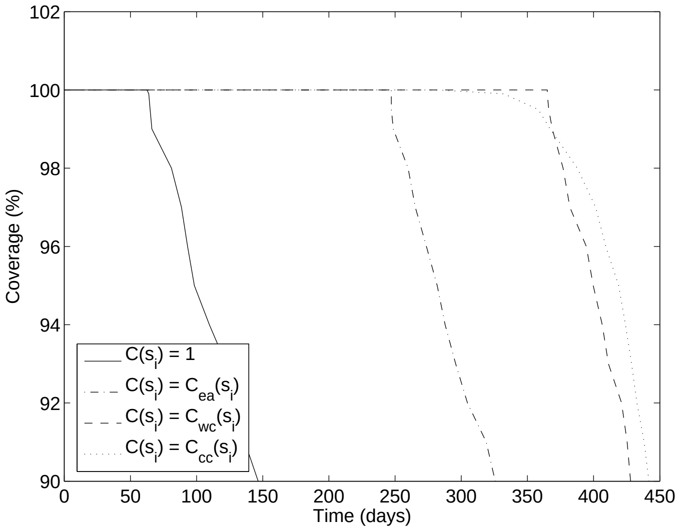


Figure 6.5: Coverage degradation over time for different routing costs in the clustered deployment scenario.

$C(s_i)$	1	$C_{ea}(s_i)$	$C_{wc}(s_i)$	$C_{cc}(s_i)$
100% coverage lifetime (days)	62	247	365	376
98% coverage lifetime (days)	81	260	377	388

Table 6.4: Simulation results for different routing costs in the clustered deployment scenario.

Figure 6.5 and Table 6.4 present the results for the clustered deployment scenario. Because coverage is less uniform throughout the network, the gains that can be obtained from the use of the application-aware routing costs are higher than in the case of the uniform deployment scenario. The worst-coverage-based routing cost gives an improvement of 48% over the energy-aware routing cost in terms of lifetime before the first break in coverage. The comprehensive coverage-based cost gives an improvement of 49% in lifetime before coverage drops below 98% over the energy-aware routing cost.

Figure 6.6 and Table 6.5 present the results for the video scenario. The worst-coverage-based cost gives a significant gain in network lifetime before the first break in coverage, increasing lifetime by 24%. However, the comprehensive-coverage-based

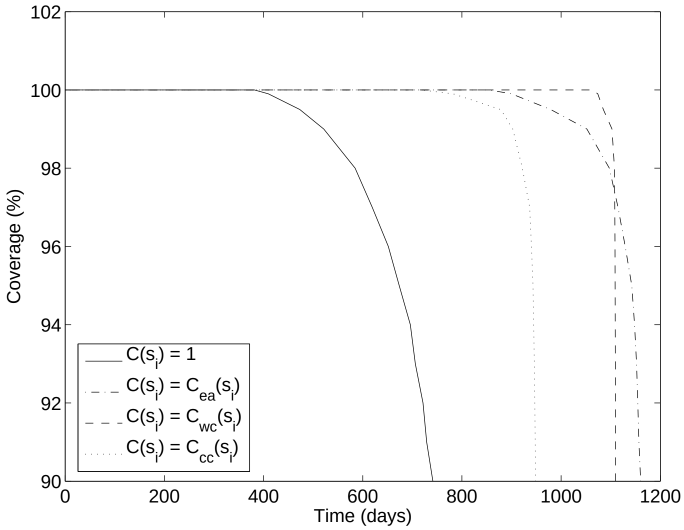


Figure 6.6: Coverage degradation over time for different routing costs in the video scenario.

$C(s_i)$	1	$C_{ea}(s_i)$	$C_{wc}(s_i)$	$C_{cc}(s_i)$
100% coverage lifetime (days)	381	855	1063	717
98% coverage lifetime (days)	585	1097	1108	921

Table 6.5: Simulation results for different routing costs in the video scenario.

cost performs very poorly in this scenario. We suspect that the reason for this is that this cost does not consider the utility of a node as a router, but rather as a sensor only. Nodes that should be kept alive for routing purposes may be used too liberally, causing them to die and forcing other sensors in the region to use suboptimal routes for the remainder of the network lifetime. This is not a problem in the uniform and clustered deployment scenarios since a node's importance as a sensor and as a router are both tied to its location. One way to avoid this problem for video networks is to use a combined routing cost, as discussed in Section 6.2.4.

6.2.3 Effect of Sensor Selection Criteria

In this section, we explore the effect of the sensor selection criteria when using the worst-coverage based routing cost. Recall that sensors are deactivated by sending a

Scenario	Random	Individual Cost	Cumulative Routing Cost
Uniform	1036	1088	1178
Clustered	364	365	365
Video	818	800	1063

Table 6.6: Network lifetime when using the worst-coverage-based routing cost with different selection criteria.

deactivation beacon to neighboring sensors after a backoff timer expires. In these simulations, we compare network lifetime when setting the backoff timer according to three different criteria – randomly, based on the sensor's individual cost, and based on the sensor's cumulative route cost, given in Equation 6.8. As the activation or deactivation of a sensor affects its routers as well as itself, we expect lifetime to be highest when setting the backoff timer according to the cumulative route cost. As shown in Table 6.6, choosing sensors based on their cumulative cost improves network lifetime over using the sensors' individual costs by a modest 8% in the uniform deployment scenario, almost nothing in the clustered deployment scenario, and a more significant 33% in the video scenario. These values can be explained by the fact that nearby sensors typically have very similar routes to the data sink. For this reason, when selecting which sensors to deactivate among multiple nearby sensors that cover the same region, the choice will probably affect only the sensors being deactivated, but few or none of the routers, since they are probably the same for all sensors under consideration. This is the case in the clustered deployment scenario and to less of an extent, in the uniform deployment scenario. However, in the video scenario, two sensors that cover the same region may have very dissimilar routes to the base station. In this situation, the choice of which sensor to deactivate will affect different groups of sensors. Thus, the gain in network lifetime is highest in this scenario.

6.2.4 Combining Routing Costs

In this section, we explore the effectiveness of combining the application-aware routing costs with the energy-aware routing cost in order to account for both coverage and connectivity requirements. We simulated similar networks as in the previous sections as well as heterogeneous networks, in which 150 sensors capable of sensing the environment and generating data for the data sink were deployed along with additional nodes that could only be used to route data (50 for the uniform and clustered scenarios and 32 for the video scenario). We ran simulations in which we set the nodes' routing costs to $C(s_i) = \max(C_{wc}(s_i), \beta C_{ea})$ and others in which we set the nodes' routing costs to $C(s_i) = \max(C_{cc}(s_i), \beta C_{ea})$, as described in Section 6.1.4, and we tuned β to maximize network lifetime.

Network lifetime (for 100% coverage) when setting the routing cost as the combined cost $C(s_i) = \max(C_{wc}(s_i), \beta C_{ea})$ are summarized in Table 6.7. The results show that network lifetime before coverage degrades below 100% is typically maximized or very nearly maximized when β is set around 0.25 in these scenarios. The improvement is most significant in the heterogeneous networks since the router-only nodes' value is most misrepresented by the application-aware cost in this case. In the heterogeneous networks, the use of the combined routing cost with this value of β improves network lifetime by 20% over the use of the worst-coverage-based routing cost and by 17% over the use of the energy-aware routing cost for the uniform deployment scenario. For the clustered deployment scenario, these numbers grow to 21% and 43%, respectively. In the video network, the combined cost improves lifetime by 8% over the use of the worst-coverage-based routing cost and by 20% over the use of the energy-aware routing cost.

The effects of the combined cost are less dramatic in the scenarios that contain only sensing-capable nodes, as in the simulations of the previous sections. The network lifetime improvement is about 4% for the uniform and video scenarios when using a β value of 0.25. Meanwhile, the clustered scenario does not benefit at all from the use of the combined cost.

Network lifetime (for 98% coverage) for each scenario when setting the routing cost as $C(s_i) = \max(C_{wc}(s_i), \beta C_{ea})$ are summarized in Table 6.8. The results show that network lifetime before coverage degrades below 98% is typically maximized or nearly maximized when β is set at $100m^2$ for each scenario. Again, the impact is greatest in the heterogeneous networks. The use of the combined routing cost with this value of β improves network lifetime by 20% over the use of the comprehensive-coverage-based routing cost and by 2% over the use of the energy-aware routing cost for the uniform deployment scenario. In other words, we gain very little in using the combined cost

β	0	0.05	0.25	0.5	1
Uniform	1178	1192	1224	1093	1094
Clustered	365	365	360	245	247
Video	1062	1082	1106	853	855
Uniform (add.routers)	1368	1572	1635	1549	1402
Clustered (add.routers)	476	567	576	525	403
Video (add.routers)	1214	1299	1306	1083	1083

Table 6.7: Network lifetime (days) before first coverage break for heterogeneous networks when using a combination of worst-coverage-based energy-aware routing costs. $\beta = 0$ represents the case when $C(s_i) = C_{wc}(s_i)$, while $\beta = 1$ represents the case when $C(s_i) = C_{ea}(s_i)$.

over using the energy-aware cost alone. For the clustered deployment scenario, these numbers grow to 29% and 28%, respectively. For the video scenario the combined cost improves network lifetime by 42% over the use of the comprehensive-coverage-based routing cost and by 7% over the use of the energy-aware routing cost.

As in the case of the worst-coverage-based cost, the improvements are not as great in the networks containing only sensing-capable nodes.

6.2.5 Comparison With Optimizations

In these simulations, we compared the lifetime achieved using DAPR with the optimal lifetime that can be achieved through the large-scale optimization program described in Chapter 4. In this program, cover sets were chosen using a version of the Garg-Könneman algorithm applied to sensor network selection [70, 71]. Once a large enough group of cover sets, each providing complete coverage of the region, were found, we ran the optimization program outlined in Chapter 4 to find the maximum achievable lifetime. Our simulation results of DAPR with the worst-coverage-based cost are compared with the optimal lifetime in Figures 6.7(a), 6.7(b), and 6.7(c) for the uniform, clustered, and video scenarios, respectively. We compare several approaches here:

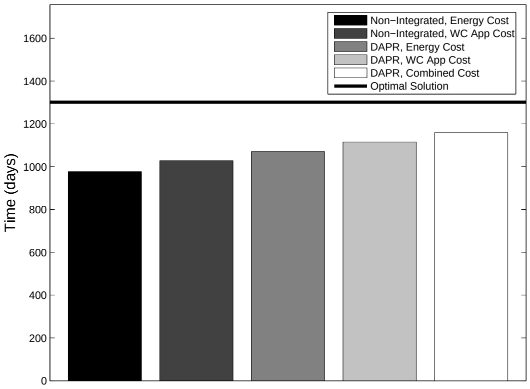
1. a typical non-integrated approach - setting node costs as $C_{ea}(s_i)$ (energy-aware routing) with sensor selection based on the individual sensors' costs,

β	0	$20m^2$	$100m^2$	$400m^2$	∞
Uniform	1203	1210	1240	1212	1212
Clustered	419	419	418	346	281
Video	943	1064	1163	1142	1143
Uniform (add.routers)	1392	1554	1670	1646	1645
Clustered (add.routers)	487	569	629	561	493
Video (add.routers)	1232	1595	1747	1629	1629

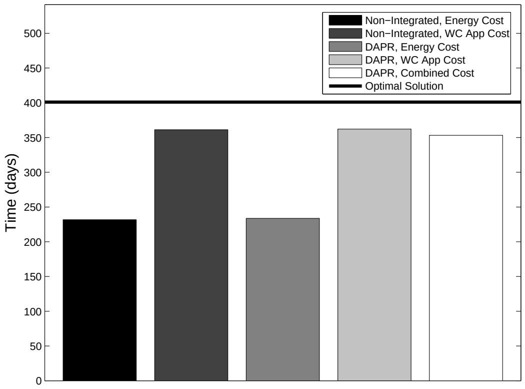
Table 6.8: Network lifetime (days) before coverage drops below 98% for heterogeneous networks when using a combination of comprehensive-coverage-based energy-aware routing costs. $\beta = 0$ represents the case when $C(s_i) = C_{cc}(s_i)$, while $\beta = \infty$ represents the case when $C(s_i) = C_{ea}(s_i)$.

2. the non-integrated approach, but setting node costs as $C_{wc}(s_i)$,
3. DAPR, using energy-aware routing cost $C_{ea}(s_i)$,
4. DAPR, using the worst-coverage routing cost $C_{wc}(s_i)$, and
5. DAPR, using a combination of the energy-aware routing cost and the worst-coverage routing cost, as described in Section 6.1.4, with β set to 0.25.

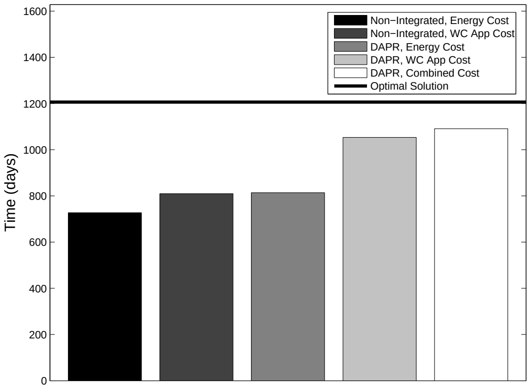
In the uniform scenario, the use of the combined worst-coverage and energy-aware cost with DAPR gives a total network lifetime gain of 14% over the non-integrated approach, closing the gap with the optimal solution by 56%. The results for the clustered deployment scenario show that DAPR with the worst-coverage routing cost performs especially well in this scenario, improving lifetime by 56% and closing the gap between the non-integrated approach and the optimal solution by 77%. Most of this improvement in this scenario is due to the use of the application-aware routing cost rather than the selection of sensors based on the cumulative route cost. Finally, the use of DAPR with the combined worst-coverage and energy-aware cost in the video scenario gives a total network lifetime gain of 50% and closes the gap between the non-integrated approach and the optimal solution by 76%. Most of the improvement in this case comes from the selection of sensors based on the cumulative route cost.



(a) Uniform



(b) Clustered



(c) Video

Figure 6.7: Comparison of DAPR with globally optimal solution calculated using linear programming.

6.3 Summary

In this chapter, we have presented several “application-aware” routing costs that are especially beneficial for sensor networks where sensor deployment is nonuniform. These routing costs are the first that consider the application’s QoS goals and the importance of individual sensors to the application’s requirements. We have also presented DAPR, a distributed protocol for sensor selection and route selection that uses these application-aware routing costs. Simulation results have shown that the application-aware costs can increase network lifetime by almost 50% in some scenarios. While these results indicate the advantages of application-aware costs for convergecast traffic patterns, many wireless sensor networks utilize multicast of data from the sensors to a number of data sinks. In the next chapter, we explore the advantages of using application-aware costs in such a multicast scenario.

Chapter 7

Applying Application-Aware Costs to Multicast Traffic Patterns

In the previous chapter, we showed the advantage of using application-aware routing costs in a network model where sensors route traffic to a single destination within the network (i.e., a static base station that forwards this data to the end user via a high-bandwidth uplink or a mobile robot wandering throughout the network). This network model is frequently assumed in the current sensor network literature. However, we would like to explore the benefits of these application-aware cost in other network models as well. Specifically, we would like to show the gains that can be obtained in a network that employs multicast as its traffic model. Multicasting is a model that is well suited for several sensor network scenarios. For example, in a sensor-actuator network [90], the sensor data often must arrive at several actuator locations within the network so that the actuators can act on the information that the sensors provide. Another viable multicast scenario is one in which several base stations exist in the network, each with a high-bandwidth uplink the end-user. If some of these links are unreliable, it may be advantageous to route the data to several of these base stations so that at least one of them will be able to send the data back to the end user.

7.1 Multicasting in a Wireless Environment

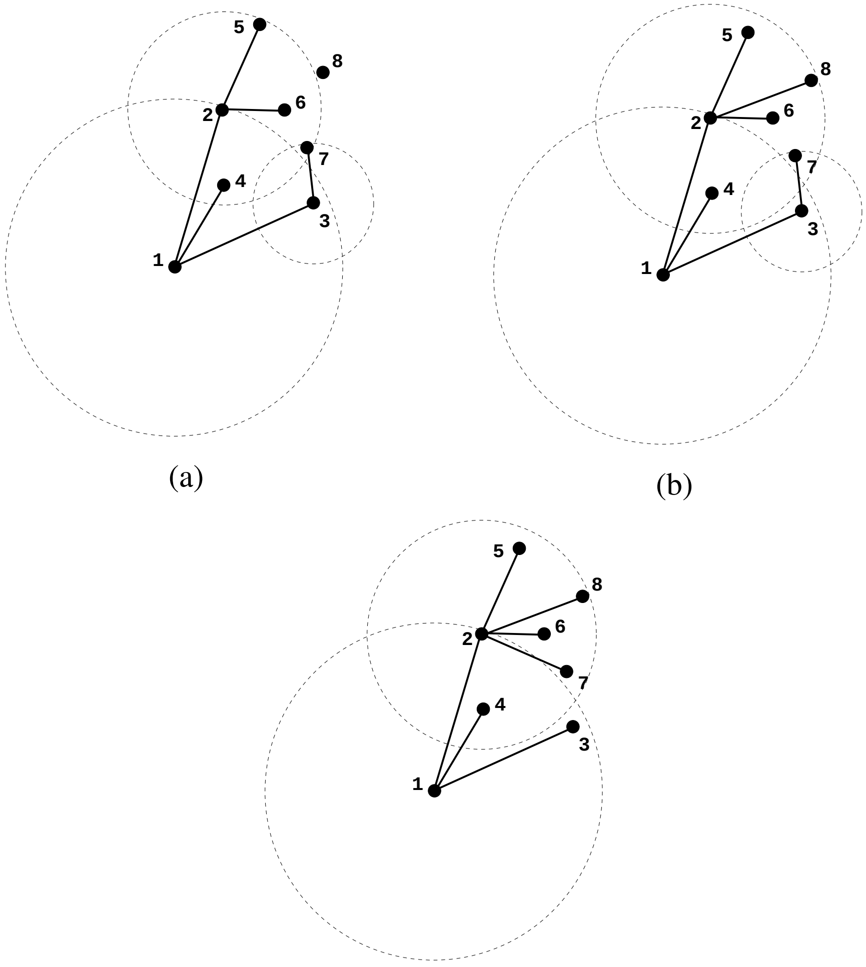
The establishment of optimal multicast trees is a problem that has been studied extensively in the networking community [48]. Traditionally, optimal broadcasting and multicasting has been thought of as the minimization of the edge weights of the multicast trees (i.e., the formation of minimum spanning trees and Steiner trees, respectively). This model is an accurate one for wired networks; however, in wireless networks, nodes can exploit the inherent "wireless multicast advantage" to further reduce multicast tree cost. More recent algorithms for setting up minimum cost multicast trees take advantage of the fact that a single node's transmission can be received by more than just a single intended destination. Thus, these algorithms can be thought of as node-based, rather than link-based minimization algorithms.

The Broadcast Incremental Protocol (BIP) takes advantage of the aforementioned "wireless multicast advantage" to generate low-cost multicast trees in wireless networks [49]. The BIP protocol is based on Prim's algorithm for finding minimum spanning trees (MST). In Prim's algorithm, nodes are added iteratively to the spanning tree starting with the source node. During each iteration, the node not included in the current tree with the minimum link cost to a node included in the current tree is chosen for addition to the tree. In BIP, nodes are also added iteratively to the tree, starting with the source node. However, when determining the node with the minimum cost, BIP considers the incremental cost to join the tree. In other words, if a node is already transmitting at a certain power level to reach some neighbor, its *incremental* cost is the additional cost required to increase its transmission power to the level necessary to reach the additional node. Thus, the incremental cost is given as

$$C_{ij}^{inc} = P'_{ij} = P_{ij} - P(i) \tag{7.1}$$

where P_{ij} represents the power required to send from node i to node j and $P(i)$ represents the transmission power at which node i is currently set to transmit.

Consider the network shown in Figure 7.1(a). At this stage in the spanning tree's construction, node 1's downstream neighbor set includes nodes 2, 3, and 4. Meanwhile, node 2's downstream neighbor set includes nodes 5 and 6 and node 3's downstream neighbor set includes node 7. Node 2 is the furthest node in node 1's neighbor set and node 1 must transmit at power level $P(1) = P_{1,2}$. Similarly, node 2 must transmit at power level $P(2) = P_{2,5}$ and node 3 must transmit at power level $P(3) = P_{3,7}$. In order for node 1 to reach node 8, it must increase its transmission power to the level $P_{1,8}$ for an incremental cost of $P'_{1,8} = P_{1,8} - P_{1,2}$. Similarly, $P'_{2,8} = P_{2,8} - P_{2,5}$ and $P'_{3,8} = P_{3,8} - P_{3,7}$. For all other nodes i, $P'_{i,8} = P_{i,8}$. In our example, the minimum



(a)

(b)

(c)

Figure 7.1: Example of a BIP tree construction.

incremental cost is attained by node 2 and its transmission power is increased during the next iteration, allowing the network to become fully connected as shown in Figure 7.1(b).

Following construction of the broadcast tree, a “sweeping” operation is performed on the network in order to remove unnecessary transmissions. In the sweep operation, all leaf nodes are analyzed to determine if their current power level will allow neighboring nodes to reduce their transmission power. Each node in turns looks at any downstream neighbors that are reachable with its current transmission power. If any of these neighbors’ parents would benefit from their children switching parents to the current node, this transition is made and the old parent's transmission power is reduced to the minimum level that will allow it to reach its remaining neighbors.

Consider the network shown in Figure 7.1(b). Since node 1 does not reach any neighbors except its own downstream neighbors in the broadcast trees, it has no options to participate in the sweep operation. Since node 2's transmission power allows it to reach node 7 as well as its downstream neighbors, node 3's transmissions are redundant and thus can be removed. Node 2 adds node 7 as a downstream neighbor and node 3 is allowed to cease transmitting. The new tree following the sweep operation is shown in Figure 7.1(c).

The MIP protocol is based on the BIP protocol and uses a pruning procedure to remove unnecessary transmissions. After establishment of the broadcast tree, links that are not necessary to reach the multicast group members are removed. Each node transmits at the minimum power level required to reach each of its neighbors on the remaining outbound links.

We have made modifications to the MIP protocol to allow for the incorporation of node-based costs and to allow for the consideration of receiver-based costs, which exist in real-life hardware. When determining the incremental cost for a given node, we replace Equation 7.1 with Equation 7.2.

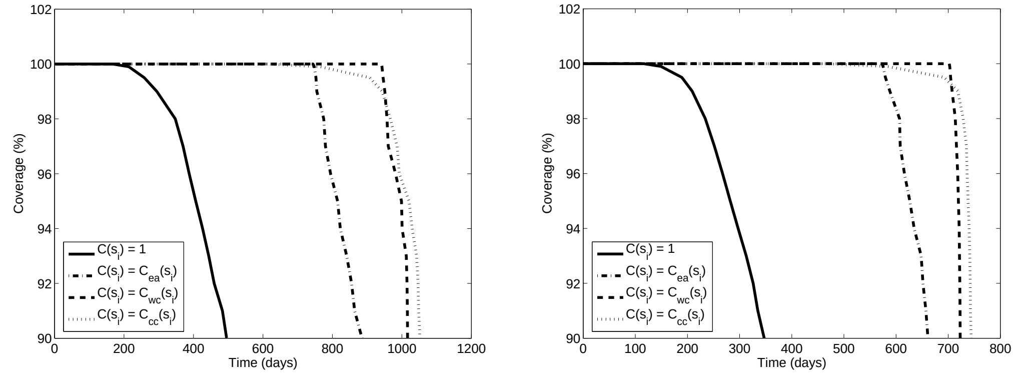
$$C_{ij}^{inc} = C(s_j)e^{rx} + C(s_i)\{e_{ij}^{tx} - \max_{k:s_k \in N(s_i)} e_{ik}^{tx}\} \tag{7.2}$$

Here, $N(s_i)$ represents the current downstream neighbor set of sensor s_i and $C(s_i)$ represents the individual node cost.

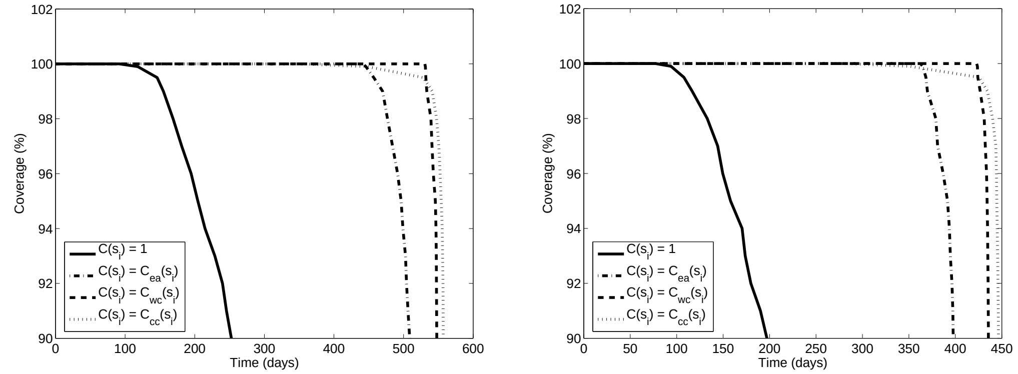
7.1.1 Simulations

Simulation results have shown that the coverage-aware costs proposed in Chapter 6 can extend the lifetime of multicast networks over the lifetimes that can be achieved using simple energy-aware costs. In our simulations, we used similar energy and network models as we did in Section 6.2.1. We varied the number of multicast sinks from 1 to 8 to observe the effect of the different routing costs on network lifetime in the clustered sensor deployment scenario.

Figures 7.2(a)-(d) show the coverage degradation over time for the different multicast group sizes. From this plot and the results summarized in Table 7.1, we can see that



(a) 1 sink (b) 2 sinks



(c) 4 sinks (d) 8 sinks

Figure 7.2: Coverage degradation over time for different routing costs in the clustered deployment scenario (multicast).

the lifetime before the first break in coverage is highest for the worst-coverage-based routing cost. However, even the use of energy-aware costs within MIP, which had not previously been explored, significantly increases lifetime over the use of the standard MIP. The worst-coverage-based routing cost gives improvements of 26%, 23%, 20%, 17% over the energy-aware routing cost for 1, 2, 4, and 8 multicast sinks, respectively. Networks using the comprehensive coverage-based cost, which was designed so that coverage degrades more gracefully, were the last to drop below 98%, giving improvements of 25%, 20%, 15%, and 16% over the energy-aware routing cost for 1, 2, 4, and 8 multicast sinks, respectively.

$C(s_i)$	1	$C_{ea}(s_i)$	$C_{wc}(s_i)$	$C_{cc}(s_i)$
$N_{sinks} = 1$				
100% coverage lifetime (days)	168	744	941	616
98% coverage lifetime (days)	348	775	957	967
$N_{sinks} = 2$				
100% coverage lifetime (days)	116	573	702	464
98% coverage lifetime (days)	234	607	714	729
$N_{sinks} = 4$				
100% coverage lifetime (days)	92	442	531	355
98% coverage lifetime (days)	169	477	539	548
$N_{sinks} = 8$				
100% coverage lifetime (days)	77	362	423	283
98% coverage lifetime (days)	133	379	431	440

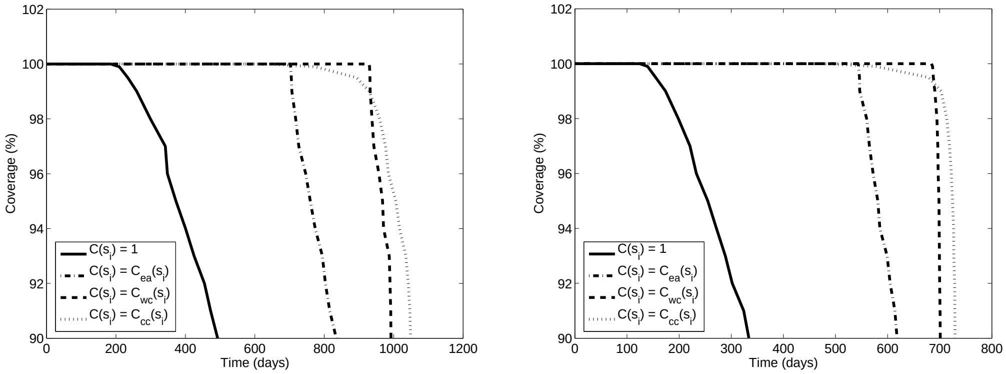
Table 7.1: Simulation results for different routing costs in the clustered deployment scenario (multicast).

7.2 Distributing the Multicast Algorithm

Following the development of the BIP and MIP protocols, attempts have been made to distribute protocols for multicasting in a wireless environment [50, 51, 91–94]. Among these is Dist-BIP [50], of which there are two versions – Dist-BIP-A, which is a slightly better approximation to the centralized version, and Dist-BIP-G, which requires less overhead. We will focus on the latter. In Dist-BIP-G, selected nodes form a local broadcast tree using knowledge of their first and second neighborhood only. The broadcast tree is started from the source node, which generates its local broadcast tree and selects its first hop neighbors that are required to reach its second hop neighbors as gateways. The local tree and the IDs of the gateways are broadcast to its local neighborhood. The nodes selected as gateways, in turn, perform a similar operation, and this process continues until all of the nodes in the network are included in the broadcast tree. If a node receives multiple messages indicating that it should be the child of more than one parent, it chooses its parent based on the first message, but continues to aggregate children according to subsequent messages. Like in the centralized version, Dist-MIP-G is identical to Dist-BIP-G in the early stages, and is followed by a pruning procedure, which is easily distributable, to reduce unnecessary transmissions. We implemented a modified version of Dist-MIP-G to test whether our application-aware routing costs would still provide some improvement over the simpler energy-aware costs even in a distributed multicast protocol. The modifications were similar to those used in the centralized version, with each gateway building its local broadcast tree using similar rules and costs.

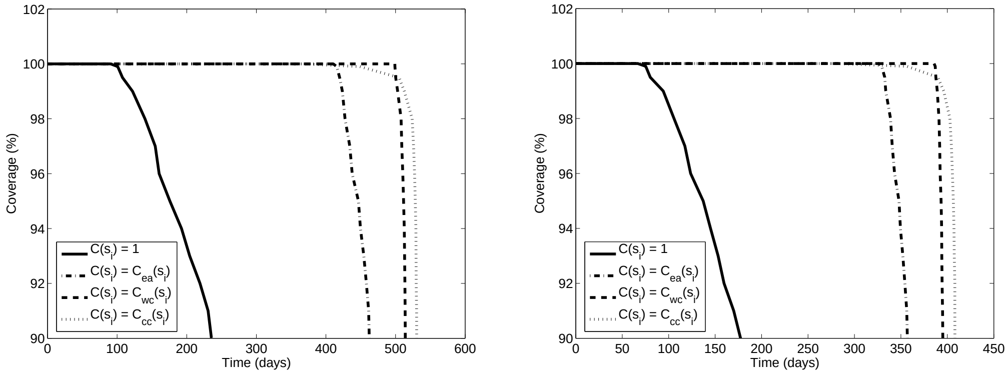
7.2.1 Simulations

Figures 7.3(a)-(d) show the coverage degradation over time for the different multicast group sizes when using Dist-MIP-G. From this plot and the results summarized in Table 7.2, we can see that the lifetime before the first break in coverage is highest for the worst-coverage-based routing cost. Again, the energy-aware cost performs well compared to standard MIP. The worst-coverage-based cost gives improvements of 32%, 26%, 22%, and 18% over the energy-aware routing cost for 1, 2, 4, and 8 multicast sinks, respectively. Networks using the comprehensive coverage-based cost, which was designed so that coverage degrades more gracefully, were the last to drop below



(a) 1 sink

(b) 2 sinks



(c) 4 sinks

(d) 8 sinks

Figure 7.3: Coverage degradation over time for different routing costs in the clustered deployment scenario (distributed multicast).

98%, giving improvements of 34%, 27%, 23%, and 19% over the energy-aware routing cost for 1, 2, 4, and 8 multicast sinks, respectively.

7.3 Summary

In this chapter, we have shown how the “application-aware” routing costs proposed in the previous chapter can be incorporated into networks with alternative traffic models. Specifically, we have incorporated these costs into the MIP protocol for multicasting in a wireless environment and shown that network lifetime can be extended significantly when using these costs over simpler costs based solely on the nodes’ residual energy, depending on the number of multicast sinks. We have also incorporated these costs into

$C(s_i)$	1	$C_{ea}(s_i)$	$C_{wc}(s_i)$	$C_{cc}(s_i)$
$N_{sinks} = 1$				
100% coverage lifetime (days)	185	703	929	642
98% coverage lifetime (days)	299	718	937	960
$N_{sinks} = 2$				
100% coverage lifetime (days)	124	543	684	478
98% coverage lifetime (days)	198	560	695	713
$N_{sinks} = 4$				
100% coverage lifetime (days)	91	410	499	369
98% coverage lifetime (days)	140	427	508	524
$N_{sinks} = 8$				
100% coverage lifetime (days)	67	326	386	281
98% coverage lifetime (days)	106	339	391	403

Table 7.2: Simulation results for different routing costs in the clustered deployment scenario (distributed multicast).

a distributed version of MIP and shown even greater network lifetime improvement.

Chapter 8

Optimal Transmission Power Control for Sensor Networks

So far in this thesis, we have presented several protocols to allow energy-efficient operation of the network while meeting application requirements. Previous chapters have focused primarily on routing and sensor selection decisions to meet these requirements while operating in an energy-efficient manner. In addition to application requirements, nodes must adapt to the environment in which they operate. In this section, we explore the situation in which the deployment environment is not ideal but nodes must make the best use of their resources in the given situation. Specifically, we consider networks with a single sink, where a "hot spot" around the data sink forms as a result of a convergecast traffic pattern. Transmission power and traffic distribution are optimized to extend network lifetime over simple routing policies.

A great deal of attention has been given to the reduction of unnecessary energy consumption of sensor nodes in areas such as hardware design, collaborative signal processing, transmission power control polices, and protocols at all levels of the network stack, as exemplified by the work in the previous chapters of this dissertation. However, reducing an individual sensor's power consumption alone may not always allow networks to realize their maximal potential lifetime. In addition, it is important to maintain a balance of power consumption in the network (assuming uniformly distributed, equally important sensors) so that certain nodes do not die much earlier than others, leading to unmonitored areas in the network.

Early work in transmission range optimization assumed that forwarding data packets towards a data sink over many short hops is more energy-efficient than forwarding over a few long hops, due to the nature of wireless communication. The problem of setting transmission power to a minimal level that will allow a network to remain connected has been considered in several studies [39, 40]. Later, others noted that because of the electronics overhead involved in transmitting packets, there exists an optimal non-zero transmission range, at which power efficiency is maximized [88]. The goal of these studies is to find a fixed network-wide transmission range. When using such a fixed transmission range in many-to-one (convergecast) sensor network applications, however, an energy imbalance problem manifests itself, as a hot spot is created around the data sink, or base station. The nodes in this hot spot are required to forward a disproportionately high amount of traffic and typically die at a very early stage. If we define the network lifetime as the time when the first subregion of the environment (or a significant portion of the environment) is left unmonitored, then the residual energy of the other sensors at this time can be seen as wasted.

Intuition leads us to believe that the hot spot problem can be solved by varying the transmission range among nodes at different distances to the base station so that energy consumption can be more evenly distributed and the lifetime of the network can be extended. In this chapter, we consider the problem of optimizing transmission ranges among nodes for a given deployment scenario. We formalize the transmission range distribution optimization problem, which is solved by determining how a node should distribute its outgoing data packets over multiple distances, always using the minimum transmission power necessary to send over each distance. Given the energy constraints and data generation rate of each sensor node, the lifetime of the network can be maximized by using this optimal distribution.

Table 8.1 lists the parameters that we use in this chapter, including those used in the general network model, the power model, the lifetime model, and the cost model.

8.1 Assumptions

In our model, we make several basic assumptions. First, we assume that the power consumption of sensor nodes is dominated by communication costs, as opposed to sensing and processing costs. This assumption is reasonable for many types of sensor nodes that require very little energy, such as pressure and temperature sensors. Another

Table 8.1: Variables used in network modeling throughout this chapter.

Network Model	
N_s	Total number of sensors
s_i	Sensor i, $i \in \{1, \cdots, N_s\}$
s_0	Base station (single sink)
λ_a	Minimum sensor coverage density
λ_e	Energy density
A	Network area
r_i	Traffic generation rate of sensor i
e_i^{init}	Initial energy of sensor i
$e^{init,total}$	Total initial energy
d_{ij}	Distance between sensors i and j
d^{max}	Maximum transmission distance
Power Model	
E_{tx}	Energy consumption for each bit transmitted
E_{rx}	Energy consumption for each bit received
E_{elec}	Energy consumption from electronic overhead
ϵ_{amp}	Transmitter amplifier coefficient
α	Path loss exponent
Lifetime Model	
e_i	Energy consumption of sensor i
t_{ij}	Amount of traffic that i forwards to j
d_{ij}^t	Transmission distance between sensors i and j
L	Sensor network lifetime
$\tilde{L}$	Normalized sensor network lifetime
Cost Model	
C	Overall deployment Cost
C_s	Cost of sensors
C_e	Extra cost

assumption that we make is that the network is small enough so that all nodes can directly reach the data sink using their maximum setting of transmission power.

We also ignore the overhead that would typically be introduced by the routing layer. However, for long lasting sensor networks with little or no mobility, route updates should be performed infrequently and should not significantly affect the overall power consumption in the network.

We also ignore any potential overhead at the MAC layer. Due to the scarce energy supplies in sensor nodes, many have proposed the use of coordinated TDMA scheduling in the MAC layer. Because of the low data rates expected in many sensor network applications, even localized TDMA scheduling (as opposed to globally coordinated scheduling) should not cause much communication overhead in the form of collisions and retransmissions. Furthermore, TDMA scheduling can eliminate most overhead introduced by idle listening and overhearing. As with the overhead associated with routing updates, the establishment of schedules can take place very infrequently and should not contribute significantly to overall power consumption.

In this work, we also assume that channels are lossless. We can assume that the minimum energy necessary to transmit over a link is the minimum energy such that the packet loss rate is below some small threshold. However, it may be possible to reduce transmission energy further and introduce some non-zero packet loss rate as long as overall power savings are achieved. These factors may contribute to interesting effects, which is the subject of future research.

8.2 Lifetime Optimization

We adopt as a common lifetime definition the time when the first sensor dies. This lifetime definition, proposed in [27], is widely utilized in the sensor network research field. An alternative lifetime definition that has been used is the time at which a certain percentage of total nodes run out of energy. This definition is actually quite similar in nature to the one we use here. In a well designed network, sensors in a certain area exhibit similar behaviors to achieve energy balance. In other words, when one sensor dies, it can be expected that the neighbors of this node will run out of energy very soon since they will have to take over the responsibilities of that sensor. Therefore, in a well designed network, there should be little or no difference in lifetime when using these two definitions.

In our network model, a set of N_s sensors is deployed in a region in order to monitor some physical phenomenon. We refer to the complete set of sensors that has been deployed as $S = \{s_1 \ldots s_{N_s}\}$. Sensor i generates traffic at a rate of r_i bits per second. All of the data that is generated must eventually reach a single data sink, labeled s_0. We adopt the power model from [36], where the amount of energy to transmit a bit can be represented as $E_{tx} = E_{elec} + \epsilon_{amp} d^{\alpha}$, and the amount of energy to receive a bit can be represented as $E_{rx} = E_{elec}$, where E_{elec} represents the electronics energy, ϵ_{amp} is determined by the transmitter amplifier's efficiency and the channel conditions, d represents the distance over which data is being communicated, and α represents the path loss exponent. The network scenario parameters also include the traffic generation rate r_i for each sensor, the distances d_{ij} between sensors, and the maximum transmission distance d^{max}.

The goal of our network lifetime model is to discover the maximum network lifetime L given a fixed deployment strategy and network scenario parameters. The model is able to determine this maximum by optimizing the amount of traffic that each sensor should distribute to the other sensors in order to balance energy consumption among the sensors. This traffic distribution is denoted by t_{ij}, indicating the amount of traffic that sensor i transmits to sensor j. Note that $t_{ii} = 0$ for all i.

During network lifetime L, sensor i will generate a total of $r_i L$ traffic. The first constraint of our problem, related to the conservation of data flow at all sensor nodes, is

$$\sum_{j=1}^{N_s} t_{ji} + r_i L = \sum_{j=0}^{N_s} t_{ij} \qquad \forall i \in \{1, \cdots, N_s\} \tag{8.1}$$

Equation 8.1 states that the sum of all traffic received at sensor i and generated by sensor i must be transmitted to other sensors or to the data sink (s_0). The energy consumed by sensor i includes the energy required for both transmitting and receiving data and can be expressed as

$$e_i = \sum_{j=0}^{N_s} \left(E_{elec} + \epsilon_{amp} (d_{ij}^{t})^{\alpha}\right) t_{ij} + \sum_{j=1}^{N_s} E_{elec} t_{ji} \tag{8.2}$$

The second constraint, related to the initial energy at each sensor, e_i^{init}, is,

$$e_i \leq e_i^{init} \qquad \forall i \in \{1, \cdots, N_s\} \tag{8.3}$$

The third constraint, related to the maximum transmission range d^{max} of each sensor, depends on the sensors' transmission power control capabilities. If the sensors can vary transmission power to accommodate the distance over which they must transmit, then the transmission power required to deliver a packet from sensor i to sensor j will be controlled in such a way that the transmission distance d_{ij}^t equals the physical distance In many real radio transmitters that employ transmission power control, transmission power can only be set at a number of discrete levels, rather than at any arbitrary continuous value. However, these levels can be rather finely spaced, and we model the sensors as being able to set their transmission power arbitrarily to simplify analysis.

$$d_{ij}^t = d_{ij} \qquad \forall i \in \{1, \cdots, N_s\}, \forall j \in \{0, \cdots, N_s\} \tag{8.4}$$

If nodes must use a fixed transmission range d^{max}, then the constraint simply becomes

$$d_{ij}^t = d^{max} \qquad \forall i \in \{1, \cdots, N_s\}, \forall j \in \{0, \cdots, N_s\} \tag{8.5}$$

Note that if $d_{ij} > d^{max}$, then no traffic can be sent between sensors i and j and the following constraint must be applied.

$$t_{ij} = 0 \qquad \forall (i, j) : d_{ij} > d^{max} \tag{8.6}$$

The last constraint, related to the energy distribution at each sensor, depends on how freely energy can be assigned to each sensor. If energy can be freely assigned, then the total energy consumption of all sensors must simply satisfy

$$\sum_{i=1}^{N_s} e_i^{init} = e^{init,total} \tag{8.7}$$

If sensors are initially assigned the same amount of energy, then

$$e_i^{init} = \frac{e^{init,total}}{N_s} \qquad \forall i \in \{1, \cdots, N_s\} \tag{8.8}$$

The optimal network lifetime can be obtained using a linear programming approach that sets the constraints as in Equations 8.1, 8.3, 8.4 (or 8.5), 8.6, and 8.7 (or 8.8), and sets the goal of maximizing L. The linear program finds the maximum lifetime L for

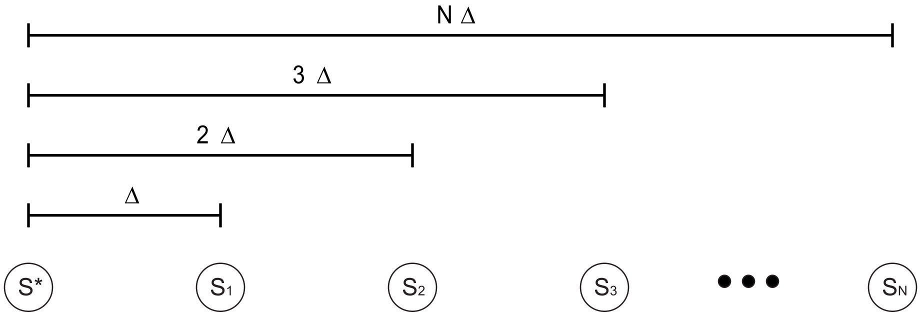


Figure 8.1: A one-dimensional topology network. Nodes are equally spaced in this scenario.

a given scenario, and it also discovers the traffic distribution t_{ij}, indicating how this lifetime can be obtained through intelligent traffic distribution.

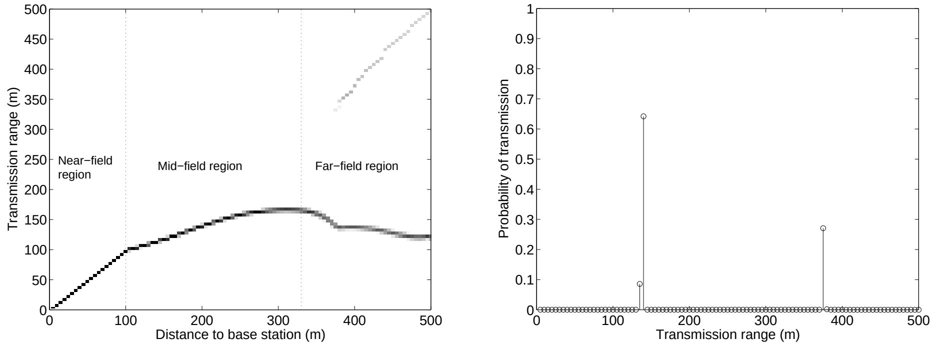
8.3 Simulations

In this section, we provide simulation results in order to observe the optimal transmission range distributions for several typical sensor network deployment scenarios. In all simulations, we used values of $E_{elec} = 50\ nJ/bit$ and $\epsilon_{fs} = 10\ pJ/bit/m^2$ in the power model. According to this power model and the analysis provided in [88], the ideal transmission range for nodes in a general ad hoc network would be $100m$. In other words, $100m$ is the most power-efficient operating point in the absence of the hot spot problem. The initial energy and data generation rate of all nodes in the network were arbitrarily set to $1\ J$ and $1\ bit/second$, respectively, in all simulations.

8.3.1 One-Dimensional Scenario

First, we observe a special case of the transmission range distribution optimization problem where the topology consists of a one-dimensional deployment of sensors, separated by a distance of Δ, leading to the data sink. This scenario may occur in such applications as highway traffic congestion monitoring. The scenario is depicted in Figure 8.1.

The optimal transmission range distribution for a one-dimensional network with node spacing of $5m$ and a radius of $500m$ is shown in Figure 8.2. Figure 8.2(a) illus-



(a

(b)

Figure 8.2: Optimal transmission range distribution for a one-dimensional network with a $500m$ radius and $5m$ spacing between nodes. Figure (b) shows the transmission range probability density function of the node located $400m$ from the base station.

trates the optimal transmission range distribution as a function of distance to the data sink. Dark areas in the distribution images refer to areas of dense transmission probability. By taking a vertical cross section of the image data, as shown in Figure 8.2(b) for a node at a distance of $400m$ from the base station, we can more easily interpret the data. A cross section gives us the probability density function $p_i(d)$ for a sensor s_i located at the position where the cross section is taken. For example, Figure 8.2(b) shows that using optimal scheduling, the node located $400m$ from the base station should send 9% of its traffic over a distance of $135m$, 64% of its traffic over a distance of $140m$, and 27% of its traffic over a distance of $375m$.

As shown in Figure 8.2(a), the sensors can be separated into three regions: a near-field, a mid-field, and a far-field. In the near-field region, nodes transmit directly to the base station. Nodes at farther distances, in the mid-field region, transmit most of their packets over multiple hops, and the distribution of the packets is concentrated at a distance that increases approximately linearly with distance from the base station. Packets from nodes in the far-field region are split between being sent over an energy-efficient transmission range and directly to the base station.

While extending lifetime is the primary goal of the transmission range distribution optimization problem, we consider energy usage in our analysis as well. Lifetimes approaching that of the optimal solution can often be attained when using only a fraction of the total network-wide energy available. If we allocate the same amount of initial energy to each individual node but use just a fraction of the total network energy, we find an interesting trend in the attainable lifetime, shown in Figure 8.3. In fact, as the total energy consumed in the network decreases from 100%, changes in the lifetime are initially very limited. Figure 8.4 illustrates the changes in the optimal transmission range distribution and power consumption for each node at different points on the energy-lifetime curve. The energy inefficiency is caused primarily by the nodes in the far-field region and the nodes at the border of the near-field and the mid-field regions.

The inefficiency of the transmissions in the far-field region can be explained as follows. Nodes closer to the base station in general must forward more traffic than those far away. The nodes in the farthest region need to send only their own traffic, and so in an energy efficient solution (in which they transmit over reasonably small transmission ranges), they consume the least energy. In order to balance the energy and increase lifetime by a minimal amount, they may take several of the packets that were sent over the "reasonable" transmission ranges and send them over longer distances (e.g., directly to the base station), thus reducing the load on the intermediate nodes. However, because of the long distances, the number of additional packets that are sent over longer distances rather than using the "reasonable" transmission ranges is limited. As the plots in Figure 8.4 show, packets that are sent over longer distances rather than using the "reasonable" transmission ranges are the primary ones to be transferred to a more energy-efficient range (see the regions of the distributions enclosed by the dashed lines as well as the energy consumption plots) as the total network-wide energy consumption drops from 100% to 90%.

Nodes at the border of the near-field and mid-field regions also contribute to the energy inefficiency by transmitting packets to nodes located at very small distances from the base station. Consider a node far from the base station that sends its packets to a node located close to the base station. The total energy consumption for such packet transmissions would be slightly higher than the energy consumption to send packets directly to the base station. However, in the absence of these two-hop packet transmissions, the farther node consumes more energy than the closer node and is more of a limiting factor in terms of network lifetime. Thus, while requiring the nearer node to forward traffic increases total energy consumption, it minimizes the maximum energy consumption among the nodes and is included in the optimal solution. Figure 8.4 (see the regions enclosed by the solid lines) illustrates that as the total network-

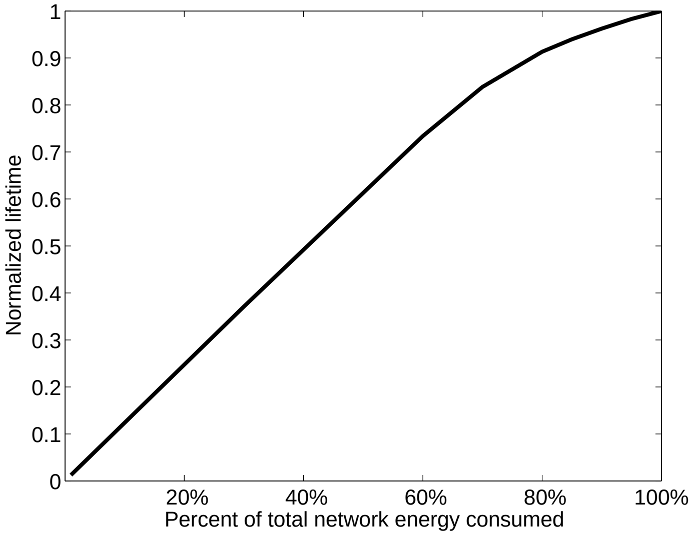
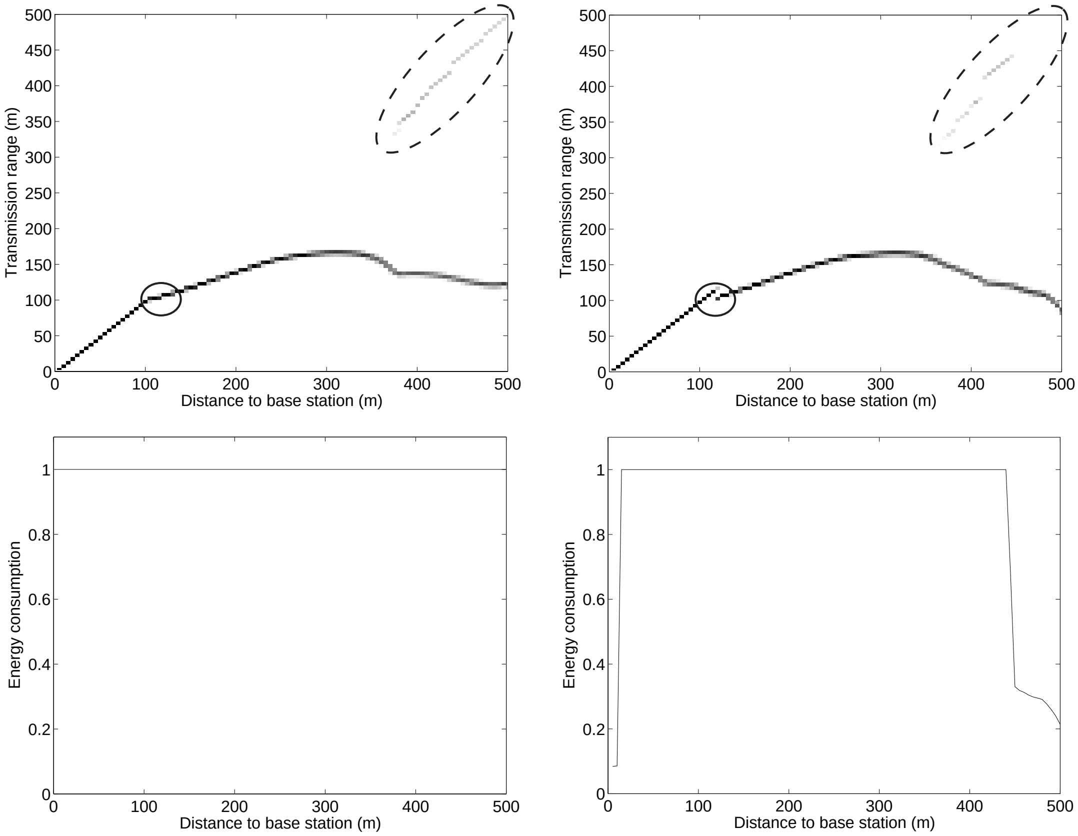


Figure 8.3: Lifetime vs. percentage of total network energy consumed.

wide energy consumption drops from 100% to 90%, some of the packets that would have been sent over a long-hop followed by a short-hop are instead sent over a single slightly longer hop. Note that the reduction in energy consumption actually occurs at the nodes that would have received these packets (i.e., those closest to the base station), rather than at the nodes that would have sent them.

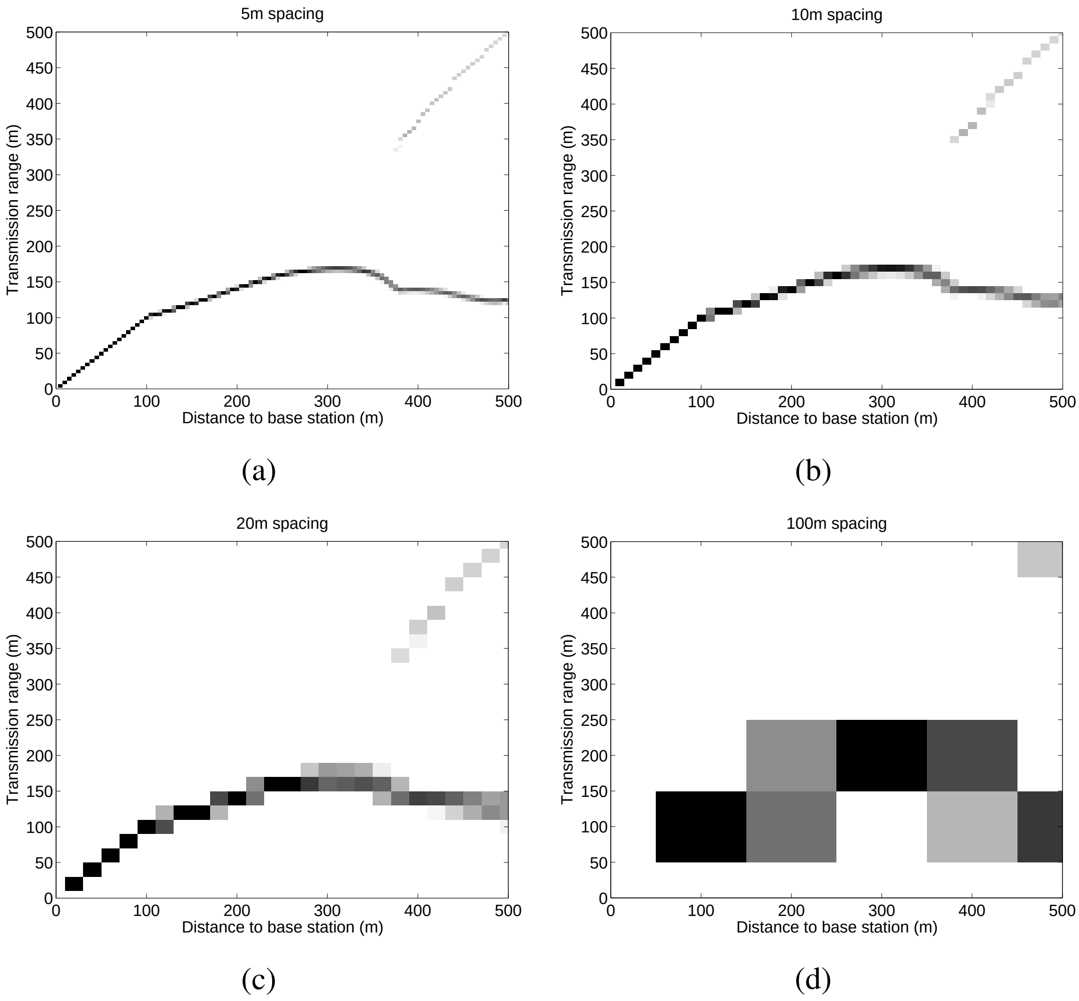
Next, we study the impact of node density on the optimal transmission distribution, which we expected would be minimal. As density increases, the ability of individual nodes to rotate activity or generate data at a lesser rate lengthens lifetime if intelligent schemes are used; however, if the density is increased uniformly throughout the network, the additionally deployed nodes should exhibit similar trends in terms of transmission range distribution. Figure 8.5, which shows the optimal transmission range distribution for networks with a radius of $500m$, with $5m$, $10m$, $20m$, and $100m$ spacing between the nodes, verifies our intuition. The only anomaly is in the $100m$ spacing scenario, where the large spacing prevents transmissions over optimal distances from being realized. In scenarios with higher node density, the optimal transmission range distributions are very similar, as expected.

Next, we observe the effect of network radius on network lifetime and optimal transmission range distribution. In Figure 8.6, we show the optimal transmission range distribution as we set the network radius at $250m$ and $1000m$, using $5m$ sensor spac-



(a) (b)

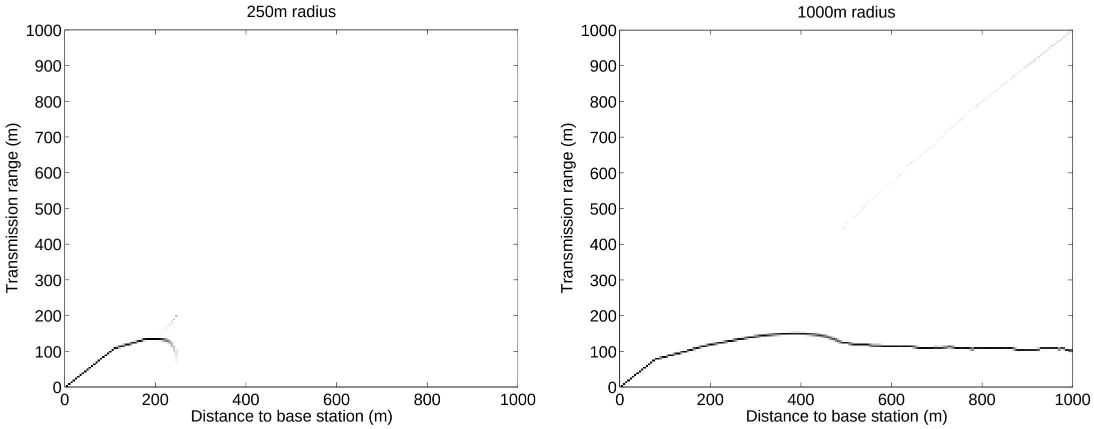
Figure 8.4: Optimal transmission range distribution and energy consumption as a function of distance for 100% (a) and 90% (b) of total network energy consumed.



(a) (b)

(c) (d)

Figure 8.5: Optimal transmission range distribution as a function of distance from the base station for $5m$ (a), $10m$ (b), $20m$ (c), and $100m$ (d) spacing, $500m$ radius.



(a)

(b)

Figure 8.6: Optimal transmission range distribution as a function of distance from the base station for radius of $250m$ (a) and $1000m$ (b), $5m$ sensor spacing.

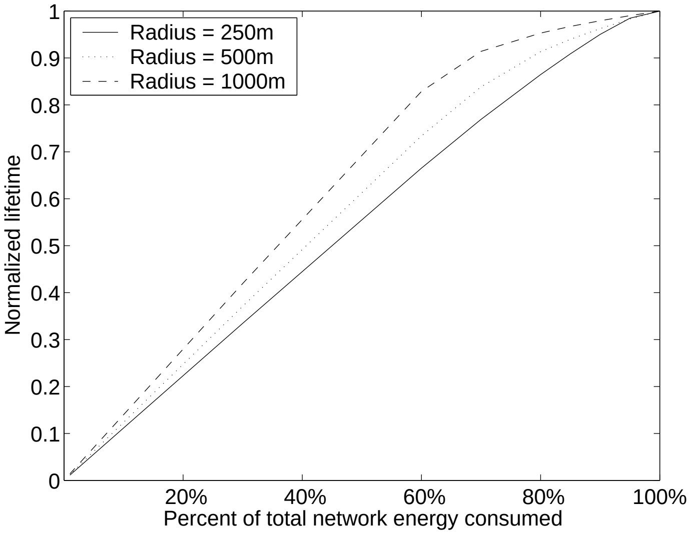
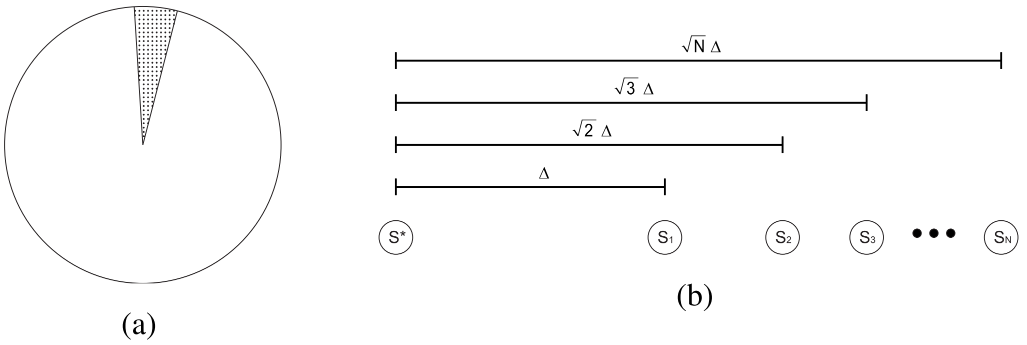


Figure 8.7: Lifetime vs. energy consumption tradeoff for network radius of $250m$, $500m$, and $1000m$.



(a)

(b)

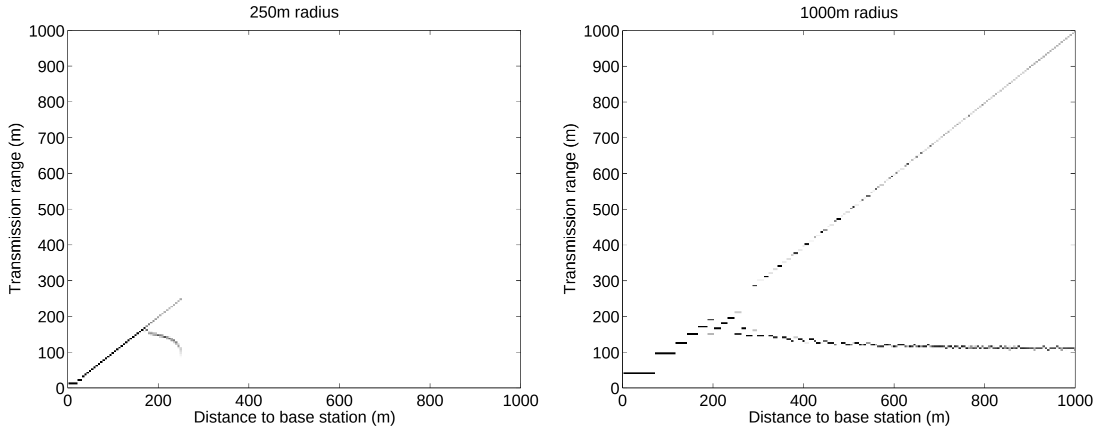
Figure 8.8: Two-dimensional sensor field (a) and one-dimensional modeling (b).

ing. The energy-lifetime curve is plotted in Figure 8.7. In larger networks, the farthest nodes must operate at much more inefficient points in order to balance energy consumption and so inefficiency occurs earlier as the percentage of network energy consumed increases.

8.3.2 Two-Dimensional Sensor Fields

Next, we consider the scenario of a two-dimensional sensor field, with a base station located in the center of the field. We modeled a dense two-dimensional field as a one-dimensional field with nonuniform spacing. With infinitely dense sensor deployment, we can assume that sensors will always send their packets within an infinitesimally thin angle toward the base station, as shown in Figure 8.8(a). Since the number of nodes n within distance r from the base station satisfies $n \propto r^2$ for two-dimensional networks, when mapped into the one-dimensional space, the distance of a node to the base station should be proportional to the square root of the node index, as shown in Figure 8.8(b).

We ran simulations to find the optimal transmission range distribution for a two-dimensional sensor field. In these simulations, the number of nodes in the network was kept constant and their spacing scaled up according to the network radius. The optimal transmission range distributions are shown in Figure 8.9 for network radii of $250m$ and $1000m$. The trends seem similar to those in the one-dimensional scenario, although it seems that a larger portion of the packets are sent over long distances. This is to be expected since there are disproportionately fewer nodes near the base station to route data and disproportionately more data being generated by nodes far from the base



(a) (b)

Figure 8.9: Optimal transmission range distribution as a function of distance from the base station for a two-dimensional sensor field with radius of $250m$ (a) and $1000m$ (b).

station. Since it is inefficient to send packets over long distances, we can expect the energy-lifetime curve to reflect higher inefficiency at points of high energy consumption. Figure 8.10 verifies this and shows that inefficiency occurs at even lower points of total network-wide energy consumption than in the one-dimensional scenario, especially for large networks.

A similar optimal transmission range distribution is also observed even in networks where infinitely high density is not assumed. In our next simulations, we randomly placed 150 sensors in a circular field. We ran 25 simulations and calculated the optimal transmission range distribution for each simulation. The mean transmission range distribution is given in Figure 8.11. In the far-field region, nodes in the randomly deployed network transmit more packets directly to the base station than in the ideally dense scenario. This is because the nodes often do not have another node directly on line to the base station at the optimal distance. For the same reason, among the transmissions that are sent over multiple hops, the transmission distances vary quite a bit, blurring this region in the optimal transmission range distribution.

Overall, in a two-dimensional space, nodes are more densely located in the areas farthest from the base station, aggravating the hot spot problem even further. As a result, the energy inefficiency in the far-field region becomes more obvious than in the one-dimensional case.

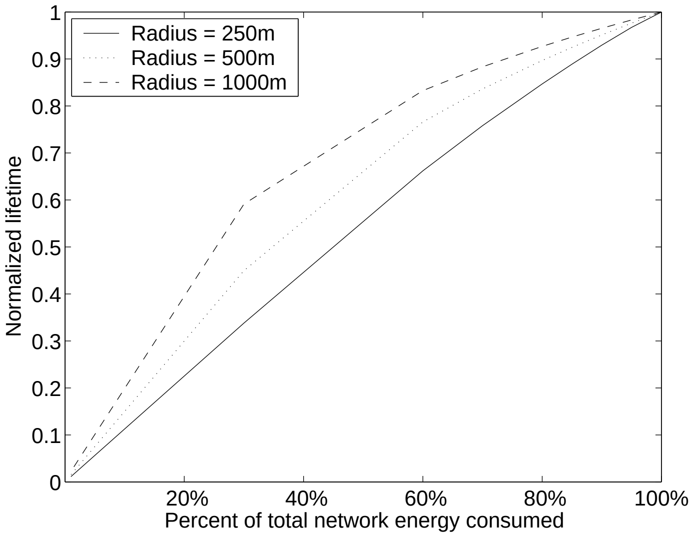
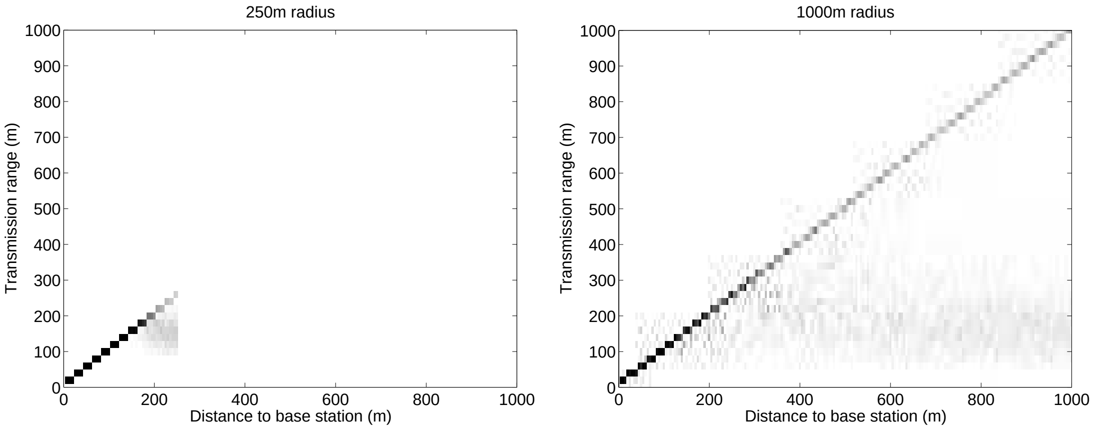


Figure 8.10: Lifetime vs. percentage of the total energy consumed in the network for a two-dimensional sensor field with various network radii.



(a) (b)

Figure 8.11: Optimal transmission range distribution as a function of distance from the base station for a randomly deployed two-dimensional sensor field with radius of $250m$ (a), $500m$ (b), and $1000m$ (c). The trends are similar to those in the ideally dense model, except that the distributions are blurred.

8.4 Summary

In this chapter, we have formalized the transmission range distribution optimization problem for one-dimensional and two-dimensional networks and shown how to obtain the maximum lifetime for convergecast networks using linear programming. Analysis of the solution shows that any network lifetime improvement over simpler transmission range policies that can be obtained through optimization comes at the expense of energy-inefficiency and a wasting of system resources. This is especially true in large networks and two-dimensional networks. The results of this chapter motivate the research presented in the following chapter, in which we present a complete cost analysis of the single-sink convergecast deployment strategy as well as a number of other, more energy-efficient deployment strategies.

Chapter 9

Cost-Efficiency of Sensor Network Deployment Strategies

In the previous chapter, we showed the limited extent to which the hot spot problem that arises in wireless sensor networks can be mitigated when the data sink is a static node. In this chapter, we argue that transmission power control alone is not enough to solve the hot spot problem and that data sink rotation is necessary for the network to operate in an energy-efficient manner. Here, we explore the problem of optimizing data sink rotation and observe the benefits that can be obtained from deploying even a few extra data sinks (or moving the data sink a few times during the lifetime of the network). Alternatively, it may be possible to distribute the processing of sensor network data to regional cluster heads within the network. This mitigates the hot spot problem significantly and increases the energy-efficiency of the system, as data is sent over many fewer hops on average. However, as the deployment of a moving data sink or a few aggregator nodes may be more expensive than deployment of an ordinary sensor node, the optimal deployment method should be considered from a cost perspective. Here, we explore this tradeoff and propose cost-efficient deployment solutions.

We begin by discussing several different strategies for sensor network deployment and some assumptions we make in order to model the network for these different deployment strategies.

9.1 Deployment Strategy Options

Several key parameters can be used to describe sensor network deployment strategies. These parameters include the following.

1. Sensor capabilities. In some cases, sensors have a non-adjustable transmission power and thus a fixed transmission range, while in other cases, sensors equipped with more advanced transceivers may vary their transmission ranges by using different transmission powers.

2. Base station options. Some sensor networks are deployed with a fixed base station that cannot change its physical location. However, another deployment option is to utilize a mobile base station that changes its physical location over time. A third option is to deploy multiple base stations, where each base station can collect data from a portion of the network.

3. Initial energy assignment. The initial energy assignment for each sensor reflects how much freedom a sensor network deployment strategy has. When the deployment is in a controlled manner, nodes can be assigned different levels of initial energy depending on their locations and their roles in the network. For general sensor network deployments, however, we usually assume that the initial energy of all the sensors is the same. This might be true especially when sensors are manufactured in large quantities without differentiation.

4. Sensor locations. Similarly, the locations of sensors, relay nodes and data sinks depend on how much freedom a sensor network deployment has. If the deployment is under full control, more sensors can be placed where energy is needed, and relay nodes can be placed in areas likely to receive the most traffic.

5. Traffic generation pattern. The traffic generation pattern is closely related to the sensing application. For environmental monitoring applications (e.g., temperature monitoring), sensors may generate samples at the same rate. The traffic generation pattern is uniform in this type of network. For intruder detection applications where an intruder is expected to be detected at the farthest end from the base station, more traffic is expected to be generated at far distances. The traffic generation pattern is thus non-uniform in this case.

Table 9.1: Sensor network deployment strategies, corresponding scenarios, and potential difficulty/extra costs.

Strategy	Scenario: {Traffic, Sensors, Energy, Sink}	Difficulty/Extra Costs
DS_1: Single static sink	{uniform, homogeneous, uniform, single/static}	
DS_2: Mobile data sink	{uniform, homogeneous, uniform, **single/mobile**}	Data sink mobility
DS_3: Multiple data sinks	{uniform, homogeneous, uniform, **multiple/static**}	Extra data sink deployment
DS_4: Non-uniform energy	{uniform, homogeneous, **non-uniform**, single/static}	Individual energy assignment
DS_5: Non-uniform placement	{uniform, **heterogeneous**, uniform, single/static}	Sensor/relay placement
DS_6: Non-uniform traffic	{**non-uniform**, uniform, uniform, single/static}	Case dependent

A good network deployment strategy should resolve energy imbalance while maintaining high energy-efficiency. We list some potential sensor network deployment strategies in Table 9.1, labeled as DS_1 through DS_6. We do not intend to list every possible deployment strategy in Table 9.1, but rather merely to highlight some possible solutions to achieve both energy balance and energy efficiency.

The ultimate goal for sensor deployment is to provide a certain quality of service for a maximum lifetime using a minimum cost. Although the more complex deployment strategies listed in Table 9.1 may provide much longer network lifetimes, the extra cost of sensor hardware, base station hardware, and incurred deployment complexity may lead to a disproportionate increase in deployment cost. While maximizing network lifetime is most often the desired research goal, the ultimate goal for a real sensor network deployment plan is to reduce network deployment cost per network lifetime without sacrificing quality of service. Therefore, cost must be considered along with network lifetime during the analysis of different deployment strategies.

9.2 Normalized Network Lifetime

While the lifetime L found in the previous chapter allows us to determine an absolute maximum time that the network can operate, this value is highly dependent on the network scenario parameters, including the network area, the required density of active sensors, the energy density, and the data generation rate. In order to obtain a more general understanding of the energy-efficiency of different deployment strategies, we propose a normalized network lifetime $\widetilde{L}$, which measures how many total bits can be transported on the network per unit of energy. Similar sensing tasks should result in the same normalized network lifetime for a given sensor network deployment strategy.

A typical sensing task can be described as the requirement to monitor an area, providing a certain quality of service, for a certain period of time. For example, suppose that we want to monitor the temperature of a region for one year with a temperature sample rate of once per hour. Design parameters of this task include the average traffic generation rate among active sensors ($\bar{r}$), the minimum sensor coverage density λ_a, the initial energy assigned to each node ($\bar{e^{init}}$), and the monitoring period or network lifetime (L). These parameters affect the absolute lifetime, and they should be factored out during the calculation of the normalized network lifetime. Note that energy-efficiency of each deployment strategy is dependent on the area of the region being monitored, and so we do not attempt to remove this factor.

In typical sensor networks, the network designer can calculate the minimum number of sensors that are required to cover an area for a given application and required quality of service. We denote this minimum sensor coverage density as λ_a. Sensors may be deployed more densely than the sensing application requires and allow their sensing activity to be rotated while maintaining the same sensing coverage goals [20, 23, 81, 95]. Once the network is fully covered, network lifetime can be arbitrarily increased by simply putting more energy into the network. This can be realized by scaling up the deployed sensor density, or increasing the initial energy per sensor. Network lifetime can also be increased by reducing the traffic generation rate $\bar{r}$ among active sensors. A normalized lifetime $\widetilde{L}$ that accounts for the total energy consumption by considering the above factors can be expressed as

$$\widetilde{L} = L\left(\frac{\bar{r}\lambda_a}{\lambda_e}\right) \tag{9.1}$$

where λ_a represents the minimum sensor coverage density, $\bar{r}$ represents the average bit rate among active sensors, λ_e represents the energy density of the network (i.e., how much energy is available per unit area), and L is the lifetime achievable with the given scenario's parameters.

In terms of units, L is measured in seconds, $\bar{r}$ is measured in bits per second, λ_a is measured in the number of sensors per square meter, and λ_e is measured in Joules per square meter. $\widetilde{L}$ is thus measured in terms of bits per Joule, which explicitly indicates the energy efficiency of a particular deployment strategy for a given network scenario.

9.3 Cost Model

Normalized lifetime reflects the energy-efficiency of different deployment plans. From the normalized lifetime, we can deduce the number of sensors that will need to be deployed in order to meet the sensing requirements, giving some indication of the cost to deploy the network. For a particular deployment strategy DS_i, given the sensing requirements and a target network lifetime goal L, we can calculate the number of required sensors N_s as follows.

$$N_s(DS_i) = \begin{cases} \min(\lambda_a A, \frac{L\bar{r}\lambda_a A}{\widetilde{L}e^{\bar{init}}}) & i = 1, 2, 3, 5 \\ \lambda_a A & i = 4 \end{cases} \tag{9.2}$$

For deployment strategies DS_1, DS_2, DS_3, and DS_5, where each node has a uniform data generation rate $\bar{r}$ and a uniform initial energy $e^{\bar{init}}$, Equation 9.2 determines the number of sensors that are needed based on the normalized lifetime $\widetilde{L}$ as well as the application quality of service (sensing density). For deployment strategy DS_4, Equation 9.2 simply specifies that the minimum number of sensors that support the application quality of service (sensing density) should be deployed since unequal energy assignment can be used to ensure that the lifetime goal is met. In our cost analysis of strategies DS_1, DS_2, DS_3, and DS_5, we will assume that the application quality of service (sensing density) constraints are always met and that network lifetime is the driving factor when determining how many sensors should be deployed.

More energy-efficient deployment strategies will have a higher normalized lifetime (i.e., they will carry more traffic per unit of energy) and thus require a lower number of sensors $N_s(DS_i)$ to meet the target lifetime. Thus, the deployment cost from sensors, $C_s(DS_i)$, is lowered. However, these complex strategies may have higher extra deployment cost $C_e(DS_i)$. Our cost model explores these extra costs that are often overlooked, and it enables the evaluation of different deployment strategies from a monetary cost perspective. The total cost for the sensors is $C_s(DS_i) = c_s N_s(DS_i)$, where c_s represents the cost of an ordinary microsensor, and the overall deployment cost $C(DS_i)$

becomes

$$C(DS_i) = C_e(DS_i) + C_s(DS_i) \tag{9.3}$$

This cost model is a simple yet effective method for allowing a network designer to compare different deployment strategies on an equal basis.

9.4 Comparison of Different Deployment Strategies

In the previous section, we showed that optimal traffic load distribution with transmission power control is not very effective in extending network lifetime in some scenarios. However, for deployment strategy DS_1, this is the only option for extending network lifetime. In this section, we investigate how well each of the other strategies listed in Table 9.1 improves network lifetime. We will evaluate these strategies using the general normalized lifetime and deployment cost models, defined in Sections 9.2 and 9.3.

When determining the normalized lifetime for each deployment scenario, we use an arbitrary sample scenario that is manageable in terms of memory and processing for solving the linear programs. In the sample scenario, $N_s = 180$ nodes are deployed in a disc with a radius of 250 m, and sensors send traffic at a rate of $\bar{r} = 1$ bit/s. 180 Joules of total energy is assigned to the sensors. The network parameters for the sample scenario are summarized in Table 9.2. Once the values of $\widetilde{L}$ have been determined for each deployment strategy via analysis of this sample scenario, they can be used to compare the cost-efficiency of different deployment strategies for a larger scale target scenario. For each deployment strategy, we find the normalized lifetime with and without transmission power control. For the scenarios without transmission power control, we perform a brute force search over all possible fixed transmission ranges so that we find an upper bound for that scenario using any transmission power.

9.4.1 Deployment Strategy DS_1: Single Static Sink

We begin our study of the optimized network lifetime for the simplest, most common sensor network deployment scenario, DS_1 in Table 9.1. For this deployment strategy, the only option to reduce the effects of the hot spot problem and maximize network lifetime is to employ intelligent traffic distribution.

Table 9.2: Network parameters for the sample scenario.

Parameter	Value
Network Radius	250 m
N_s	180
λ_a	$\frac{180}{\pi 250^2}/m^2$
$e^{init,total}$	180 J
λ_a	$\frac{180}{\pi 250^2} J/m^2$
$\bar{r}$	1 bit/sec

We assign nodes the same initial energy $e_i^{init} = 1$ Joule and the same traffic generation rate $r_i = 1$ bit per second. In all simulations and analysis, we use values of $E_{elec} = 50\ nJ/bit$ and $\epsilon_{amp} = 100\ pJ/bit/m^2$ [36]. We assume that nodes can adjust their transmit power large enough to transmit to sensors $100m$ away (i.e., $d^{max} = 100m$). We solve a linear program with Equations 8.1, 8.3, 8.4, 8.6, and 8.8 as the constraints and a goal of maximizing L. The linear program solution provides us with the maximum achievable network lifetime for this strategy. This lifetime is shown in Figure 9.1 for various network radii.

In the optimal traffic distribution matrix, nodes that are very close to the base station simply transmit all of their traffic directly to the base station. Nodes at farther distances transmit most of their packets over multiple hops and send a smaller share of their packets directly to the base station over long distances.

To investigate the improvement that transmission power control provides, we compared its lifetime with the lifetime of a fixed transmission power scheme, which we found using an optimization program with Equations 8.1, 8.3, 8.5, 8.6, and 8.8 as the constraints. The results are shown in Figure 9.1. For each network radius, the optimal fixed transmission range was found using a brute force search. While it seems that transmission power control greatly improves energy-efficiency, analysis shows that the improvements of Figure 9.1 are a result of inefficiency of the last-hop transmissions when transmission power control is not used. The sensors transmitting along the last hop have their power set unnecessarily high compared with the required level to reach the base station. To isolate the effect of this, we also propose and analyze a heuristic

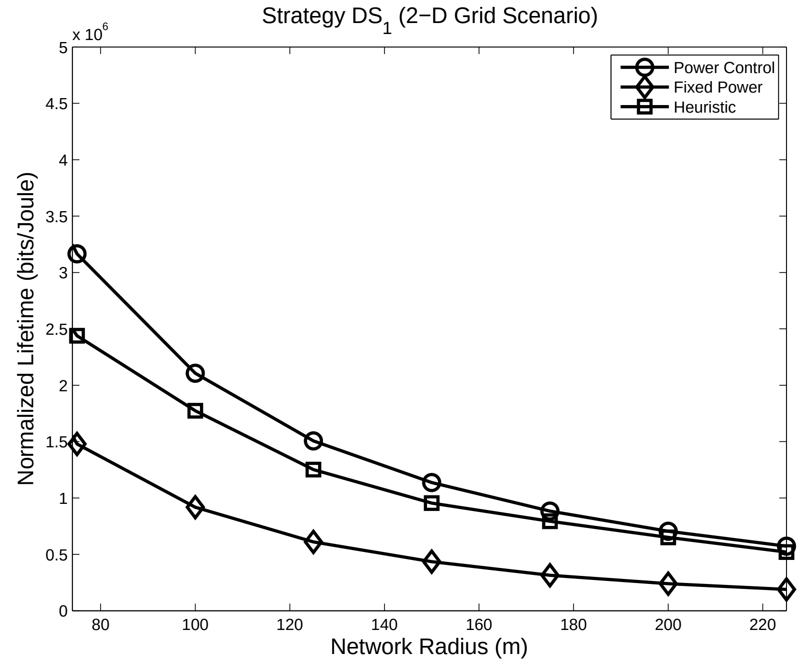


Figure 9.1: Network lifetime as a function of network radius for the optimal power control scheme, the fixed transmission power scheme and the heuristic power control scheme in a two-dimensional scenario.

power control scheme in which nodes transmit using a fixed transmission power over most hops, while using transmission power control for the last hop. To model this heuristic scheme, the constraint imposed by Equation 8.5 must be modified as follows.

$$d_{ij}^{t} = \begin{cases} d^{max} & j \in \{1 \dots N\} \\ d_{ij} & j = 0 \end{cases} \tag{9.4}$$

The lifetime performance of this heuristic power control scheme is shown in Figure 9.1. The optimal transmission distances d^{max} for each network radius are obtained through brute force searches on all possible transmission ranges. The heuristic power control scheme performs somewhere between the optimal power control scheme and the fixed scheme.

The normalized lifetime for a two-dimensional network utilizing the parameters in the sample scenario (Table 9.2) was found to be 4.69×10^5 bits/J when using transmission power control and 1.62×10^5 bits/J when not using transmission power control.

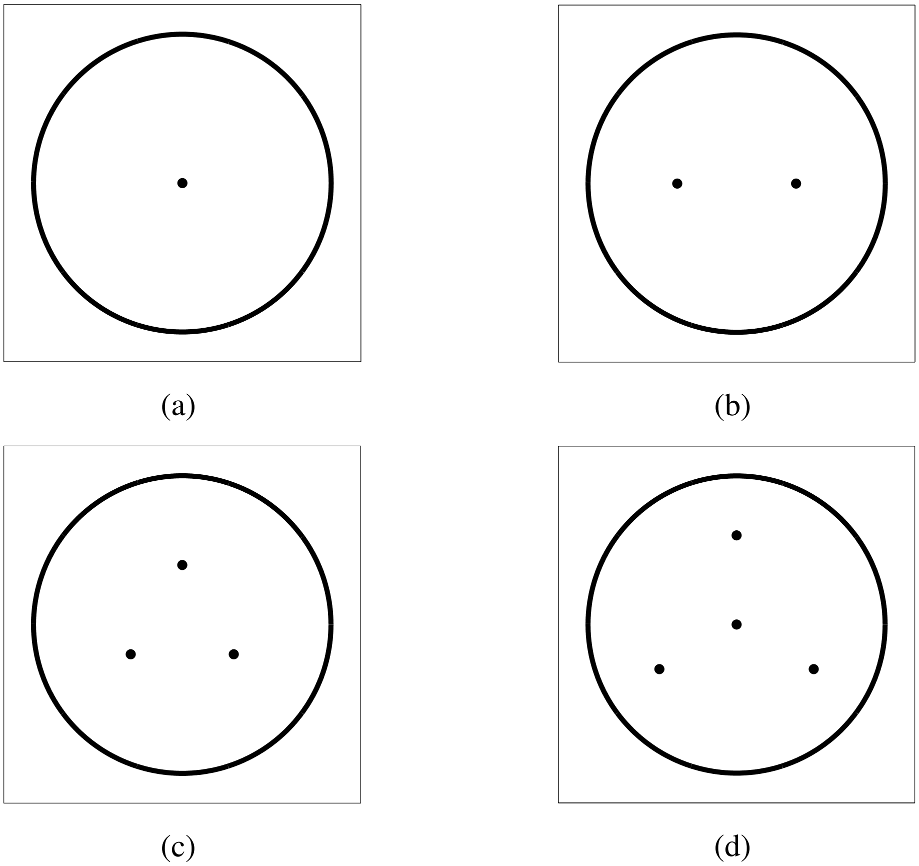


Figure 9.2: Data sink locations for the lifetime optimization using 1 (a), 2 (b), 3 (c), and 4 (d) data sink locations.

9.4.2 Deployment Strategy DS_2: Mobile Data Sink

In this section, we analyze the effectiveness of a mobile data sink for extending network lifetime. Suppose that the mobile data sink stops at a given number N_l of data sink locations, and all of the active sensors report to this sink when it stops at a new location. For small values of N_l such as 2, 3, and 4, we assume that the optimal sink locations form a symmetric pattern, as shown in Figure 9.2. To find the optimal locations, we can use a brute force search, slowly varying the distances between the base stations and the center of the deployment region, while finding the maximum lifetime achievable for each set of sink locations. For values of N_l larger than four, it is more difficult to determine the optimal base station locations. Therefore, we resort to random location deployment.

During the period that the data sink is at each of the locations, the data flow at each sensor should be balanced. To account for this, several modifications must be made to our model's constraints. We will refer to the time during which each data sink location l is operational as L_l. The amount of traffic sent from sensor i to sensor j during the time when sink location l is active will be denoted as t_{ijl}. The conservation of flow constraints become

$$\sum_{j=1}^{N_s} t_{ijl} + r_i L_l = \sum_{j=0}^{N_s} t_{ijl} \qquad \forall i \in \{1, \cdots, N_s\}, \;\; \forall l \in \{1, \cdots, N_l\} \tag{9.5}$$

Meanwhile, the energy consumption of each sensor should be defined as

$$e_i = \sum_{j=0}^{N_s} \sum_{l=1}^{N_l} \left(E_{elec} + \epsilon_{amp} (d_{ij}^t)^\alpha \right) t_{ijl} + \sum_{j=1}^{N_s} \sum_{l=1}^{N_l} E_{elec} t_{jil} \tag{9.6}$$

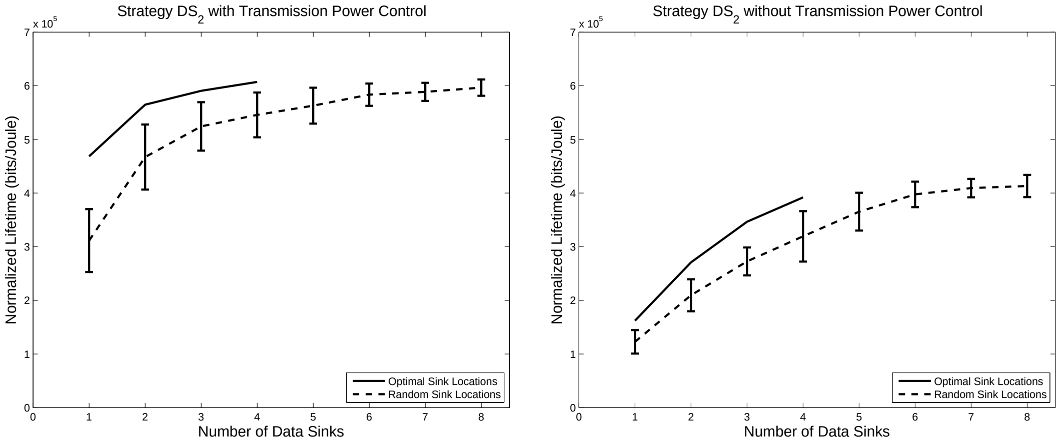
The goal of the linear program is now to maximize $\sum_{l=1}^{N_l} L_l$.

Note that sensors are required to send their traffic in a timely manner. While we do not consider packet delay in our analysis, we make a fundamental underlying assumption that the data must reach its destination before the data sink moves to a new location. Otherwise, a node could simply hold the data until the base station moves to a nearby location and lifetime could be made arbitrarily high.

Figures 9.3(a) and 9.3(b) show plots of the normalized lifetime $\widetilde{L}$ as a function of the number of data sink locations N_l with and without transmission power control, respectively. Plots of $\widetilde{L}$ using optimal data sink locations are given by the solid lines, and plots of $\widetilde{L}$ using randomly chosen data sink locations are given by the dashed lines with standard deviation bars.

The use of DS_2 with random data sink deployment and transmission power control improves network lifetime by 92% to 5.97×10^5 when using eight data sink locations instead of just one. However, the normalized lifetime flattens out at about 8 data sink locations since the hot spot problem is already solved effectively at this point. When transmission power control is not available, the use of 8 data sink locations improves lifetime 237% to 4.13×10^5. However, the improvement again flattens out at 8 data sink locations.

Although the normalized lifetime of DS_2 for a large number of sink locations is higher than that of DS_1 (and thus, the required number of sensors is lower), a mobile



(a)

(b)

Figure 9.3: Normalized lifetime vs. number of data sinks deployed for the sample scenario with transmission power control (a) and without transmission power control (b). Increasing the number of sink locations improves lifetime until a certain threshold is met and the hot spot problem has been effectively solved.

data sink may be much more expensive than a stationary data sink used in DS_1. This cost may affect the overall desirability when we compare and evaluate the different deployment strategies.

9.4.3 Deployment Strategy DS_3: Multiple Data Sinks/Clustering

In a clustering approach, multiple aggregator-capable nodes are deployed and each sink collects data from only a portion of the sensor network for the entire network lifetime. Previous work in this area deals primarily with homogeneous networks, in which any of the deployed nodes is capable of acting as cluster head. While this may be the case in some network scenarios, it can be expected that data aggregation will require more powerful processors and thus, more expensive sensor nodes, resulting in the deployment of heterogeneous networks. In this section, we consider such heterogeneous networks, where cluster heads are actually data sinks that are more capable (e.g., they contain larger batteries, more processing power and memory, and possibly a second radio to link back to a central base station) and significantly more expensive than ordinary microsensors. In our model, a sensor may send its traffic to whichever cluster head it chooses. The chosen cluster head is typically, but not necessarily, the closest cluster head.

This deployment strategy also requires a modification of the first constraint in our network lifetime model. Since we have multiple data sinks, we can no longer refer to a single sink s_0. Rather, we will refer to the data sinks as $S^* = \{s_{N_s+1}, \ldots, s_{N_s+N_l}\}$. Equation 8.1 should be modified as follows.

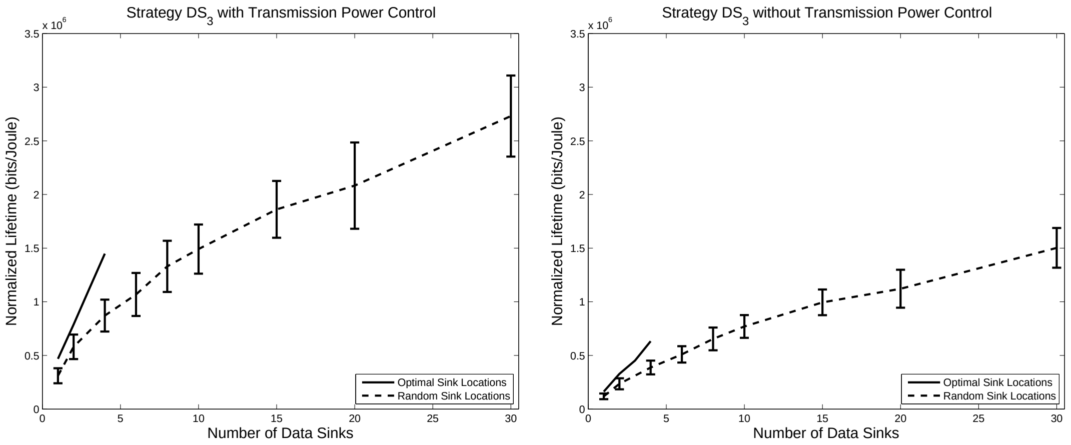
$$\sum_{j=1}^{N_s} t_{ji} + r_i L = \sum_{j=1}^{N_s+N_l} t_{ij} \qquad \forall i \in \{1, \cdots, N_s\} \tag{9.7}$$

The energy consumption of each sensor can be described as

$$e_i = \sum_{j=1}^{N_s+N_l} \left(E_{elec} + \epsilon_{amp} (d_{ij}^t)^\alpha \right) t_{ij} + \sum_{j=1}^{N_s} E_{elec} t_{ji} \tag{9.8}$$

Using the sample scenario, we find the relationship between the normalized lifetime and the number of data sinks that are deployed, as shown in Figures 9.4(a) and 9.4(b) for schemes with and without transmission power control, respectively. The normalized lifetime is given for optimal cluster head placement as well as random placement. Again, optimal cluster head location patterns are only achievable for a small number of cluster heads through brute force searching. As expected, when more cluster heads are deployed, the hot spot problem is reduced and the network lifetime improves. In the most extreme case, so many data sinks are deployed that every sensor can find a data sink just one hop away. The hot spot problem is completely solved in this case. When transmission power control is used, the normalized lifetime is found to be 3.11×10^5 bits/J for a single base station and increases to 2.73×10^6 bits/J for 30 base stations. When transmission power control is not applied, increasing the number of randomly deployed data sinks from 1 to 30 increases the normalized lifetime from 1.19×10^5 bits/J to 1.50×10^6 bits/J.

Note that the performance of DS_3 is better than that of DS_2 since on average, traffic is forwarded over much shorter distances (to the closest data sink rather than the single global data sink). Another potential advantage of clustering is that it may better accommodate certain scheduling schemes. The cluster heads can serve as local controllers for scheduling, which brings additional advantages over the single sink uniform deployment strategy. However, unlike the assumed fixed extra cost for a mobile data sink,



(a) (b)

Figure 9.4: Normalized lifetime vs. number of cluster heads deployed with transmission power control (a) and without transmission power control (b). Large gains in network lifetime can be achieved when even a few extra cluster heads are deployed, especially when their locations are optimized. Random sink locations can provide lifetime improvement, but it is not as large as that obtained using the optimal sink locations. When power control is unavailable, the gap between random sink locations and optimal sink locations is greatly reduced.

the extra cost of this strategy is more likely to have a linear relationship with the number of data sinks deployed. Therefore, a proper number of data sinks must be chosen according to the cost ratio of data sinks and normal sensors.

9.4.4 Deployment Strategy DS_4: Non-uniform Energy Assignment

In strategy DS_4, we loosen the initial energy constraint and allow each sensor to be deployed with a different value of initial energy. In this strategy, Equations 8.1, 8.3, 8.4, 8.6, and 8.7 are used as the constraints for the linear program. The lifetime performance of non-uniform energy assignment for various network radii is shown in Figure 9.5. Compared to the results shown in Figure 9.1, in which optimal traffic distribution is the only option, normalized lifetime improves from 4.69×10^5 bits/J to 9.09×10^5 bits/J for the sample scenario when using transmission power control. The normalized lifetime increases from 1.62×10^5 bit/J to 7.25×10^5 bit/J when transmission power control is not used. Figures 9.6(a) and 9.6(b) show the optimal energy assignment map for nodes at different distances to the base station along with an interpolated polynomial function, for schemes using transmission power control and not using transmission power control, respectively.

Intelligent energy assignment seems to be a good choice for sensor network deployment. However, this strategy is inherently difficult since energy must be assigned differently for individual nodes at different locations. When sensor deployment is performed in a random manner, this becomes almost impossible. However, deployment strategy DS_5 (non-uniform sensor placement) is very similar in nature to DS_4 and its use seems more realistic. We will omit further discussion of DS_4 and focus on strategy DS_5 in the remainder of this chapter.

9.4.5 Deployment Strategy DS_5: Non-uniform Relay/Sensors

If we assume that all sensors must be deployed with equal initial energy, we may deploy more relay/sensor nodes according to the energy maps shown in Figure 9.6, achieving the same goal of providing more energy at particular points in the network. The normalized lifetime obtained using this approach is equivalent to that calculated for DS_4.

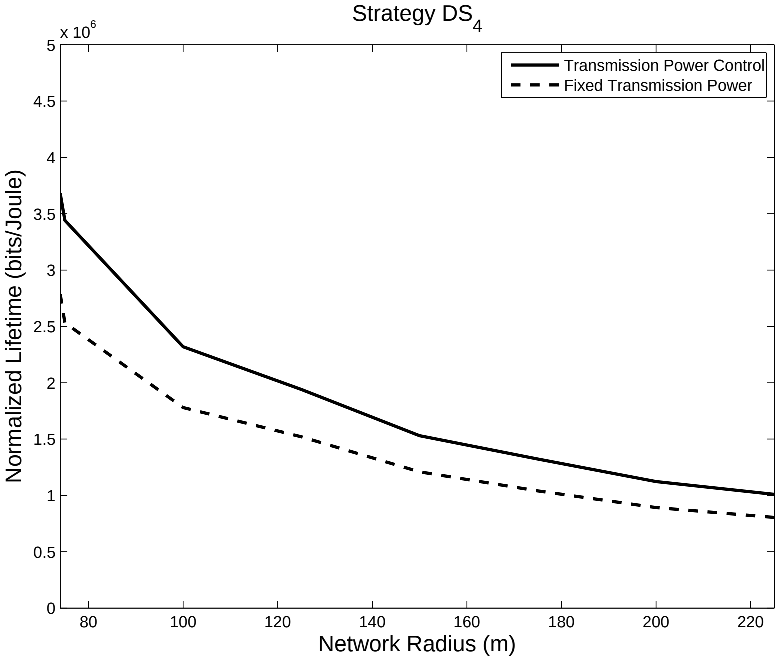
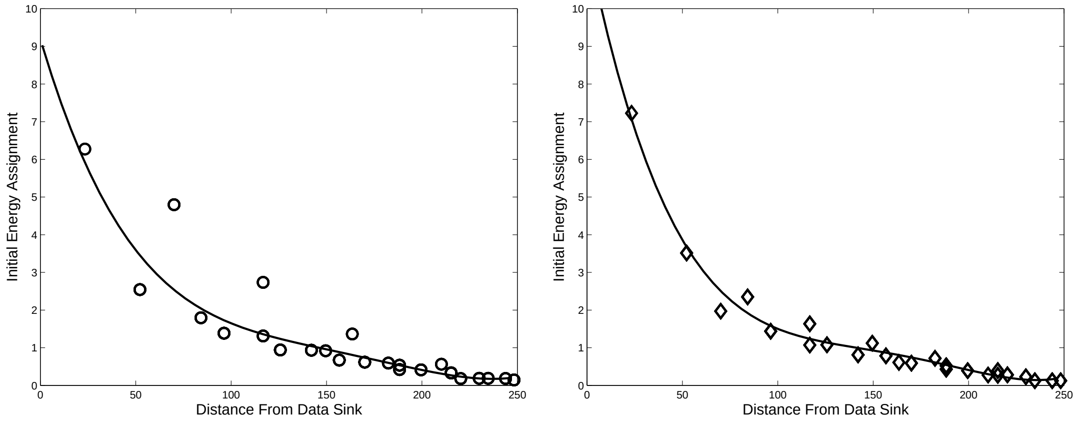


Figure 9.5: Normalized lifetime as network radius varies for the non-uniform energy assignment strategy.



(a) (b)

Figure 9.6: Energy distribution map for the sample scenario with transmission power control (a) and without transmission power control (b). Nodes closest to the base station should be assigned the most energy. The assignment can be approximated using a polynomial function .

9.4.6 Deployment Strategy DS_6: Non-uniform Traffic Generation

In certain sensor networks, more traffic may be generated at distances farther from the base station. For example, in sensor networks designed for intruder detection, the sensors in the network periphery may provide the most important data, as these sensors notify the application when an intruder has entered the network area. The majority of the work for nodes closest to the base station is to forward the traffic, rather than to generate it. In this type of traffic generation pattern, the hot spot problem is automatically alleviated. Consider an extreme case in the one-dimensional scenario in Figure 8.1. If only sensor N_s, the sensor furthest from the data sink, generates traffic, the traffic will be forwarded hop by hop to the data sink. Choosing the next hop closest to the optimal transmission range will be the most energy-efficient forwarding method, and the energy imbalance trends seen in other deployment strategies will not exist.

Data aggregation can be considered a variation of non-uniform traffic generation as well. As data are forwarded to the base station, sensors may perform some processing and aggregate their data with the received data before forwarding. Even if the data generation rate is uniform within the network, data aggregation actually transforms it into a non-uniform traffic generation pattern. Again, this helps to reduce the hot spot problem.

However, in sensor networks where areas closer to the data sink are more of interest for monitoring, more traffic is generated around the data sink. This actually aggravates the hot spot problem. All of the strategies mentioned earlier can be applied to alleviate the problem, and our model is still applicable to these scenarios. However, these scenarios are essentially different from the previous scenarios. Therefore, we will not compare the performance of this strategy with that of the previous strategies.

9.4.7 Cost Comparison of Deployment Strategies

Now that we have analyzed the normalized lifetime of the different deployment strategies, in this section, we analyze the cost-efficiency of the deployment strategies for a chosen target scenario. In our target scenario, we wish to monitor a disc with a radius of 250 m with sensors that send traffic at an average rate of $\bar{r} = 100$ bits/s. The sensors are activated with a density of 0.001 sensors/m^2 and are deployed with an initial energy of $e^{init} = 1000$ J. The target network lifetime L is one year. Network parameters for

Table 9.3: Network parameters for the target scenario.

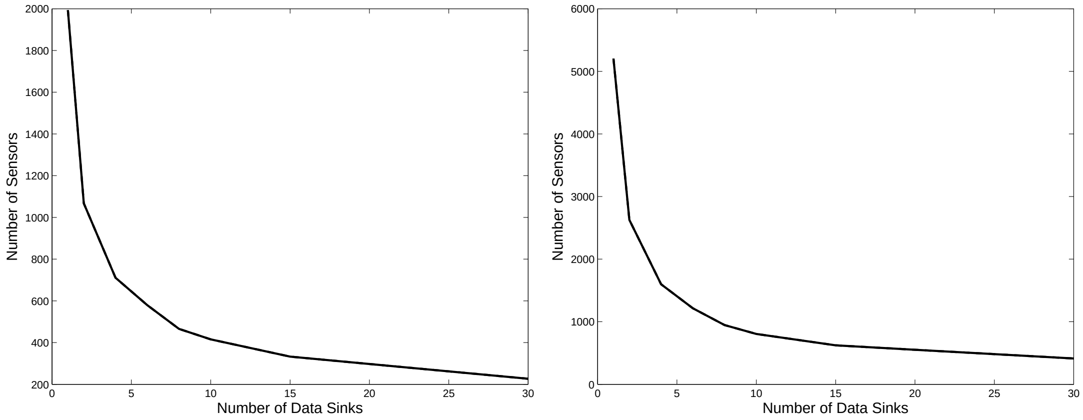
Parameter	Value
Network Radius	250 m
λ_a	0.001 $/m^2$
$\bar{e^{init}}$	1000 J
$\bar{r}$	100 bit/sec
L	1 year

the target scenario are summarized in Table 9.3.

We compare the cost-efficiency of several deployment strategies under two cost scenarios. In both scenarios, the cost of a normal microsensor is assumed to be $75 per unit [96]. In the first cost scenario, the base stations are relatively cheap units ($500 per unit), such as simple Crossbow Stargate nodes [97]. In the second cost scenario, the base station becomes a much more expensive unit ($5000 per unit), such as a high power laptop or custom-designed base station.

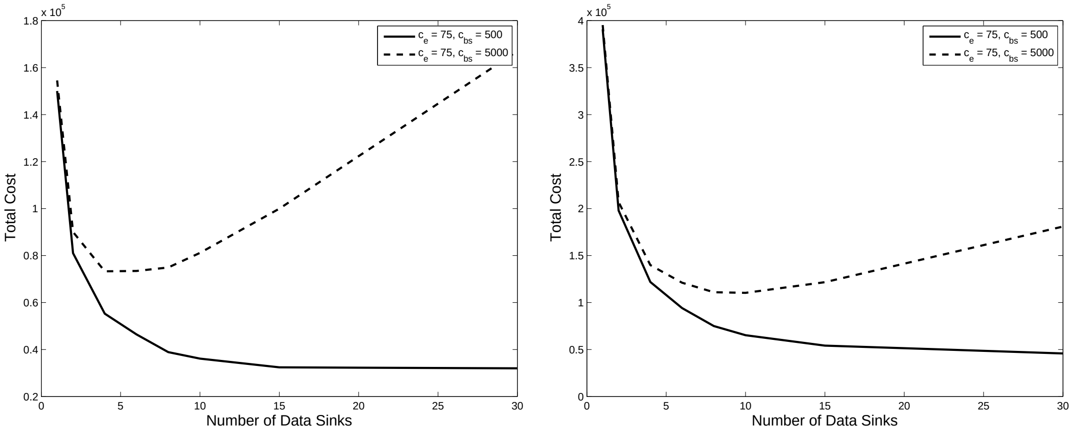
When analyzing the cost-efficiency of DS_2, we assume a large number of movements by the data sink and use the normalized lifetime obtained from using 8 data sink locations, as it seems to be fairly close to an asymptotic bound. For strategy DS_3, before comparing the strategy as a whole, we must find the optimal number of sink locations to use for a given cost model. The number of sensors required for our target scenario when using strategy DS_3 is plotted in Figures 9.7(a) and 9.7(b) for schemes with and without transmission power control, respectively. The total costs for both cost scenarios are plotted in Figures 9.8(a) and 9.8(b). As expected, when data sinks are cheaper, it is more cost-efficient to deploy more of them. As a representative set of solutions, we will consider the use of 8 data sinks and 30 data sinks in our cost analysis.

The number of sensors required to meet the target lifetime for different deployment strategies when using transmission power control and not using transmission power control is summarized in Tables 9.4 and 9.5, respectively. We can see that in Cost Scenario 1, when the base station is relatively cheap, it is wise to use a clustering approach with many base stations, and DS_3 with $N_l = 30$ becomes the most cost-efficient approach. When sensors become much cheaper than base stations, it becomes



(a) (b)

Figure 9.7: Number of sensors required to meet the target scenario lifetime when using transmission power control (a) and when not using transmission power control (b) for the multiple data sinks deployment strategy.



(a) (b)

Figure 9.8: Total cost of deployment for target scenario when using transmission power control (a) and when not using transmission power control (b) for the multiple data sinks deployment strategy.

Table 9.4: Cost evaluation for the two cost scenarios with transmission power control.

	DS_1	$DS_2, N_l = 8$	$DS_3, N_l = 8$	$DS_3, N_l = 30$	DS_5
$N_s(DS_i)$	1322	1038	466	227	681
Cost Scenario 1					
c_s	\$75	\$75	\$75	\$75	\$75
c_{e1}	\$500	\$500	\$500$\times$ 8	\$500$\times$ 30	\$500
C_1	\$99,626	\$78,353	\$38,923	**\$32,006**	\$51,607
Cost Scenario 2					
c_s	\$75	\$75	\$75	\$75	\$75
c_{e1}	\$5,000	\$5,000	\$5,000$\times$ 8	\$5,000$\times$ 30	\$5,000
C_2	\$104,130	\$82,850	\$74,920	\$167,010	**\$56,110**

Table 9.5: Cost evaluation for the two cost scenarios without transmission power control.

	DS_1	$DS_2, N_l = 8$	$DS_3, N_l = 8$	$DS_3, N_l = 30$	DS_5
$N_s(DS_i)$	3820	1499	947	412	854
Cost Scenario 1					
c_s	\$75	\$75	\$75	\$75	\$75
c_{e1}	\$500	\$500	\$500$\times$ 8	\$500$\times$ 30	\$500
C_1	\$286,990	\$112,930	\$75,030	**\$45,900**	\$64,540
Cost Scenario 2					
c_s	\$75	\$75	\$75	\$75	\$75
c_{e1}	\$5,000	\$5,000	\$5,000$\times$ 8	\$5,000$\times$ 30	\$5,000
C_2	\$291,490	\$117,430	\$111,030	\$180,900	**\$69,040**

more effective to use a single base station and deploy more sensors, and thus DS_5 becomes the most cost-effective. For such a scenario, non-uniform sensor deployment is the best option. If this is not possible, then a clustering approach with fewer data sinks is the next best option. After this, if the difference in costs between DS_1 and DS_2 is enough to make up for the hidden costs of DS_2 not shown here (those that are difficult to quantify in a general sense, such as the cost to manually move the data sink or the extra cost of adding robotics to the data sink), then DS_2 should be used. As a last resort, DS_1 can be used.

While the normalized lifetime for some strategies is higher than others, when considering the extra cost of these strategies, they become less desirable than some of the less energy-efficient strategies. A complete evaluation of different strategies should be performed from both an energy and a cost perspective. Although these conclusions sound straightforward, our method provides a quantification on the overall cost, and thus a clear method for making a decision between several potential strategies.

9.5 Summary

In this chapter, we presented a general deployment cost model to evaluate multiple senor network deployment strategies. In our work, we have made the following observations.

1. A good sensor network deployment strategy is one that achieves both energy balance and energy-efficiency.
2. A good strategy should allow sensors to send most of their traffic at the general optimal transmission range.
3. The strategy of mobile data sink deployment has some limitations on lifetime improvement, while the strategy of deploying multiple data sinks can continue to improve network lifetime as more sinks are added until the sub-networks become one-hop networks.
4. The strategy of non-uniform energy assignment achieves both energy-efficiency and energy balance simultaneously. However, it is inherently difficult to apply in practice.

5. Although more intelligent strategies may have better lifetime performance, the cost of these strategies must be fully considered because once the quality of service of a network is satisfied, cost becomes the primary concern for a practical sensor deployment plan.

Chapter 10

Conclusions and Future Work

As the use of wireless sensor networks continues to spread and large-scale networks become realizable, a great deal of effort has focused on energy-efficient sensor network planning. This dissertation contributes to this effort and proposes means to optimally assign roles to individual nodes so that they can most efficiently serve application goals.

10.1 Summary of Contributions

In this dissertation, we have made the following research contributions

- We have developed a sensor network middleware named MiLAN that allows a system designer to optimize sensor selection and network parameters. MiLAN provides mechanisms that allow system QoS requirements and sensor relationships to be specified by a network programmer. Preliminary versions of this middleware have been developed for the BlueZ Bluetooth protocol stack as well as the TinyOS platform.
- We have formalized the problem of joint sensor selection and route selection and shown how optimal network lifetime can be achieved through a linear programming approach.
- We have presented several distributed protocols for role assignment in sensor networks. A blue-noise spatial sampling protocol was presented for sensor selection in a network whose goal is to reconstruct the signal of a bandlimited

phenomenon by a field of sensors. Variations on this protocol allow the end user to collect high-quality data for extended periods of time. We have also presented DAPR, a distributed protocol for joint sensor and route selection that can be used in sensor networks that require the entirety of a region to be monitored by sensors with a nominal coverage range. DAPR is the first known protocol that uses an "application-aware" routing cost, helping to avoid use of sensors that are most important to the application's end goals. We have also shown how existing multicast protocols such as MIP can be modified to use these "application-aware" costs to extend network lifetime.

- We have formalized the transmission range distribution optimization problem and analyzed how transmission ranges should be optimally set in sensor networks characterized by many-to-one traffic patterns. Motivated by the limited improvements that optimizations can achieve over simpler policies, we have presented a general framework for analyzing the cost-efficiency of several deployment strategies that can be used as alternatives to a single-sink network.

10.2 Future Work

The work in this dissertation has opened up many new avenues for future exploration of role assignment in wireless sensor networks. Possible future work in area includes the following.

- The MiLAN middleware should be further developed and tested on the TinyOS platform. Also, for a given hardware platform (i.e., the TelosB motes on which the current implementation exists), a list of tunable network parameters should be enumerated and policies for setting these parameters according to network and application conditions should be explored. The further development of MiLAN for TinyOS should be made simpler through use of the X-Lisa architecture [68].
- The application-aware routing costs that were presented in this dissertation pertain only to networks in which a certain level of coverage should be maintained over the lifetime of the network. However, the concept of application-aware route costs is not limited to such networks. Similar routing costs can be developed for

networks whose goal is to reconstruct a bandlimited phenomenon such as those considered in Chapter 5. Also, the implications of the use of distributed source coding, where certain nodes reduce traffic rates and are partially rather than completely turned off, should be considered.

- The use of application-aware routing costs should be considered in networks with traffic models other than convergecast and multicast. For example, we have done preliminary work integrating application-aware costs into networks that employ network coding as a means to disseminate data to a number of multicast sinks within the network.
- The use of application-aware costs can be extended to levels of the protocol stack other than routing. For example, in congested networks where significant energy is dissipated in the MAC protocol by nodes contending for access to the channel, backoff timers can be set according to these costs.

Bibliography

[1] C. Kidd et al., “The aware home: A living laboratory for ubiquitous computing research,” in *Proceedings of the Second International Workshop on Cooperative Buildings (CoBuild)*, 1999.

[2] S. Intille, “Designing a home of the future,” *IEEE Pervasive Computing*, vol. 1, pp. 76–82, Apr. 2002.

[3] S. Tilak, N. Abu-Ghazaleh, and W. Heinzelman, “Infrastructure tradeoffs for sensor networks,” in *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[4] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, “Next century challenges: Scalable coordination in sensor networks,” in *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1999.

[5] J. Kahn, R. Katz, and K. Pister, “Mobile networking for smart dust,” in *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1999.

[6] C. Raghavendra, K. Sivalingam, and T. Znati, *Wireless Sensor Networks: An Information Processing Approach.* Morgan Kaufmann Publishers, 2004.

[7] C. Raghavendra, K. Sivalingam, and T. Znati, eds., *Wireless Sensor Networks*. Kluwer Academic Publishers, 2004.

[8] K. Nahrstedt, D. Xu, D. Wichadakul, and B. Li, “Qos-aware middleware for ubiquitous and heterogeneous environments,” *IEEE Communications Magazine*, vol. 39, no. 11, 2001.

[9] P. Bonnet, J. Gehrke, and P. Seshadri, “Querying the physical world,” *IEEE Personal Communications*, vol. 7, pp. 10–15, Oct. 2000.

[10] C.-C. Shen, C. Srisathapornphat, and C. Jaikaeo, “Sensor information networking architecture and applications,” *IEEE Personal Communication*, vol. 8, pp. 52–59, Aug. 2001.

[11] I. Lazaridis, Q. Han, X. Yu, S. Mehrotra, N. Venkatasubramanian, D. V. Kalashnikov, and W. Yang, “QUASAR: Quality-aware sensing architecture,” *SIGMOD Record*, vol. 33, pp. 26–31, Mar. 2004.

[12] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, “TAG: a tiny aggregation service for ad-hoc sensor networks,” in *Proceedings of the ACM Symposium on Operating System Design and Implementation (OSDI)*, 2002.

[13] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, “The design of an acquisitional query processor for sensor networks,” in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2003.

[14] T. Abdelzaher, B. Blum, D. Evans, J. George, S. George, L. Gu, T. He, C. Huang, P. Nagaraddi, S. Son, P. Sorokin, J. Stankovic, and A. Wood, “EnviroTrack: Towards an environmental computing paradigm for distributed sensor networks,” in *Proceedings of the 24th International Conference on Distributed Computing Systems*, 2004.

[15] K. Whitehouse, C. Sharp, E. Brewer, and D. Culler, “Hood: A neighborhood abstraction for sensor networks,” in *Proceedings of ACM International Conference on Mobile Systems, Applications, and Services (MobiSYS)*, 2004.

[16] M. Welsh and G. Mainland, “Programming sensor networks using abstract regions,” in *Proceedings of the First USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI)*, 2004.

[17] J. Gutierrez, M. Naeve, E. Callaway, M. Bourgeois, V. Mitter, and B. Heile, “IEEE 802.15.4: A developing standard for low-power, lowcost wireless personal area networks,” *IEEE Network Magazine*, vol. 15, pp. 12–19, Sep.-Oct. 2001.

[18] G. Lu, B. Krishnamachari, and C. Raghavendra, “An adaptive energy-efficient and low-latency MAC for data gathering in sensor networks,” in *Proceedings of the Fourth International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, 2004.

[19] S. Meguerdichian, F. Koushinfar, M. Potkonjak, and M. Srivastava, “Coverage problems in wireless ad-hoc sensor networks,” in *Proceedings of the Twentieth International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2001.

[20] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang, “PEAS: A robust energy conserving protocol for long-lived sensor networks,” in *Proceedings of the Twenty-Third International Conference on Distributed Computing Systems (ICDCS)*, 2003.

[21] D. Tian and N. Georganas, “A node scheduling scheme for energy conservation in large wireless sensor networks,” *Wireless Communications and Mobile Computing Journal*, vol. 3, pp. 271–290, Mar. 2003.

[22] T. Yan, T. He, and J. Stankovic, “Differentiated surveillance for sensor networks,” in *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[23] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, “Integrated coverage and connectivity configuration in wireless sensor networks,” in *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[24] H. Gupta, S. Das, and Q. Gu, “Connected sensor cover: Self-organization of sensor networks for efficient query execution,” in *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[25] E. Royer and C. Toh, “A review of current routing protocols for ad-hoc mobile wireless networks,” *IEEE Personal Communications*, vol. 6, pp. 46–55, Apr. 1999.

[26] S. Singh, M. Woo, and C. Raghavendra, “Power-aware routing in mobile ad hoc networks,” in *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1998.

[27] J. Chang and L. Tassiulas, “Energy conserving routing in wireless ad hoc networks,” in *Proceedings of the Nineteenth International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2000.

[28] R. Shah and J. Rabaey, “Energy aware routing for low energy ad hoc sensor networks,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2002.

[29] W. Heinzelman, J. Kulik, and H. Balakrishnan, “Adaptive protocols for information dissemination in wireless sensor networks,” in *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1999.

[30] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed diffusion: A scalable and robust communication paradigm for sensor networks,” in *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCom)*, 2000.

[31] D. Braginsky and D. Estrin, “Rumor routing algorithm for sensor networks,” in *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[32] S. Ratnasamy and B. Karp, “GHT: A geographic hash table for data-centric storage,” in *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[33] B. Karp and H. Kung, “GPSR: Greedy perimeter stateless routing for wireless networks,” in *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[34] H. Takagi and L. Kleinrock, “Optimal transmission ranges for randomly distributed packet radio terminals,” *IEEE Transactions on Communications*, vol. 32, pp. 246–257, Mar. 1984.

[35] D. Niculescu and B. Nath, "Trajectory based forwarding and its applications," in *Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2003.

[36] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 660–670, Oct. 2002.

[37] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in *Proceedings of the Twenty-Third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2004.

[38] D. Blough, M. Leoncini, G. Resta, and P. Santi, "The k-neigh protocol for symmetric topology control in ad hoc networks," in *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[39] R. Ramanathan and R. Hain, "Topology control of multihop wireless networks using transmit power adjustment," in *Proceedings of the Nineteenth International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2000.

[40] V. Rodoplu and T. Meng, "Minimum energy mobile wireless networks," in *Proceedings of the IEEE International Conference on Communications (ICC)*, 1998.

[41] Y. Tseng, Y. Chang, and P. Tseng, "Energy-efficient topology control for wireless ad hoc sensor networks," in *Proceedings of the International Computer Symposium*, 2002.

[42] Q. Li and D. Rus, "Sending messages to mobile users in disconnected ad-hoc wireless networks," in *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[43] M. Stemm and R. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices," *IEICE Transactions on Communications*, vol. E80-B, no. 8, pp. 1125–31, 1997.

[44] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[45] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *ACM Wireless Networks*, vol. 8, pp. 481–494, Sept. 2002.

[46] A. Cerpa and D. Estrin, "ASCENT: Adaptive self-configuring sensor network topologies," in *Proceedings of the Twenty-First International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2002.

[47] A. Boukerche, X. Cheng, and J. Linus, "Energy-aware data-centric routing in microsensor networks," in *Proceedings of the Sixth ACM/IEEE International Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM)*, 2003.

[48] C. Diot, W. Dabbous, and J. Crowcroft, "Multipoint communication: A survey of protocols, functions, and mechanisms," *IEEE Journal on Selected Areas in Communications*, vol. 15, pp. 277–290, Apr. 1997.

[49] J. Wieselthier, G. Nguyen, and A. Ephremides, "Energy-efficient broadcast and multicast trees in wireless networks," *Mobile Networks and Applications*, vol. 7, pp. 481–492, Dec. 2002.

[50] J. Wieselthier, G. Nguyen, and A. Ephremides, "Distributed algorithms for energy-efficient broadcasting in ad hoc networks," in *Proceedings of the Military Communications Conference (MILCOM)*, 2002.

[51] J.-P. H. M. Cagalj and C. Enz, "Energy-efficient broadcasting in all-wireless networks," *Wireless Networks*, vol. 11, pp. 177–188, Jan. 2005.

[52] B. Calhoun, D. Daly, N. Verma, D. Finchelstein, D. Wentzloff, A. Wang, S. Cho, and A. Chandrakasan, "Design considerations for ultra-low energy wireless microsensor nodes," *IEEE Transactions on Computers*, vol. 54, pp. 727–740, June 2005.

[53] I. Stojmenovic and X. Lin, “Power-aware localized routing in wireless networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, pp. 1122–1133, Oct. 2001.

[54] C. Efthymiou, S. Nikoletseas, and J. Rolim, “Energy balanced data propagation in wireless sensor networks,” in *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS)*, 2004.

[55] M. Bhardwaj and A. Chandrakasan, “Bounding the lifetime of sensor networks via optimal role assignments,” in *Proceedings of the Twenty-First International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2002.

[56] M. Perillo and W. Heinzelman, “Simple approaches for providing application qos through intelligent sensor management,” *Elsevier Ad Hoc Networks Journal*, vol. 1, no. 2–3, pp. 235–246, 2003.

[57] G. Zussman and A. Segall, “Energy efficient routing in ad hoc disaster recovery networks,” in *Proceedings of the Twent Second International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2003.

[58] R. Shah, S. Roy, S. Jain, and W. Brunette, “Data MULEs: Modeling a three-tier architecture for sparse sensor networks,” in *Proceedings of the First IEEE Workshop on Sensor Network Protocols And Applications (SNPA)*, 2003.

[59] H. Kim, T. Abdelzaher, and W. Kwon, “Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks,” in *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[60] J. Luo and J. Hubaux, “Joint mobility and routing for lifetime elongation in wireless sensor networks,” in *Proceedings of the Twent Fourth International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2005.

[61] M. Kalantari and M. Shayman, “Design optimization of multi-sink sensor networks by analogy to electrostatic theory,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2006.

[62] C. Wan, S. Eisenman, A. Campbell, and J. Crowcroft, “Siphon: overload traffic management using multi-radio virtual sinks in sensor networks,” in *Proceedings of the Third ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2005.

[63] S. C. Ergen and P. Varaiya, “Optimal placement of relay nodes for energy efficiency in sensor networks,” in *Proceedings of the IEEE International Conference on Communications (ICC)*, 2006.

[64] Y. T. Hou, Y. Shi, H. D. Sherali, and S. F. Midkiff, “Prolonging sensor network lifetime with energy provisioning and relay node placement,” in *Proceedings of the IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON)*, 2005.

[65] I. Howitt and J. Wang, “Energy balanced chain in wireless sensor networks,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2004.

[66] J. Haartsen, M. Naghshineh, J. Inouye, O. Joeressen, and W. Allen, “Bluetooth: Vision, goals, and architecture,” *Mobile Computing and Communications Review*, vol. 2, pp. 38–45, Oct. 1998.

[67] J. Conway, C. Coelho, D. da Silva, A. Fernandes, L. Andrade, and H. Carvalho, “Wearable computer as a multi-parametric monitor for physiological signals,” in *Proceedings of the IEEE International Symposium on Bioinformatics and Bioengineering (BIBE)*, 2000.

[68] C. J. Merlin and W. B. Heinzelman, “An information-sharing architecture for wireless sensor networks,” in *Proceedings of the IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications Networks (SECON)*, 2006.

[69] M. Perillo and W. Heinzelman, “Optimal sensor management under energy and reliability constraints,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2003.

[70] P. Berman, G. Calinescu, C. Shah, and A. Zelikovsky, “Power efficient monitoring management in sensor networks,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2004.

[71] N. Garg and J. Könemann, “Faster and simpler algorithms for multicommodity flow and other fractional packing problems,” in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, 1997.

[72] R. Ulichney, “Dithering with Blue Noise,” *Proceedings of the IEEE*, vol. 76, pp. 56–79, Jan. 1988.

[73] T. Mitsa and K. Parker, “Digital Halftoning Technique Using a Blue-Noise Mask,” *Journal of the Optical Society of America*, vol. A, pp. 1920–1929, Nov. 1992.

[74] K. Parker, T. Mitsa, and R. Ulichney, “A New Algorithm for Manipulating the Power Spectrum of Halftone Patterns,” in *Proceedings of SPSE’s 7th International Congress on Non-Impact Printing*, 1991.

[75] S. Lloyd, “Least Square Quantization in PCM,” *IEEE Transactions on Information Theory*, vol. 28, pp. 129–137, 1982.

[76] D. Mitchell, “Spectrally Optimal Sampling for Distribution Ray Tracing,” in *Proceedings of SIGGRAPH Conference Computer Graphics*, 1991.

[77] H. Feichtinger and K. Grochenig, “Theory and Practice of Irregular Sampling,” in *Wavelets: Mathematics and Applications* (J. Benedetto and M. Frazier, eds.), pp. 305–363, CRC Press, 1994.

[78] R. Stasinski and J. Konrad, “Improved POCS-based image reconstruction from irregularly-spaced samples,” in *Proceedings of the XI European Signal Processing Conference*, 2002.

[79] K. Sauer and J. Allebach, “Iterative Reconstruction of Bandlimited Images from Nonuniformly Spaced Samples,” *IEEE Transactions on Circuits and Systems*, vol. CAS-34, pp. 1497–1506, Dec. 1987.

[80] S. Soro and W. Heinzelman, “Cluster head election techniques for coverage preservation in wireless sensor networks,” in *University of Rocheswter Technical Report*, 2007.

[81] M. Perillo and W. Heinzelman, “DAPR: A protocol for wireless sensor networks utilizing an application-based routing cost,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2004.

[82] N. Bulusu, J. Heidemannm, and D. Estrin, “GPS-less low-cost outdoor localization for very small devices,” *IEEE Personal Communications*, vol. 7, pp. 28–34, Oct. 2000.

[83] A. Savvides, C. Han, and M. Srivastava, “Dynamic fine-grained localization in ad-hoc sensor networks,” in *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[84] D. Niculescu and B. Nath, “Ad hoc positioning system (APS),” in *Proceedings of the Global Telecommunications Conference (GLOBECOM)*, 2001.

[85] C. Guo, L. Zhong, and J. Rabaey, “Low power distributed mac for ad hoc sensor radio networks,” in *Proceedings of the Global Telecommunications Conference (GLOBECOM)*, 2005.

[86] K. Arisha, M. Youssef, and M. Younis, “Energy-aware TDMA-based MAC for sensor networks,” in *Proceedings of the IEEE Integrated Management of Power Aware Communications, Computing and Networking (IMPACCT)*, 2002.

[87] M. Sichitiu, “Cross-layer scheduling for power efficiency in wireless sensor networks,” in *Proceedings of the Twenty-Third International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2004.

[88] Y. Chen, E. Sirer, and S. Wicker, “On selection of optimal transmission power for ad hoc networks,” in *Proceedings of the Thirty-Sixth Hawaii International Conference on System Sciences (HICSS-36)*, 2003.

[89] I. Stojmenovic and X. Lin, “Power aware localized routing in wireless networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, pp. 1122–1133, Nov. 2001.

[90] I. Akyildiz and I. Kasimoglu, “Wireless sensor and actor networks: Research challenges,” *Ad Hoc Networks Journal (Elsevier)*, vol. 2, pp. 351–367, Oct. 2004.

[91] F. Ingelrest and D. Simplot-Ryl, “Localized broadcast incremental power protocol for wireless ad hoc networks,” in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC)*, 2005.

[92] D. Pan, P. Mavinjurve, H. Ngo, V. Verma, and A. Chandak, “DMIP3S: Distributed algorithms for power-conserving multicasting in static wireless ad hoc networks,” in *Proceedings of the Workshop on High Performance Switching and Routing (HPSR)*, 2004.

[93] B. Wang and S. Gupta, “S-REMiT: A distributed algorithm for source-based energy efficient multicasting in wireless ad hoc networks,” in *Proceedings of the Global Telecommunications Conference (GLOBECOM)*, 2003.

[94] B. Wang and S. Gupta, “Extending lifetime of multicast trees in wireless ad hoc networks,” *Journal of Information Science and Engineering, Special Issue on Mobile Computing*, vol. 20, pp. 425–447, May 2004.

[95] J. Elson, L. Girod, and D. Estrin, “Fine-grained network time synchronization using reference broadcasts,” in *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[96] Moteiv Corporation, http://www.moteiv.com.

[97] Crossbow Technology, Inc., http://www.xbow.com.

Appendix A

Publications

Book Chapters

[1] M. Perillo and W. Heinzelman, "Wireless sensor network protocols," in *Handbook of Algorithms for Wireless Networking and Mobile Computing* (A. Boukerche, ed.), pp. 813–842, CRC Hall, 2005.

[2] M. Perillo and W. Heinzelman, "Sensor management," in *Wireless Sensor Networks* (C. Raghavendra, K. Sivalingam, and T. Znati, eds.), pp. 351–372, Kluwer Academic Publishers, 2004.

[3] W. Heinzelman, A. Murphy, and M. Perillo, "Data- and event-centric communication," in *Wireless Sensor Networks: A Systems Perspective* (N. Bulusu and S. Jha, eds.), Artech House, 2005.

Journal Articles

[1] M. Perillo and W. Heinzelman, "Simple approaches for providing application qos through intelligent sensor management," *Elsevier Ad Hoc Networks Journal*, vol. 1, no. 2–3, pp. 235–246, 2003.

[2] W. Heinzelman, A. Murphy, H. Carvalho, and M. Perillo, "Middleware to support sensor network applications," *IEEE Network Magazine*, vol. 18, pp. 6–14, Jan. 2004.

[3] Z. Cheng, M. Perillo, and W. Heinzelman, “General network lifetime and cost models for evaluating sensor network deployment strategies,” *To appear: IEEE Transactions on Mobile Computing*, 2008.

[4] M. Perillo and W. Heinzelman, “An integrated approach to sensor role selection,” *In Submission*, 2007.

Conference Papers

[1] Z. Cheng, M. Perillo, B. Tavli, W. Heinzelman, S. Tilak, and N. Abu-Ghazaleh, “Protocols for local data delivery in wireless microsensor networks,” in *Proceedings of the IEEE Midwest Symposium on Circuits and Systems (MWSCAS)*, 2002.

[2] M. Perillo and W. Heinzelman, “ASP: An adaptive energy-efficient polling algorithm for bluetooth piconets,” in *Proceedings of the Hawaiian International Conference on System Sciences (HICSS)*, 2003.

[3] M. Perillo and W. Heinzelman, “Optimal sensor management under energy and reliability constraints,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2003.

[4] M. Perillo and W. Heinzelman, “Providing application QoS through intelligent sensor management,” in *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA)*, 2003.

[5] M. Perillo and W. Heinzelman, “DAPR: A protocol for wireless sensor networks utilizing an application-based routing cost,” in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2004.

[6] M. Perillo, Z. Ignjatovic, and W. Heinzelman, “An energy conservation method for wireless sensor networks employing a blue noise spatial sampling technique,” in *Proceedings of the Third International Symposium on Information Processing in Sensor Networks (IPSN)*, 2004.

[7] M. Perillo, Z. Cheng, and W. Heinzelman, “On the problem of unbalanced load distribution in wireless sensor networks,” in *Proceedings of the Global Telecommunications Conference (GLOBECOM) Workshop on Wireless Ad Hoc and Sensor Networks*, 2004.

[8] M. Perillo, Z. Cheng, and W. Heinzelman, “An analysis of strategies for mitigating the sensor network hot spot problem,” in *Proceedings of the International Conference on Mobile and Ubiquitous Systems (Mobiquitous)*, 2005.

[9] M. Perillo, Z. Cheng, and W. Heinzelman, “Strategies for mitigating the sensor network hot spot problem,” in *Proceedings of CollaborateCom*, 2005.