Fundamental Service Support for Wireless Sensor Networks

by

Chong Liu

B.Sc, Wuhan University, 2000
M.Eng, Wuhan University, 2003

A Dissertation Submitted in Partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Fundamental Service Support for Wireless Sensor Networks

by

Chong Liu

Supervisory Committee

Dr. Kui Wu, Supervisor (Department of Computer Science)

Dr. Valerie King, Department Member (Department of Computer Science)

Dr. Sudhakar Ganti, Department Member (Department of Computer Science)

Dr. Hong-Chuan Yang, Outside Member (Department of Electrical and Computer Engineering)

Dr. Vincent Wong, External Examiner (Department of Electrical and Computer Engineering, University of British Columbia)

Supervisory Committee:

Dr. Kui Wu, Supervisor (Department of Computer Science)

Dr. Valerie King, Department Member (Department of Computer Science)

Dr. Sudhakar Ganti, Department Member (Department of Computer Science)

Dr. Hong-Chuan Yang, Outside Member (Department of Electrical and Computer Engineering)

Dr. Vincent Wong, External Examiner (Department of Electrical and Computer Engineering, University of British Columbia)

ABSTRACT

Wireless Sensor Networks (WSNs) are infrastructureless wireless networks that combine sensing, embedded computing, and wireless networking technologies together. Despite the diversity of WSN applications, some functionalities are so fundamental that they are required by most WSN applications. We call them *the fundamental services* as we believe these services are most important and indispensable for a wide variety of WSNs applications. In this thesis work, we investigate solutions to three fundamental services, namely, localization service, joint scheduling service, and adaptive sampling service. Simulation results and experimental results on real testbed demonstrate the effectiveness and efficiency of these solutions. Our solutions to these fundamental services can serve as the basic building blocks in most applications with WSNs.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

- WSNs: Wireless Sensor Networks
- SNR: Signal-to-Noise Ratio
- ROCRSSI: Ring-Overlapping based on Comparison of Received Signal Strength Indicator
- APIT: Approximate Point In Triangle
- EEDC: Energy Efficient Data Collection
- TDoA: Time Difference of Arrival
- GAF: Geographic Adaptive Fidelity
- ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies
- STEM: Sparse Topology and Energy Management
- CCP: Coverage Configuration Protocol
- DOI: Degree Of Irregularity
- GPS: Global Positioning System
- STDMap: SpatioTemporal Data Map
- ARIMA: Auto Regressive Integrated Moving Average
- PLAMLiS: Piecewise Linear Approximation with Minimum number of Line Segments
- MEMS: Micro-Electro-Mechanical Systems

Acknowledgement

First of all, I would like to thank my thesis advisor, Dr. Wu. He has been a wonderful advisor, providing me with support and encouragement. His breadth of knowledge and his enthusiasm and devotion to research always inspire me. I thank him for the countless hours he has spent with me, discussing technical problems and improving my technical writing. His guidance during my time at University of Victoria has been invaluable to me.

I would also like to thank Dr. Valerie King, Dr. Sudhakar Ganti, and Dr. Hong-Chuan Yang. Their suggestions and feedback on my Ph.D. proposal trigger my thinking and give me inspiration to improve this thesis work further.

Special thanks to Dr. Dan Hoffman and Tereus Scott for improving the presentation of localization service.

Finally, I would like to thank my family: my wife Fang Huang, my grandma Wenbin Cao, my parents Jianxun Liu and Jiayu Chen, my parents-in-law Nanqin Huang and Jin Hu, my sister Qin Liu and my brother-in-law Xifang Zou. I am lucky to have such a wonderful family. They have stood by me in everything I have done, providing constant support and encouragement and filling my life with laughter and love. I know I cannot achieve anything without them by my side.

Chapter 1

Introduction

1.1 Wireless Sensor Networks Overview

Wireless Sensor Networks (WSNs) form a new information-gathering paradigm, which combine sensing, embedded computing, and wireless networking technologies together. A typical WSN consists of a large number of sensor nodes with each node equipped with four major components:

- microprocessor and small amount of memory for signal processing and task scheduling;
- sensing devices, such as acoustic sensor, temperature sensor, seismic sensor, magnetic sensor, and even video or still camera;
- wireless transceiver for data delivery;
- battery for energy supply.

Usually, once deployed, through short-range wireless communication, sensor nodes form a multi-hop wireless network to coordinate their behavior, collect and report measurement data, and process the data in a distributed fashion [3]. Sensor nodes play the role of data generator by sampling the environment with installed sensors and injecting the samples into the networks. In addition, sensor nodes play the role of router by routing the data to its ultimate destination via multi-hop radio transmission. Sensor nodes may perform in-network processing, such as data aggregation [43, 38, 63, 65], or serve as a cluster head

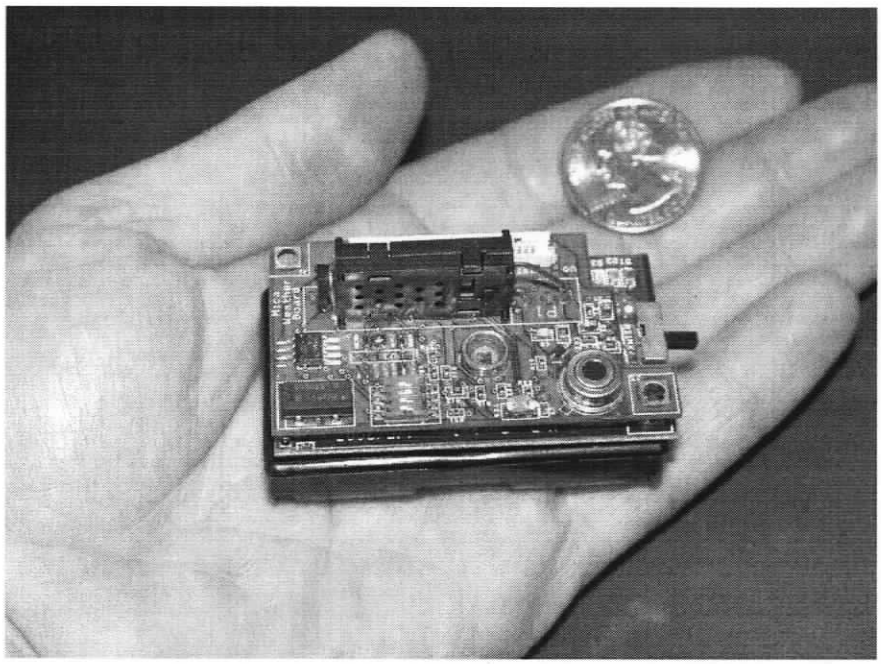

Figure 1.1. *A Mica2 Sensor Node from Crossbow Inc.*

to coordinate the behavior of other sensor nodes within the cluster [24].

The power of WSNs lies in their capability of integrating sensing, embedded computing and wireless networking into a small, low-cost, and low power device. As illustrated in Figure. 1.1, a state-of-the-art sensor node, Mica2 from Crossbow Inc. [1], is as small as a match box. Prior to WSNs, there are successful applications with distributed sensing. For instance, in the United States, temperature and humidity information is provided by over 600 weather observation stations installed near airports, which form a wired sensor network using sensors to collect local environmental observations. However, this network relies on a wired network and power supply infrastructure, hence the cost associated with each observation station is quite high. The high overhead cost makes spatially dense observation impractical.

In contrast, the low-cost self-organized WSNs can be used easily to observe the environment closely in range, densely in space, and frequently in time with reasonable energy

consumption. The feature of self organization [10] introduces an infrastructureless network and significantly reduces the involvement of human labor in the network deployment and network configuration process. Hence, low cost network deployment (for instance, randomly dropping the sensor nodes from the airplane), and network deployment in a hostile environment are possible for WSNs. Dense networks of communicating sensor nodes can significantly increase the Signal-to-Noise Ratio (SNR) of observations by reducing the average distance to the signal source. The highly dense network architecture is inherently more robust against individual sensor node failures due to the redundancy in the network. Low power consumption devices make the sensor node possible to run untethered for a fairly long time without energy replenishment. All these advantages make WSNs powerful enough to reveal previously unobservable phenomena in the physical world [45].

As a result, WSNs, with high temporal-spatial observation resolution, serve as a bridge between the physical world and the digital world and promise to change the way we live. It can be expected that in the near future, tiny and cheap sensor nodes will be deployed anywhere, from roads, walls, cars to even human skins and bodies, to sense and report a wide variety of physical measures such as temperature, light intensity, heart beat rate and blood pressure. The potential applications of WSNs are limitless, ranging from highway traffic monitoring, wildlife habitat monitoring, wildfire detection, to disaster response systems. A typical wireless sensor networks application for wildfire detection, the FireBug system [14], developed and deployed by University of California, Berkeley, is illustrated in Figure. 1.2.

1.2 Characteristics of Wireless Sensor Networks

WSNs are infrastructureless self-organized networks with their unique characteristics, which make it impossible to directly apply existing algorithms and protocols in wired and wireless networks to WSNs as effective solutions. It is worthwhile figuring out these characteristics, which directly determines the design philosophy in WSNs before carrying out our research.

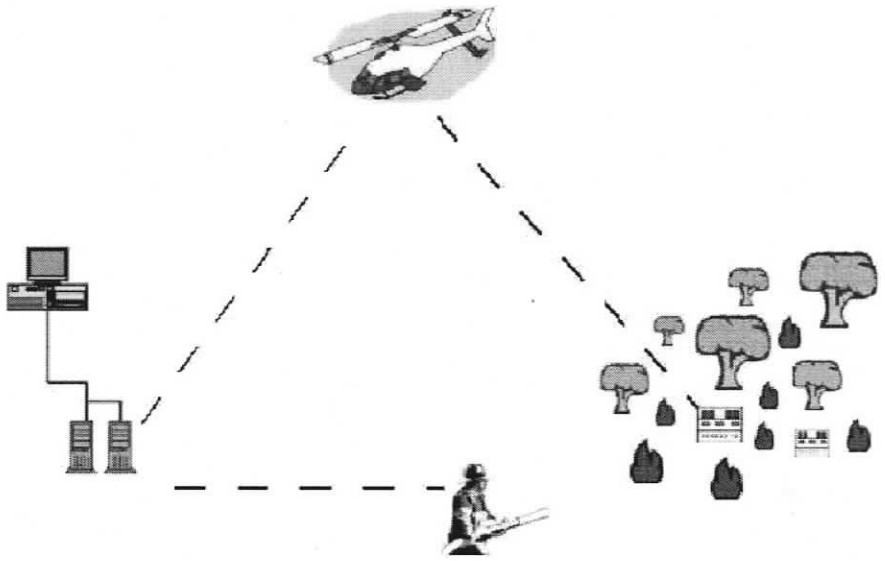

Figure 1.2. *The FireBug: A Wildfire Detection System with Wireless Sensor Networks [14]*

- *Stringent Energy Constraint*

  In WSNs, energy is the most precious resource due to its scarcity. Since there is no infrastructure support, sensor nodes have to run for weeks or months after deployment with human supervision. The battery is usually the only energy source to all the sensing, computing and communication components. Unlike other handheld terminals such as Personal Digital Assistants (PDAs), which can be recharged and maintained by humans, the large scale of WSNs and the possible hostile environment make the energy replenishment very hard. Therefore, the WSN must be very energy efficient and the energy efficiency requirement pervades all the aspects of system design, and derives most of other requirements.

- *Large Scale*

  Several factors necessitate the large scale of networks. First, a high temporal-spatial observation requirement demands a large scale deployment of sensor nodes, especially when each individual sensor node is prone to fail. Second, a large number of sensor nodes are required to prolong the network lifetime since the total energy avail-

able in the whole network is proportional to the number of deployed sensor nodes. Furthermore, a large number of sensor nodes renders enough redundancy of the network, and such redundancy forms the basis for all scheduling algorithms aiming at energy saving. For instance, coverage-preserving scheduling methods are proposed to turn off redundant sensor nodes to save energy while the monitored field is still effectively covered. Due to these reasons, WNSs are expected to consist of hundreds or even thousands of sensor nodes, which is the order of magnitude larger than that in traditional wireless networks.

- *Limited Computation Resource and Communication Capability*

  The large scale of networks makes it very important to decrease the cost of individual sensor nodes, in order to make the cost of whole system acceptable. Besides, the stringent energy consumption precludes the adoption of high performance computing units and high-bandwidth wireless communication, which are usually energy consuming. Therefore, in WSNs, the resources available to individual sensor node are limited. Low processing speed and limited memory make the designs requiring high resource consumption inapplicable in WSNs.

- *Data Centric Naming Scheme*

  The main task of WSNs is to collect environmental data. So data centric processing is an intrinsic characteristic of WSNs. Usually, the ID of individual sensor node is of no interests to applications or users. Besides, due to the sheer number of sensor nodes, it is very hard to maintain a global unique ID for each sensor node. Therefore, in WSNs, the name scheme is data-centric and the sensory data is usually accessed through its content and location. This is essentially different from the Internet, where data is accessed by IP addresses. For instance, an application in WSNs is most likely to request sensory data through queries such as "collect temperature readings larger than 22 Celsius" or "collect temperature readings in Room 257", instead of queries such as "collect temperature readings from sensor nodes with address a, b, and c".

- *High Dynamics*

In WSNs, high dynamics exist in both communication links and network traffic. Sensor node failure is common due to their low-cost components, power depletion, and hostile environment. Besides, high bit error rate, low bandwidth, and asymmetric channels, which are typical characteristics of low cost wireless transceivers, make communication links highly unpredictable. Therefore, the communication links demonstrate high dynamics.

Furthermore, many applications in WSNs are driven by environmental events, such as the approaching of an object in an object tracking application. The occurrence time and the location of interested events are unpredictable. Therefore the responsive data traffic from the sensor nodes to the data collection center (also known as the sink) is unpredictable. Usually the traffic is distributed unevenly in time and space. It is common that the data flow from a hot-spot area to the sink dominates the network traffic and causes network congestion in local network segment near the hot-spot area. Also, the location of hot-spot area varies with the evolving of the observed phenomenon.

- *High Redundancy*

  In WSNs, frequent node failures demand high redundancy to avoid network partition and maintain a good coverage of the monitored area or a satisfactory spatial observation resolution. Unreliable communication links also require high redundancy to achieve robust data delivery by providing multiple routes. Therefore, high redundancy is the essential requirement of WSNs.

  However, high redundancy is a double-edged sword. In WSNs, high data redundancy usually exists in the data generated by neighboring sensor nodes. Transmission of highly redundant data unnecessarily wastes precious energy resource. A better strategy is to remove the data redundancy as early as possible by in-network data aggregation. A more proactive and effective way to save energy is to design distributed sampling schemes to coordinate the data sampling of sensor nodes, preventing the generation of redundant data.

- *High-Level Information Delivery*

  In WSNs, users usually do not explicitly manage the actual information flow. In most cases, they are only interested in high-level information, such as the occurrence of a specific event, and they do not need to know how the data are collected and computed. Hence, in WSNs, the goal is not necessarily to provide the raw data from each sensor node, but rather to provide synthesized high-level information, for instance, the average temperature of all the sensor nodes in a particular room. This necessitates in-network data aggregation in WSNs for energy efficiency.

These distinguished characteristics of WSNs listed above impose a set of *requirements* on the underlying network service, which are significantly different from all existing networks.

- *Energy Efficiency.* It becomes the key issue of whole system and pervades all the aspects of system design.
- *Adaptability.* High dynamics in physical environment, communication links and traffic flow require sensor nodes to be adaptive to frequent changes.
- *Robustness.* The high failure rate of low-cost sensor nodes requires system to be robust and resilient to individual node failure, by exploiting the high node redundancy in the network.
- *Self-Organization.* The sheer number of sensor nodes and, sometimes, the hostile environment require the WSNs to be able to setup effective network infrastructure and configure the networks autonomously, minimizing the human involvement in the deployment and configuration process.
- *Scalability.* The large scale of networks makes scalability an important issue in algorithm and protocol design.
- *Lightweight.* Limited computation resource and communication capability of individual sensor node require lightweight algorithms and protocols and preclude the design abusing system resources.

- *Content-Awareness.* Data-centric naming scheme and in-network data aggregation require intermediate sensor nodes to be content-aware.

1.3 Fundamental Service Support for Wireless Sensor Networks

A sensor node in WSNs is a resource-stringent embedded system. Therefore, the software running on these sensor nodes should be application specific and the application specific design can be expected to achieve the lowest cost and the smallest energy consumption compared to the general purpose design.

Despite the diversity of WSN applications, some services in WSNs are so fundamental that they are required by most WSN applications. We call them *the fundamental services* and we believe these services are very important and indispensable for a wide variety of WSNs applications and can serve as the building blocks for building up WSN applications. We identify and list these fundamental services as follows:

- *Energy-Aware Routing Service.* It takes the energy consumption as a deterministic factor in route selection, and hence delivers data via the most energy-efficient paths.
- *In-Network Data Aggregation Service.* Running on intermediate nodes, it aggregates the data from other sensor nodes. The data redundancy is removed and the total size of data is decreased after the data aggregation, so the energy consumed in data transmission is reduced.
- *Topology Management Service.* It turns off redundant sensor nodes in terms of network connectivity, in order to save energy, while the connectivity of the network is still maintained.
- *Coverage-Preserving Scheduling Service.* It turns off redundant sensor nodes in terms of coverage, in order to save energy, while the whole monitored field is still sufficiently covered.

- *Joint Scheduling Service for Sensing Coverage and Network Connectivity.* Its goal is the combination of that of topology management service and coverage-preserving scheduling service, i.e., schedule sensor nodes to save energy and meet both constraints of sensing coverage and network connectivity simultaneously.
- *Adaptive Sampling Service.* By exploiting local spatiotemporal correlation of observed phenomenon, it coordinates the sampling of distributed sensor nodes to satisfy the observation fidelity requirement with minimum energy consumption. The coordination should also be adaptive to the spatiotemporal correlation changes of observed phenomenon.
- *Localization Service.* It provides location awareness for each individual sensor node.
- *Time Synchronization Service.* It establishes a common time base across sensor nodes.
- *Data Query Service.* It retrieves the required data with minimum energy consumption.

1.4 Contributions

Due to the inherent characteristics of WSNs, it is impossible to directly apply their counterparts in traditional wired and wireless networks to WSNs as effective solutions. Although all these fundamental services may be indispensable parts for particular applications, it is impossible to address the solutions to all of these services in one thesis. Therefore, we focus on investigating the solutions to three fundamental services, namely, localization service, joint scheduling service for sensing coverage and network connectivity, and adaptive sampling service. Our contributions are summarized as follows:

- *Localization Service.* For most applications, a localization scheme, which enables individual sensor node to be aware of its current location, is indispensable, since location is a very important attribute of sensing data. A temperature reading of 21

Celsius is useless if we do not know the location associated with this reading. Besides, location awareness is the prerequisite of geographic-based routing. Therefore, localization service has become an essential requirement for realistic applications over wireless sensor networks. Radio irregularity and stringent constraints on hardware cost and power supply, however, make localization in WSNs very challenging.
In this thesis work, we propose a range-free localization method using Ring-Overlapping based on Comparison of Received Signal Strength Indicator (ROCRSSI). We test its performance with simulation for large-scale networks. Also, our method has been implemplented on broadly-used MICA2 Motes [1] and tested with in-field experiment for a small-scale network by Scott [49]. Compared to one of the best existing range-free localization algorithm, Approximate Point In Triangle (APIT) [23], ROCRSSI exhibits better localization accuracy with lower communication cost and other good features.

- *Joint Coverage-Preserving and Connectivity Maintaining Scheduling Service.* For object tracking or event detection applications, sensing coverage, network connectivity, and energy efficiency are equally important: a large sensing coverage is to meet the users' requirement that an event can be detected with a high probability; network connectivity is to meet the users' requirement that the detected event can be delivered to the sink node; energy efficiency is to meet the users' requirement that the network should keep its operation as long as possible after the deployment. A good scheduling scheme should achieve energy efficiency under the constraints of sensing coverage and network connectivity. However, traditional methods for sensor scheduling use either sensing coverage or network connectivity, but rarely both.
  In this thesis work, we deal with a challenging task: How can we schedule sensor nodes to save energy and meet both constraints of sensing coverage and network connectivity? Our approach utilizes an integrated method that provides statistical sensing coverage and guaranteed network connectivity. We use randomized scheduling for sensing coverage and then turn on extra sensor nodes, if necessary, for network con-

nectivity. Our method is totally distributed, is able to dynamically adjust sensing coverage with guaranteed network connectivity, and is resilient to time asynchrony. We present analytical results to disclose the relationship among node density, scheduling parameters, coverage quality, detection probability, and detection delay. Analytical and simulation results demonstrate the effectiveness of our joint scheduling method.

- *Adaptive Sampling Service*. For continuous environmental data gathering applications, an adaptive sampling scheme is indispensable. By making the temporal and spatial sampling rate of sensor nodes adaptive to the physical phenomenon changing frequency, we can retain high fidelity of data collection with less energy consumption.

  In this thesis work, we propose a data collection method that is based on a careful analysis of the surveillance data reported by the sensors. By exploring the spatial correlation of sensing data, we dynamically partition the sensor nodes into clusters so that the sensors in the same cluster have similar surveillance time series. They can share the workload of data collection in the future since their future readings may still likely be correlated and hence predictable. Furthermore, during a short time period, a sensor may report predictable readings. Such a correlation in the data reported from the same sensor is called temporal correlation, which can be explored to further save energy. We develop a generic framework called Energy Efficient Data Collection (EEDC) to address several important technical challenges, including how the sensor nodes can be partitioned into clusters, how the clusters can be dynamically maintained in response to environmental changes, how the sensor nodes can be scheduled in a cluster, how the temporal correlation can be exploited, and how the data in the sink can be restored with high fidelity. We conduct an extensive empirical study to test our method using both a real test bed system and a large-scale synthetic dataset.

The rest of this thesis is organized as follows. Chapter 2 reviews some research advances on the related topics. Chapter 3 introduces ROCRSSI as the solution to the localization service. Chapter 4 introduces our scheduling approach that can preserve coverage

and maintain network connectivity simultaneously. Chapter 5 introduces our EEDC framework as the solution to the adaptive sampling service. Finally, Chapter 6 summarizes the research work and presents future research directions.

Chapter 2

Related Research

2.1 Related Work on Sensor Localization

Making each sensor node aware of its current location information is essential for many applications over WSNs. First, location information is needed to determine where sensing data originates, e.g., the location of a monitored vehicle or the point of entry of a burglar. Second, location information can assist in system functionalities such as geographical routing [33], location-based sensor scheduling [40], and location-based information querying [21]. Third, location information facilitates application services such as a location directory service that provides doctors with medical equipment and personnel information in a smart hospital. It is clear that with the advance of WSN technologies, many protocols and applications will depend on location-aware devices.

Due to the stringent constraints of energy consumption, cost, and bandwidth on sensor devices, traditional localization approaches are usually not suitable for wireless sensor networks. New localization approaches have been proposed for wireless sensor networks to provide per-node location information. These approaches can be divided into two main categories: range-based methods and range-free methods. The former ones depend on point-to-point distance (range) estimations to reference points to obtain nodes' locations. The latter ones, instead, do not depend on any absolute range information. In both methods, a small number of reference nodes are assumed to know their own locations and serve as anchor nodes.

The general idea of range-based approach is to estimate the absolute point-to-point distance between a sensor node to anchors. Based on these distance estimations, in most cases, triangulation is used to compute the locations of sensor nodes. Some approaches map the received radio signal strength directly to the distance between the sender and the receiver [4, 25] and thus are called RSSI (Received Signal Strength Indicator) methods. Nevertheless, due to the effects of multi-path fading, reflection, diffraction, scattering, background interference, etc., radio signal propagation demonstrates great irregularity [19]. Therefore, it is hard to get accurate distance estimation based purely on RSSI. To improve estimation accuracy, range-based approaches usually take advantage of averaging, smoothing, and other hybrid techniques.

The Time Difference of Arrival (TDOA) technique [46] presents another means to get range estimation with better accuracy than pure RSSI techniques. The main idea of TDOA is to calculate the distance based on two distinct modalities of communication: ultrasound and radio. Ultrasound signal and radio signal travel at vastly different speed. The radio signal is used for synchronization and the ultrasound signal for ranging. The TDOA method works very well but requires extra hardware on sensor nodes.

In contrast to range-based approaches, range-free approaches do not depend on absolute distance estimations. Locations are estimated by utilizing the relationship between sensor nodes and anchor nodes. Such relationship might be connectivity, hop-count information, *relative* distance, etc. Several representative range-free localization methods include the Centroid algorithm [6], mere connectivity based localization [52], the DV-hop approach [42], and the amorphous positioning algorithm [41].

In the Centroid algorithm [6], a sensor node takes the centroid of all its audible anchors as its estimated location. If all the anchors are deployed at the intersection points of a grid, the average location estimation error is about the one-third of the distance between two anchors [6].

In [52], mere connectivity information and a multi-dimensional scaling algorithm are used for location estimation. Multi-dimensional scaling is a type of data analysis technique

which requires $O(n^3)$ calculation complexity for a network of n nodes.

The DV-hop approach [42] obtains the estimation of distance between two nodes by translating the hop count between them. Initially, anchors flood their location information to all the nodes in the networks. Each node (sensor/anchor) maintains a minimum hop count to their audible anchors. Once this phase is finished, an anchor infers the distance to another audible anchor, divides the distance by the hop count to that anchor, and distributes the information of the average distance per hop count to the network via controlled flooding. The information of average distance per hop count is called "correction" information and will be used by nearby sensors to map hop counts to distances. A sensor node uses the latest "correction" information to estimate distances to anchors and finally performs multilateration to three or more anchors to estimate its own location.

The amorphous positioning algorithm [41] is very similar to the DV-hop approach. The main difference between the two methods is on the way of computing average per hop distance. Instead of using simple division to get average per hop distance, the amorphous positioning algorithm calculates average per hop distance using the Kleinrock and Sliverster formula [32].

Unlike the above range-free approaches, APIT [23] and ROCRSSI use a totally different way for location estimation: a sensor decides whether itself fall within a sequence of regular geometric areas (triangles in APIT and rings in ROCRSSI) by comparing received signal strength, and narrows down the estimated area by overlapping all the areas that the sensor node believes itself to reside in. APIT claims the best accuracy among all the existing range-free methods and our simulation results demonstrate that ROCRSSI even outperforms APIT in terms of accuracy and communication cost.

2.2 Related Work on Sensor Scheduling

As discussed in Section 1.2, high redundancy exists both in network topology and data collection. In order to save energy and prolong the lifetime of sensor nodes, sensor scheduling

plays a critical role by exploiting these redundancy and carefully coordinating the working schedule of sensor nodes.

In order to exploit redundancy in network topology, redundant (in terms of network connectivity) sensor nodes can be turned off to save energy without degrading the network connectivity. Those protocols for this purpose are called topology management protocols.

The work in [22] provides the first asymptotic result on the relationship between the power level and the network connectivity. By using percolation theory, it proves that in order to maintain connectivity in a network with randomly placed nodes, the average node degree of the network should be in the order of $(logn + c)$, where n is the total number of nodes and c is a constant. Another similar work could be found in [61].

With Geographic Adaptive Fidelity (GAF) [60], the area is divided into cells with same size by a set of virtual grids. The size of a cell is small enough so that any node can communicate directly with any other nodes within its neighboring cells. Therefore, only one node is required to be active within a virtual cell in order to maintain the network connectivity.

SPAN [8] maintains a routing backbone and allows sensor nodes that do not belong to the backbone to sleep. The backbone nodes are also called "coordinators". A coordinator selection process is triggered periodically to balance the energy consumption among different nodes, since compared to a non-coordinator node, a coordinator consumes more energy to relay data for other nodes.

Adaptive Self-Configuring sEnsor Networks Topologies (ASCENT) [7] is similar to SPAN in the sense that it chooses some sensor nodes to be active as routers while allows others to sleep to save energy. However, unlike SPAN, ASCENT selects an active router depending on not only local connectivity, but also the observed data loss rates. Therefore, ASCENT can obtain a strongly connected network with more reliable transmissions.

Sparse Topology and Energy Management (STEM) [48] puts sensor nodes into sleep state more aggressively. Sensor nodes wake up only when they have data to transfer or they receive requests from their neighbors to forward the data to the sink node. A separated

paging channel is dedicated to the wakeup operation. Therefore, STEM trades latency for further energy saving.

In addition to maintaining network connectivity, maintaining a sufficient sensing coverage is also a critical requirement of sensor networks, since sensing coverage directly determines the monitoring quality provided by sensor networks in a designated region. By turning off redundant (in terms of coverage) sensor nodes, the coverage quality can be maintained and energy efficiency can be achieved at the same time. The existing coverage-preserving scheduling schemes for wireless sensor networks include [27, 54, 62, 64].

In [27], a random scheduling scheme and a coordinated sleep scheduling scheme are proposed. In the random one, time is divided into slots with same length and each node determines randomly and independently at each slot whether it should be on or off. In the coordinated one, each node is assumed to be aware of its current location. Thus it can check whether it is totally redundant or not. If yes, it backs off randomly. When the backoff timer expires, it picks up a sponsor node set and broadcasts a request message to inform the nodes in the sponsor set to stay awake in a predefined sleep period. The sponsor node set consists of its neighbor nodes that can fully cover its sensing range. By adjusting the backoff timer based on the relative residual energy, energy balance can be achieved.

A similar method was proposed in [54]. A coverage-based off-duty eligibility rule and a backoff-based node-scheduling scheme were adopted to guarantee a high sensing coverage.

In [64], a sensor node uses a probing mechanism to determine whether it should sleep. Once a sleeping node wakes up, it broadcasts a probing message to ask for reply from its neighboring active nodes. If no reply is received within a timeout period, the node assumes that there are no working nodes nearby and starts to work till it depletes its energy. Otherwise, it believes that it is redundant and goes to sleep again. The coverage rate can be changed by adjusting the probing range and the wakeup rate.

In [62], the authors divided the whole monitored field into grids and transformed the area coverage problem into the grid intersection point coverage problem. Each sensor node knows its location and its neighbors' locations. By exchanging messages with its

neighbors through an adaptive energy-efficient sensing coverage protocol, each node is able to dynamically decide a schedule for itself, which guarantees the grid intersection points within its sensing range to be monitored by itself or by its neighbors at any time.

Unfortunately, most of the existing work addresses the connectivity problem and the coverage problem in isolation. They only solve one problem without considering the other. However, to operate successfully, a sensor network must ensure network connectivity and coverage quality at the same time. Treating coverage and connectivity in a unified framework for coverage preservation and connectivity maintenance is critical in building up practical sensor network applications.

Recently, in [51], the joint problem of coverage and connectivity is considered in a network with sensor nodes deployed strictly in grids. Each sensor node can probabilistically fail. The sufficient and necessary conditions for connectivity and coverage in this type of networks are provided.

The joint problem in more general sensor networks where the sensor nodes are deployed at random is also investigated. In [66], it is proved that to ensure that a full coverage of a convex area also guarantees the connectivity of the active nodes inside the area, the communication range should be at least twice of the sensing range. Therefore, the joint problem is simplified to maintain a complete coverage of a convex region if the communication range is greater than twice of the sensing range. In [55], the authors enhance the work in [66] by releasing the constraint. They proved that "the communication range is twice of the sensing range" is the sufficient condition and the tight lower bound to ensure that complete coverage implies connectivity, no matter the area is a convex area or not.

In paper [58], the authors draw the same conclusion as in paper [66]. In addition, they present a Coverage Configuration Protocol (CCP) that can provide fully coverage of a convex region. In the case where the communication range is greater than twice of the sensing range, the connectivity of networks is guaranteed by the full coverage, so no mechanism for connectivity maintenance is needed. To deal with the situation where the communication range is less than twice of the sensing range, the authors propose to integrate CCP

with a topology management protocol SPAN [8] to provide both coverage and connectivity. Note that since our method does not assume the availability of accurate locations of sensor nodes, it is hard to provide a fair quantitive comparison between our method and CCP+SPAN. Instead, we compare them qualitatively and summarize their main qualitative differences in Table 2.1.

Table 2.1. *Comparison of Our Method to CCP+SPAN*

Method	Full Coverage	Dynamic Coverage Adjustment	Location Awareness
Our Method	Not Guaranteed	Allowed	Not Needed
CPP+SPAN	Guaranteed	Not Allowed	Needed

In [21], the authors develop the notion of a connected sensor cover, defined as the sensor set that can fully cover the queried area and constitute a connected communication graph at the same time. The authors also demonstrate that the calculation of the smallest connected sensor cover is NP-hard, and they propose both centralized and distributed approximate algorithms to solve it and provide the performance bounds as well. However, unlike our approach, the method in [21] requires each individual sensor node to be aware of its precise location, in order to check its local coverage redundancy.

2.3 Related Work on Adaptive Sampling

In WSNs, adaptive sampling is used to ensure the spatial resolution and/or temporal resolution of collected data meets the requirement with minimum energy consumption. Theoretically, adaptive sampling problem is also a sensor scheduling problem, i.e., coordinating the working schedule of distributed sensor node to ensure data quality. However, the target, i.e., the requirement of data quality, is different from that of the sensor scheduling problem we discussed in Section 2.2.

In Section 2.2, all the sensor scheduling methods aim at the event detection applications in WSNs, i.e., detecting and reporting the occurrence of interested events to the sink node

with minimum delay. The sensor device has a coverage area in which any event occurs will be detected with high confidence. Therefore, high detection rate of interested events requires high coverage rate. Also, the timely delivery of detected event requires the network is connected all the time. The sensor node does not transmit data to the sink unless event detection occurs. In this context, the data quality is directly determined by the coverage rate and network connectivity.

Adaptive sampling aims at continuous data collection applications in WSNs. In short, this type of applications require sensor nodes continuously sampling local environmental measures, such as temperature and humidity, and transmitting them back to the sink node. Each sample of individual sensor node is associated with the sampling location and time. Then the sink node assembles all the received samples, in order to reproduce the evolving process of certain physical phenomenon in the monitored area. In this context, the data quality is directly determined by the spatial and temporal resolution of these samples. In order to save energy, it is desirable that sensor nodes can adaptively adjust their own temporal sampling rate and overall spatial sampling rate, responsive to the changing of physical phenomenon, making the resulted spatiotemporal resolution just above the predefined resolution requirement.

It is apparent that sensor scheduling methods for event detection applications are not applicable here, because they only consider the geographical distance between sensor nodes in sensor scheduling. To achieve adaptive sampling, sensor nodes must aware of, and hence exploit the spatiotemporal correlation among collected data.

Adaptive sampling by exploiting spatiotemporal correlation in WSNs is a brand new topic. Although some research exploits the spatial correlation among sampling data, the correlation is used in the context of in-network data aggregation [43, 38, 63, 65], where the redundant data are partially aggregated in certain aggregation nodes on their way to the sink. Our adaptive sampling approach utilizes the spatial correlation in the context of sensor scheduling. Therefore, data redundancy among sensor nodes can be determined in runtime to avoid the generation and transmission of redundant data. So we can expect our

adaptive sampling approach can achieve much higher energy saving than in-network data aggregation.

Rather than considering coverage as the only factor in scheduling, several pioneering methods have been proposed to adjust spatial sampling rate according to statistic features of the monitored phenomenon. In [18], a linear model was proposed to capture the spatiotemporal correlations among sampling data from different sources. With this model, most sensor nodes can be turned off and their readings can be estimated with certain accuracy by using the linear combinations of data from working sensor nodes. However, in real world, a lot of systems may not be linear. Furthermore, the method of choosing the right working nodes has not been discussed in [18].

A novel approach to adjusting spatial sampling rate with the help of mobile sensor nodes was proposed in [5]. Mobility, combined with an adaptive algorithm, allows the system to get the most efficient spatiotemporal sampling rate to achieve the specified data accuracy. Mobility can also make the system respond quickly to unpredictable environmental changes.

Unlike other energy efficient data collection techniques exploiting spatial correlation such as Slepian-Wolf coding [9, 12] and model-driven data acquisition [13] where the spatial correlation model among distributed nodes is assumed to be known as a priori, our adaptive sampling method does not make any particular assumption on the spatial correlation model. The only assumption made is that the observations of neighboring nodes are to some extent correlated, and this correlation can be captured and exploited by our method in *runtime*.

To exploit temporal correlation among sensor readings, our method adopts piecewise linear approximation technique. Piecewise linear approximation is a hot topic in the database research community and is widely used to classify, cluster, and index times series. The basic idea is to approximate complex time series with pieces of line segments. Many algorithms have been proposed [15, 44, 31, 30, 53, 57]. These algorithms are supposed to be run on a server, which has enormous computational resources for mining huge time se-

ries databases online or offline. Unfortunately, sensor nodes in wireless sensor networks are actually an embedded system, and due to their stringent computation and energy resource, these methods cannot be directly applied. A lightweight algorithm, like the proposed PLAMLiS algorithm that can achieve good balance among accuracy, optimization, and resource consumption, is hence more suitable in wireless sensor networks.

Chapter 3

Localization Service

3.1 Introduction

As discussed in Section 2.1, location information plays a crucial role for many applications over WSNs and localization in WSNs has become an important research direction. Many algorithms for WSNs have been proposed to provide per-node location information. They can be divided into two categories: range-based methods and range-free methods. Range-based methods depend on absolute point-to-point distance (range) estimation or angle estimation, while range-free methods do not require any range information at all.

Range-based localization depends on the assumption that the absolute distance between a sender and a receiver can be estimated by received signal strength or by the time-of-flight of communication signal from the sender to the receiver. The accuracy of such estimation, however, varies with the transmission medium and surrounding environment and usually relies on complex hardware [46]. As the miniaturization of sensor nodes continues, the hardware cost associated with range-based localization will make this method less and less attractive. In contrast, range-free localization does not estimate the absolute point-to-point distance based on received signal strength. As a result, the hardware can be greatly simplified, making range-free localization appealing for WSNs.

In this thesis work, we present a range-free localization algorithm which uses Ring-Overlapping based on Comparison of Received Signal Strength Indicator (ROCRSSI). While many range-free localization approaches have been proposed [23, 41, 42], their

performance has been evaluated only via simulation; we have not seen reports on real implementation of these methods. We test our ROCRSSI localization method based on a small-scale wireless sensor network with MICA2 Mote sensors. We have also performed an extensive simulation study with large-scale sensor networks. Compared to one of the best range-free localization methods, Approximate Point In Triangle (APIT), ROCRSSI is more accurate and has several other good features.

The rest of this chapter is organized as follows. In the next section, we present the ROCRSSI localization method. In Section 3.3, we discuss the impact of radio irregularity on the algorithm. In Section 3.4, we evaluate ROCRSSI experimentally using MICA2 Motes in two network configurations, and in Section 3.5, we evaluate ROCRSSI using simulation and a wide variety of network configurations. In Section 3.6, we compare ROCRSSI to APIT, one of the best existing range-free localization algorithms. Finally, we conclude this chapter in Section 3.7.

3.2 The ROCRSSI Algorithm

To calculate a position for sensor S, the ROCRSSI algorithm generates a set of rings each containing S, computes the intersection of the rings, and uses the center of gravity of the intersection as the estimated position of S.

The algorithm is based on the following assumptions, for S and the nodes within its radio range:

1. There are two kinds of nodes: anchor nodes and sensor nodes. Anchor nodes are special nodes that know their own locations via GPS (Global Positioning System) [26] or other means. The positions of the anchor nodes are known to S. The distance between anchor nodes X and Y is denoted by $d(X, Y)$.
2. The Received Signal Strength Indicator (RSSI) values between each pair of anchor nodes and between each anchor node and S are known to S. The received signal strength measured by node Y for a signal transmitted from node X is denoted by

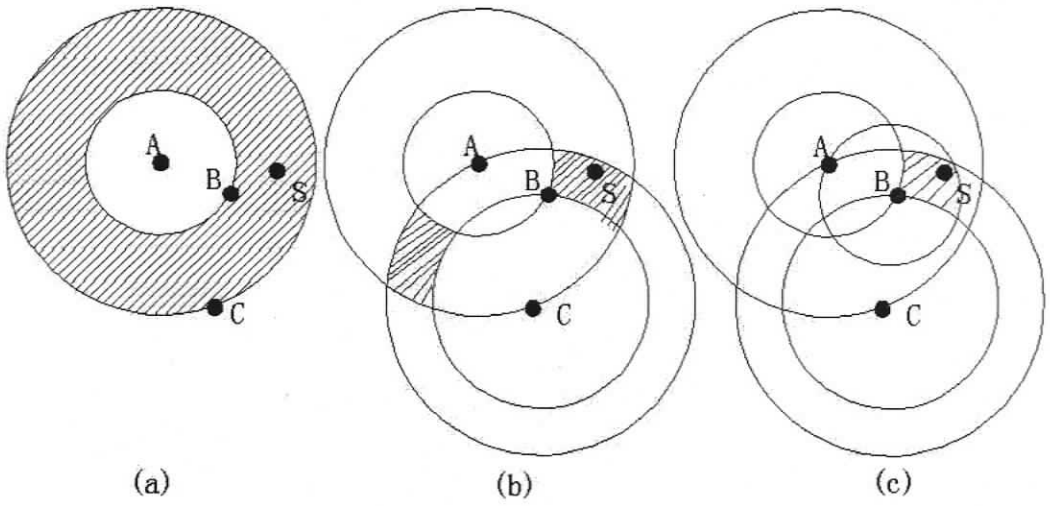

Figure 3.1. *An Example of ROCRSSI: How S Decides Its Location*

$\gamma(X, Y)$.

3. Signal strength decreases monotonically as distance increases, with some variation as noted in Section 3.3.

Consider the scenario shown in Figure 3.1. A, B and C are anchor nodes and S is a sensor node. As shown in Figure 3.1(a), using assumption 1, S can calculate the ring centered at A. The inner radius is $d(A, B)$; the outer radius is $d(A, C)$. Suppose that $\gamma(A, B) > \gamma(A, S)$. Then S can conclude that it lies outside the circle with center A and radius $d(A, B)$. Suppose further that $\gamma(A, S) > \gamma(A, C)$. Then S can conclude that it lies inside the circle with center A and radius $d(A, C)$. Hence, S lies in the shaded ring shown in Figure 3.1(a). In the same way, another ring centered at C can be computed, as shown in Figure 3.1(b). Hence, S lies in the intersection of the two rings. Circles, as well as rings, can be used to reduce the intersection area. For example, as shown in Figure 3.1(c), if $\gamma(B, S) > \gamma(B, A)$, S can conclude that it falls within the circle with center B and radius $d(B, A)$. With more rings or circles, the intersection area of the rings and circles becomes smaller and smaller.

The ROCRSSI algorithm operates in three phases:

1. *RSSI measurement*: each sensor node and anchor node measures RSSI from each anchor node in range.
2. *Anchor node data distribution*: each anchor node transmits its location and the RSSI data from the previous phase.
3. *Sensor node location estimation*: each sensor node uses the data from the previous two phases to determine its position.

In the first phase, each anchor node transmits a fixed number of messages containing the anchor identifier. Both the anchor nodes and the sensor nodes monitor the RSSI value for these messages. After the last message has been received, each node computes the mean RSSI value for each node within range.

In the second phase, each anchor node A transmits a message containing the anchor identifier and location, and a $[\gamma(X, A), \mathit{identifier}]$ pair for each anchor node X within range of A, computed in the previous phase. At this point, each sensor node S has acquired:

- The location and identifier of each anchor node within range.
- A value for $\gamma(X, S)$ for each anchor node X in range, acquired during the first phase.
- For each anchor node A within range of S, a value for $\gamma(B, A)$ for each anchor node B in range of both A and S, acquired during the second phase.

For example, in Figure 3.1(a), suppose that the nodes shown are all within radio range of each other. Among the four nodes, there are 12 ordered pairs. At the end of the second phase, S will have acquired an RSSI value for 9 of the 12 pairs. The three values $\gamma(S, A)$, $\gamma(S, B)$, and $\gamma(S, C)$ will be absent, because *sensor nodes do not transmit in the ROCRSSI algorithm.* Note that, during the second phase, sensor S may receive a message from, e.g., anchor node A which contains information about an anchor node X in range of A but not in range of S. This information will be discarded; S retains only information about anchor nodes in range.

Obtaining enough information, in the third phase, each sensor node uses a grid-scan algorithm [23] to calculate the center of gravity of the intersection area. In this algorithm,

the whole field is divided into a square grid. In general, finer grids result in better accuracy but also larger computational overhead. A counter, initialized to 0, is associated with each square in the grid. Each time when the sensor node generates a ring, it increments the counter value associated with the squares within the ring. Once all the ring regions have been processed, the sensor node scans the grid to find the area with the maximum counter value. It takes this area as the final intersection area and calculates its center of gravity as its location.

3.3 Impact of Radio Irregularity

According to measurement results over real sensor devices, radio propagation is usually not homogenous in all directions [19, 50, 67], i.e., different directions have different radio attenuation rates. Factors such as heterogeneous transmission environment in different directions [67] and antenna position in sensor devices [49] may cause such phenomenon. Figure 3.2 shows the radio propagation pattern of MICA2 Mote sensors measured with an experimental test bed [49].

Therefore, a good localization algorithm should accommodate radio irregularity. The ROCRSSI algorithm is resilient to radio irregularity for the following two reasons.

First, the grid-scan algorithm helps reduce the influence of incorrect rings generated from radio irregularity. The incorrect rings are those the sensor node assumes itself to reside in, but actually it does not, due to radio irregularity. As mentioned before, the grid-scan algorithm takes the grid elements with maximum counter value as the final intersection area. In Figure 3.3, suppose that *more than half* of the rings generated by RSSI comparison are correct, and that the intersection area of all correct rings is the gray area labeled as A. The center of gravity of A will be taken as the estimated location of the sensor, because even if all the incorrect rings have no intersections with A and all the incorrect rings overlap at one place, the counter values associated with the grid elements at that incorrect place must be smaller than that of the elements in A. As a result, incorrect rings will not impact the

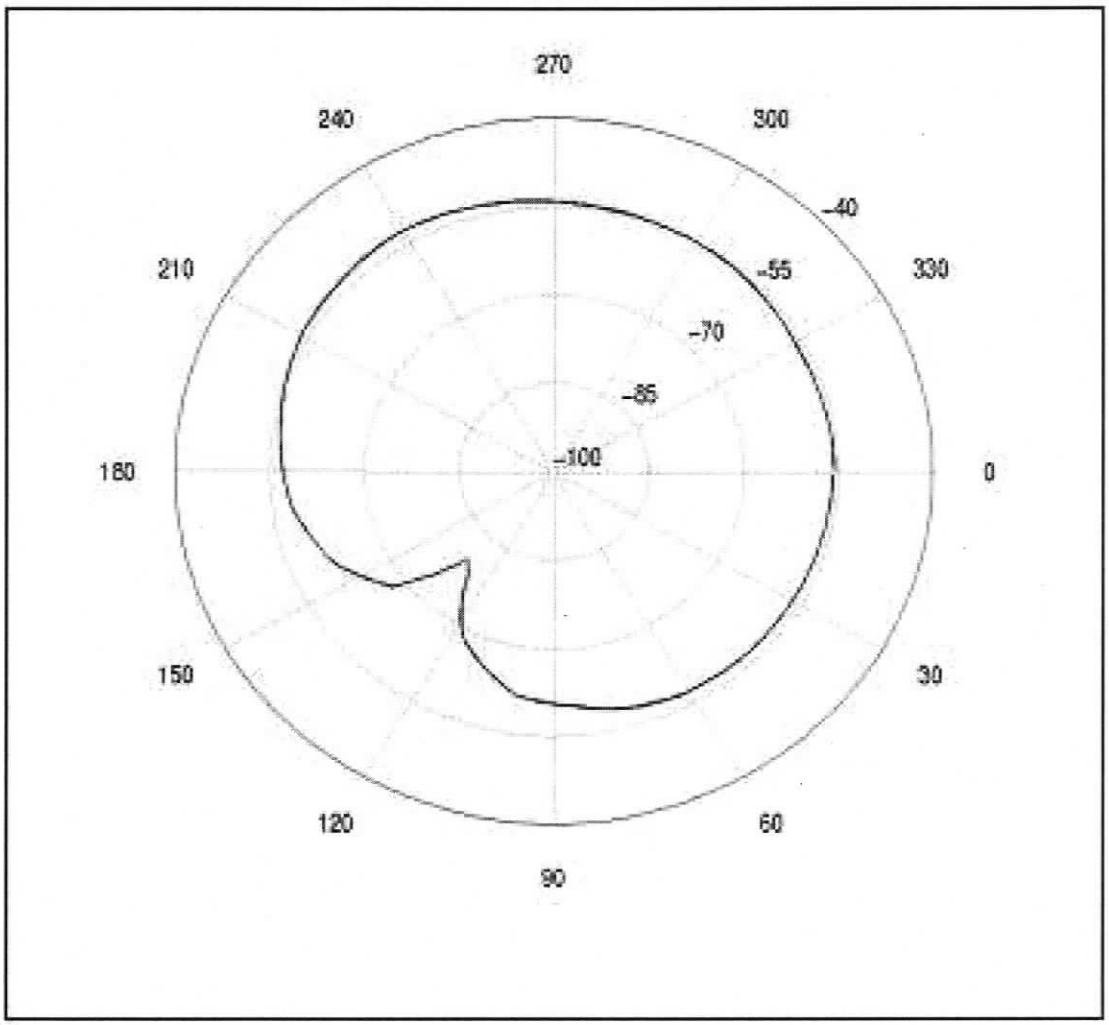

Figure 3.2. *External Antenna Radiation Pattern with MICA2 Mote Sensors (8m) [49, 50]*

location calculation. If some incorrect rings overlap with A, such as ring $R2$ in Figure 3.3, then the final intersection area may not be very accurate. Nevertheless, since the size of A is usually small when the number of in-range anchors is large enough, the center of gravity of the final intersection area will be close to the sensor. As such, even if ROCRSSI generates some incorrect rings, it can still yield fairly accurate location estimation.

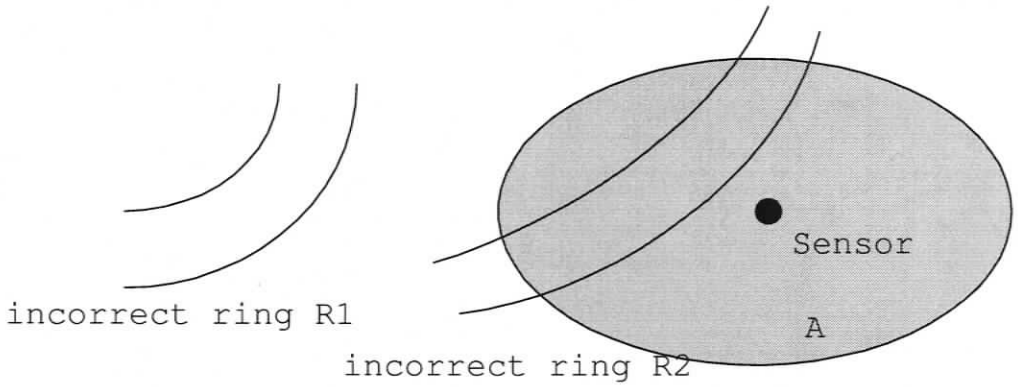

Figure 3.3. *Grid Scan Algorithm Alleviates the Influence of Incorrect Rings*

Second, the anchors that are used for sensor localization usually fall within a certain range. As shown in Figure 3.4, sensor node S will only use RSSI information from in-range anchors A, B, and C, implying that S must fall within the shadowed area. This is because in order to generate rings with A's RSSI values, S must fall within the transmission range of A, B, and C, and B and C must be able to hear A. The likely positions of anchors and sensors shown in Figure 3.4 make ROCRSSI work reasonably well since it is true in most cases that radio propagation is homogenous in a certain range of directions as shown in Figure 3.2.

In summary, while there may exist rare situations where ROCRSSI estimates are poor, the ROCRSSI algorithm is usually insensitive to radio irregularity and can achieve accurate estimates. This claim is supported by the experiments and simulation described in the sections to follow.

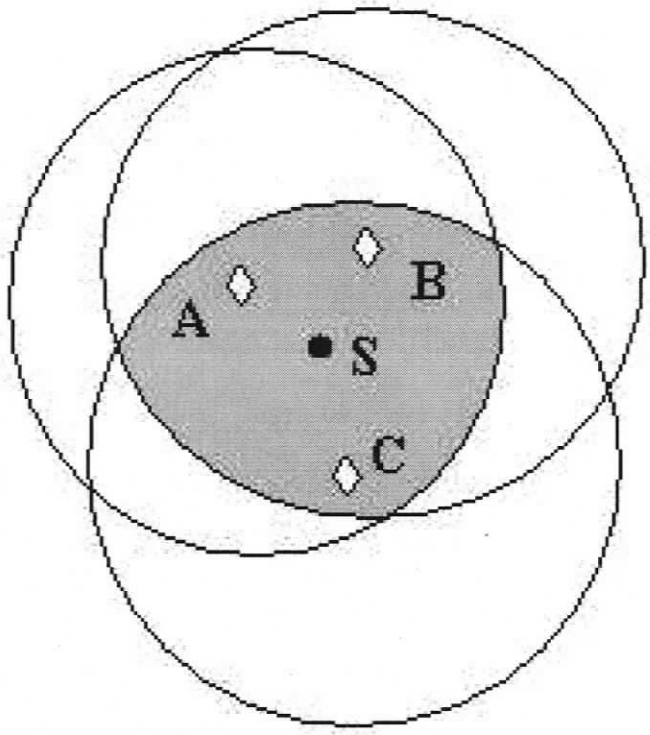

Figure 3.4. *Likely Positions of Anchors and Sensors that Can Use ROCRSSI*

3.4 Experimental Study Based on MICA2 Motes

3.4.1 Test Environment and Test Harness Setup

The ROCRSSI algorithm was implemented with MICA2 Motes by Scott [49]. For the completeness of this thesis, we include the brief introduction on the implementation and the test results. For more implementation details and more measurement results, please refer to [49].

The ROCRSSI localization method was tested with MICA2 Motes in two different environments: a rooftop test range and an outdoor yard. The rooftop range is an industrial RF emissions test range originally designed for measuring the RF emissions from telecommunications equipment. This range was used in order to avoid some of the unwanted and unpredictable RF affects that can occur indoors such as movement of people and furni-

ture. Although it is outdoors, this range exhibits some distinct multipath effects due to the presence of a copper floor, metal air-conditioners and a metal shed at one end of the range.

The outdoor yard is a grass yard on a small slope with large trees and shrubs on the perimeter and a one-storey wood-frame house along the side. This environment was chosen because it should have fewer multipath effects. Unlike the rooftop environment, there are no large metal surfaces to reflect the signal.

Each mote was placed on a wooden pillar, 63 cm from the ground. This distance was used to ensure that the antenna is close to the 0.6 Fresnel zone to reduce the impact of spectral reflections. The sensor mote was placed in the MIB510 programming cradle and a serial cable was connected to a laptop computer located outside of the range. This was done to ensure that there was no impact on the signal strength readings by having a human body close to the equipment under test.

The motes were placed in the same configuration in both the rooftop test and the outdoor-yard test, as shown in Figure 3.5. There are three anchor motes, labeled A, B and C. Each pair of anchor motes defines a circle with one mote as the center and the distance between the two as the radius. The sensor motes are labeled $S_1, S_2, \ldots, S_{11}$. Note that the test configuration, i.e., the number of anchors, the locations of anchors, and the locations of sensors, is to thoroughly test potential areas that a sensor may be located in. Obviously, some locations, such as the locations of S_2 and S_4, are intentionally selected not in favor of ROCRSSI since they either are out of any rings or fall within a large intersection area.

In both test environments, each test run lasted approximately 30 minutes and resulted in 21 to 33 cycles. Each cycle consists of 9 signal strength samples.

3.4.2 Design and Implementation

MICA2 mote sensors run TinyOS, an event-driven operating system. The sensor begins by sending a *startBeacon* message to the anchors. Then the sensor listens for beacons and RSSI reports. The anchors start the beaconing process after they get the *startBeacon*

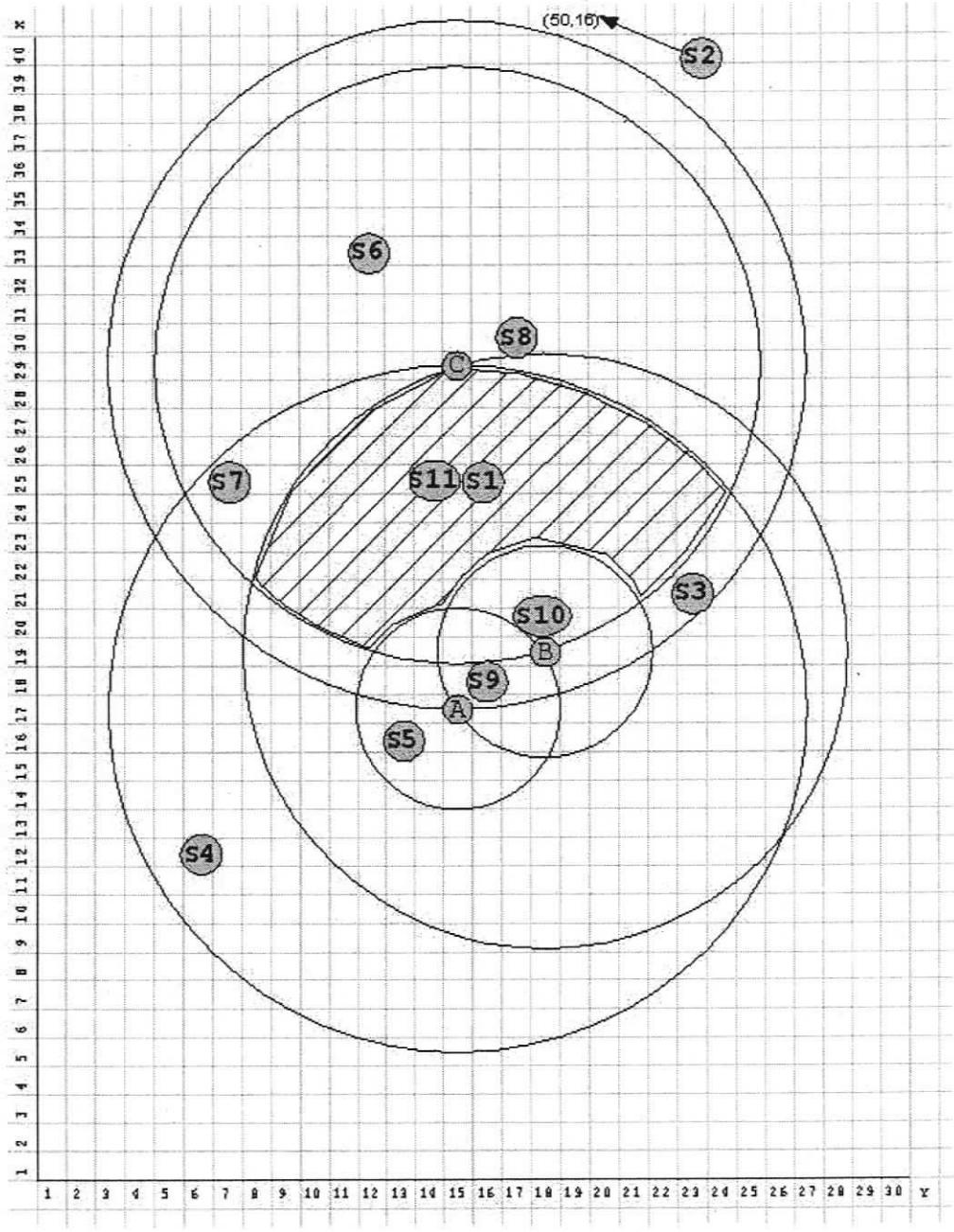

Figure 3.5. *Test Range Configuration, Anchor and Sensor Locations [49]*

message. Each anchor waits for a different length of time before it starts sending beacons to avoid collisions.

The anchors then send periodic beacons while at the same time listening for beacons from other anchors. Once they have heard all the beacons from another anchor, they broadcast an RSSI report to the sensor with the average RSSI heard from that anchor.

The message exchange between the motes is shown in Figure 3.6. Because we found that the sensor was not always getting all the RSSI reports from the anchors, we implemented an acknowledgement scheme: the sensor sends an acknowledgement when it receives an RSSI report from an anchor. If the anchor does not receive an acknowledgement, it retransmits the RSSI report. Note that this acknowledgement mechanism is for test purposes only. In real applications, a sensor might omit acknowledgements to save energy. As long as a sensor collects enough beacon messages from anchors, it can estimate its location. While the loss of several beacons may delay the estimation, it will not have a large impact on the estimation accuracy.

For testing purposes the beaconing process restarts after it is done. There is a short delay before the restart. When the anchors receive the restart message, they reinitialize and start the beaconing process again. Using this approach, a test can be started and restarted at any time. If the sensor mote is reset (using the hardware reset button) it will send out a start beaconing message that will cancel any beaconing that is currently in process at the anchors.

3.4.3 Test Results and Analysis

3.4.3.1 Verification

The purpose of this test was to verify that the ROCRSSI algorithm would work correctly given a small number of test cases. We kept the anchor locations the same and looked at a number of different regions that the sensor could be in, as shown in Figure 3.5. In these tests, we used the distance between the nodes to generate an ideal RSSI value. These values

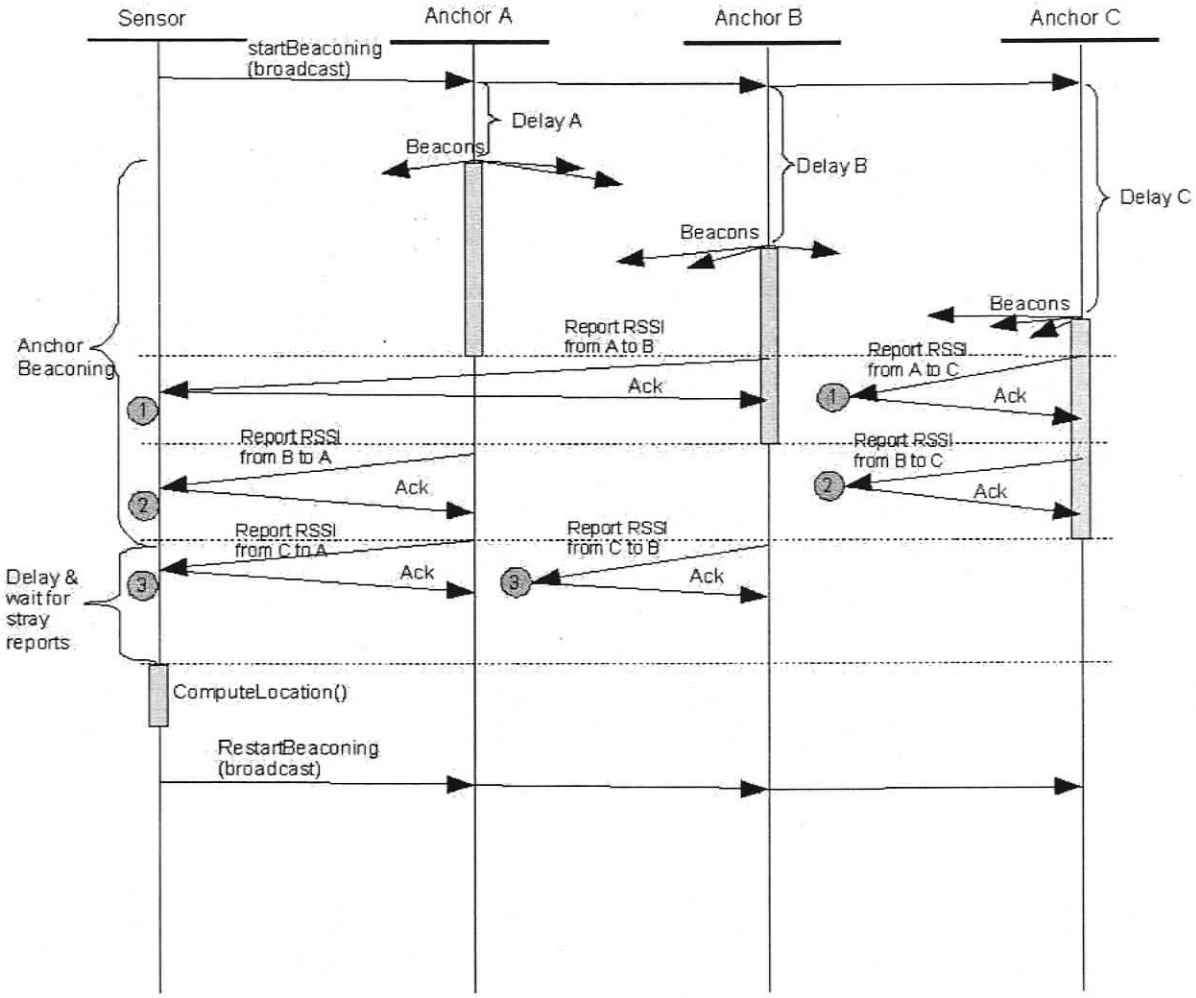

Figure 3.6. *Beaconing and RSSI Report Message Sequence Diagram [49]*

had to be inverted so that we had a higher number for a closer distance. We selected RSSI equal to 200 minus the distance. With this method, we made up a set of synthetic test files to test ROCRSSI.

We use the synthetic test files to see if the algorithm could properly identify the area containing the sensor. The result was a predicted location for each sensor: the location that the ROCRSSI algorithm should choose if the difference between the signal strength measurements is consistent with our assumptions about signal propagation. Table 3.1 shows the results of the preliminary verification, comparing the predicted and actual locations. Note that S_2 is a "undetermined node" since it is outside any ring. The estimate of S_2 is based on the fact that it falls outside any ring but within the boundary of the test area. This table also compares the average error for all sensor positions as well as the average error with the undetermined node removed.

Since we use synthetic test files to generate ideal RSSI values, the results in Table 3.1 represents the best results ROCRSSI can achieve in the test configuration shown in Figure 3.5. Note that in our calculation the granularity in the grid-scan algorithm is 0.5 meter.

3.4.3.2 Measurement Results

Compared to the results in Table 3.1, results with real measurement data have larger errors for some sensor nodes such as S_2, S_6, S_5, which fall outside any ring, fall only within one or two circles, or are close to the boundary between two regions. Under these situations, there is a chance that ROCRSSI will either use a large overlapping area or put the sensor in the wrong region. We stress again that the test configuration is selected to test all potential scenarios. Nevertheless, even with this test configuration, our real measurement indicates that some sensors such as S_1 and S_{10} have estimation accuracy close to the best results in Table 3.1.

For each sensor, we tested the ROCRSSI algorithm at least 22 times. Note that we intentionally omit the statistical results of total sensors, e.g. average error and standard deviation. Such calculation is meaningless since one "undetermined node" such as S_2 will

Table 3.1. *Verification Results with Ideal RSSI Values [49]*

Sensor Position	Anchor Positions			Predicted Location	Actual Location	Error (m)
	A	B	C	(X, Y)	(X,Y)	
S_1	(17, 15)	(19, 18)	(29, 15)	(24, 16)	(25, 16)	1.00
S_2	(17, 15)	(19, 18)	(29, 15)	(24, 24)	(50, 16)	27.20
S_3	(17, 15)	(19, 18)	(29, 15)	(20, 16)	(21, 23)	7.07
S_4	(17, 15)	(19, 18)	(29, 15)	(13, 14)	(12, 6)	8.06
S_5	(17, 15)	(19, 18)	(29, 15)	(15, 15)	(16, 13)	2.24
S_6	(17, 15)	(19, 18)	(29, 15)	(32, 15)	(33, 12)	3.16
S_7	(17, 15)	(19, 18)	(29, 15)	(25, 10)	(25, 7)	3.00
S_8	(17, 15)	(19, 18)	(29, 15)	(32, 15)	(30, 17)	2.83
S_9	(17, 15)	(19, 18)	(29, 15)	(18, 16)	(18, 16)	0.00
S_{10}	(17, 15)	(19, 18)	(29, 15)	(20, 18)	(20, 18)	0.00
S_{11}	(17, 15)	(19, 18)	(29, 15)	(24, 16)	(25, 14)	2.24
Average Error = 5.16						
Average Error with undetermined node S_2 removed = 3.18						

make the error extremely large. Instead, we list the detailed measurement results. To save space, we only list part of our measurement results. In Table 3.2, we compare the measurement results of S_1 with the rooftop range and the yard-range. Table 3.3 lists the measurement results of S_8, S_9, and S_{10} with the yard-range.

3.5 Simulation Study In Large-Scale Networks

3.5.1 Radio Model

For cost reasons, we did not test the ROCRSSI algorithm in a large-scale sensor network. Instead, we used simulation to investigate the performance of ROCRSSI localization in large-scale networks with different settings, i.e., network density, radio irregularity, and anchor deployment strategy.

As demonstrated in [23, 49, 67], radio propagation exhibits the feature of non-isotropic path losses. Thus, to closely approximate a real network environment, we do not use the perfect circular radio model in our simulation. Instead, we adopt and extend the Degree Of Irregularity (DOI) radio model [23] in our simulation. In this model, the DOI value is defined as the maximum range variation per unit degree change in the direction of radio propagation. Large DOI values represent large variation of radio irregularity. Unlike the previous DOI model, we do not assume any lower bound on radio irregularity, i.e., even close to the sender, radio irregularity may still be present. This model makes localization challenging. The extended DOI model can calculate the possible received signal strength at any point within the maximum radio range of a sender.

In the extended DOI model, the DOI value is still used to adjust the degree of radio irregularity. The signal strength is $C \times K(\theta)/d^2$ where C is a constant, d is the distance between the receiver and the sender and is smaller than the maximum radio range of the sender, and $K(\theta)$ is the coefficient representing the radio propagation feature in the specific direction θ. Thus the radio irregularity is expressed by different $K(\theta)$'s associated with

Table 3.2. *Comparison of Test Results of S_1 with Different Test Ranges [49]*

Test #	Estimated Position (X,Y)		Error (m)	
	Rooftop Range	Yard Range	Rooftop Range	Yard Range
	(units = 0.5m)	(units = 1m)		
1	(23, 16)	(23, 16)	1	2
2	(24, 16)	(23, 16)	0.5	2
3	(24, 16)	(23, 16)	0.5	2
4	(24, 16)	(23, 16)	0.5	2
5	(24, 16)	(23, 16)	0.5	2
6	(24, 16)	(23, 16)	0.5	2
7	(24, 16)	(23, 16)	0.5	2
8	(24, 16)	(23, 16)	0.5	2
9	(24, 16)	(23, 16)	0.5	2
10	(24, 16)	(23, 16)	0.5	2
11	(24, 16)	(23, 16)	0.5	2
12	(24, 16)	(23, 16)	0.5	2
13	(24, 16)	(23, 16)	0.5	2
14	(24, 16)	(23, 16)	0.5	2
15	(24, 16)	(23, 16)	0.5	2
16	(24, 16)	(23, 16)	0.5	2
17	(24, 16)	(24, 24)	0.5	8.06
18	(24, 16)	(23, 16)	0.5	2
19	(24, 16)	(23, 16)	0.5	2
20	(24, 16)	(23, 16)	0.5	2
21	(24, 16)	(23, 16)	0.5	2
22	(24, 16)	(23, 16)	0.5	2
			Actual location of S_1: (25, 16)	
			Average error with rooftop range = 0.52m	
			Average error with yard range = 2.25m	

Table 3.3. *Test Results of S_8, S_9, and S_{10} with the Yard Range [49]*

Actual Location of S_8 : (30,17), Average Error: 7.07m
Actual Location of S_9 : (18,16), Average Error: 3.2m
Actual Location of S_{10} : (20,18), Average Error: 0.19m

Test #	Estimated Location (X, Y)			Error (m)		
	S_8	S_9	S_{10}	S_8	S_9	S_{10}
1	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
2	(24, 24)	(18, 13)	(20, 18)	9.22	3.00	0
3	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0
4	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0
5	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0
6	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
7	(23, 16)	(18, 13)	(24, 16)	7.07	3.00	4.47
8	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
9	(26, 20)	(18, 13)	(20, 18)	5.00	3.00	0
10	(23, 16)	(24, 16)	(20, 18)	7.07	6.00	0
11	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
12	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
13	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0
14	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
15	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0
16	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
17	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0
18	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0
19	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
20	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
21	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
22	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
23	(23, 16)	(19, 13)	(20, 18)	7.07	3.16	0
24	(23, 16)	(18, 13)	(20, 18)	7.07	3.00	0

different directions. For $\theta \in [0, 360)$:

$$K(\theta) = \begin{cases} 1 & \theta = 0 \\ K(\theta - 1) + Rand \times \text{DOI} & \theta \text{ is a positive integer} \\ K(t) + (\theta - s) \times (K(t) - K(s)) & \text{Otherwise} \end{cases}$$

In the formula, *Rand* is a random number uniformly distributed between $(-1, 1)$, $s = \lfloor \theta \rfloor$, and $t = \lceil \theta \rceil$.

3.5.2 System Parameters and Performance Measurements

In our simulation, we studied several system-wide parameters which directly affect the accuracy of location estimation of ROCRSSI:

- Radio transmission range (R): the maximum distance over which a sensor can successfully send messages. This value is used for normalization only. All distances including error estimation are normalized by R to ensure generally applicable results.
- Anchor to Node Range Ratio (ANR): the average distance an anchor beacon can travel divided by R. We assume that all anchors have the same ANR in our simulation.
- Anchor Number (AN): the number of anchors deployed. We use AN, rather than Anchor Percentage [23] because ROCRSSI does not depend on the density of sensors; with ROCRSSI sensor nodes do not exchange information.
- Anchor deployment strategies. We investigated four anchor deployment strategies within a square with edge length $10R$:
  1. *Even deployment at the edges of field.* The anchors are deployed evenly at the edges of field.
  2. *Random deployment at the edges of field.* The anchors are deployed uniformly at random at the edges of the field, and the number of anchors on each edge is the same.

3. *Uniform deployment inside field.* The field is partitioned into a grid with each grid element the same size, and anchors are evenly divided amongst these grid elements. The locations of anchors in each grid element are randomly selected.
4. *Random deployment inside field.* The anchors are distributed uniformly at random inside the square field.

We define the location estimation error as the Euclidian distance between the real location of a node and its estimated location. For each simulation scenario, 100 sensor nodes are uniformly at random deployed in the field and ten runs with different random seeds were executed and the results were averaged. We use average location error as the metric to evaluate the accuracy of location estimation. It is defined as the mean of location estimation errors collected over all determined sensor nodes in ten runs. A sensor node is determined if it falls within at least three rings (or circles). Otherwise, it is an "undetermined node."

3.5.3 Results and Analysis

Varying Deployment Strategy Figure 3.7 indicates, that no matter how DOI and ANR vary, uniform deployment inside field always has the best performance given the same number of anchors. Also, the performance of random deployment inside field is very close to that of uniform deployment inside field. Therefore, the random deployment has close to the best performance, which is useful because this deployment is the easiest to carry out in practice.

That deployment of anchors inside field has better performance than deployment of anchors at the edges of field can be explained as follows. With ROCRSSI, anchors on the edge reduce the coverage of rings since a ring will be cut by the edges and the usable range of a ring with ROCRSSI is only half of the ring at most. In contrast, placing anchors within the field can fully exploit a ring because the whole ring can be used potentially by several sensors.

Nevertheless, the differences of average location errors among different deployment

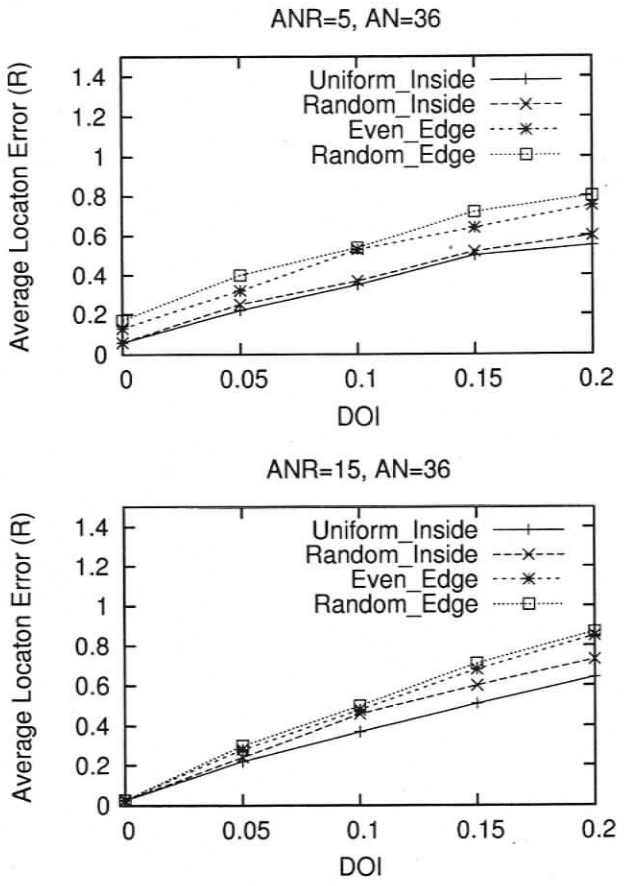

Figure 3.7. *Average Location Error under Different Deployment Strategies*

strategies are not significant when ANR is large. When deployment strategies do impair the performance of ROCRSSI, the impact can be controlled effectively by increasing the radio range of the anchors. In the simulation, we used uniform deployment inside field to obtain the best accuracy in each scenario.

Varying DOI Figure 3.8 shows that, with the increase of DOI, the average estimation error increases. This is because, with the increase of DOI, radio propagation has large variation along different directions. As a result, a ring generated by the comparison of signal strength may not represent the real area containing a sensor. Estimation based on incorrect rings is likely to have errors.

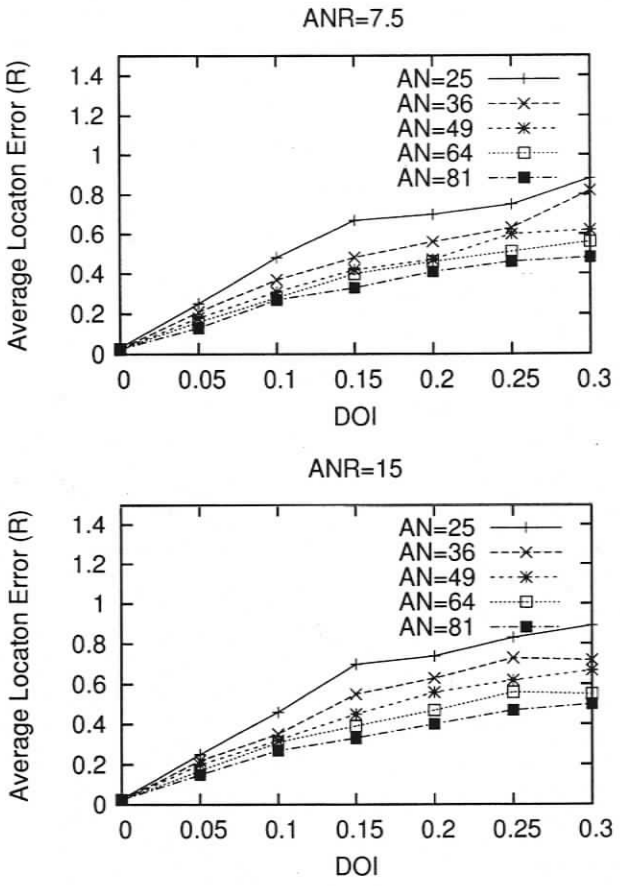

Figure 3.8. *Average Location Errors vs. DOI with Anchors Uniformly Deployed inside Field*

Fortunately, as explained in Section 3.3, the grid-scan algorithm used to calculate the center of gravity of the intersection area helps reduce the influence of wrong rings. As a result, ROCRSSI can achieve accurate estimation, even with irregular radio propagation with DOI as large as 0.2. For instance, from Figure 3.8, when AN= 81 and DOI= 0.2, ROCRSSI can achieve estimation with average error of 0.4R, which is acceptable for most applications [23].

As shown in Figure 3.8, the performance with ANR value of 15 is very similar to the performance with ANR value of 7.5, demonstrating that when ANR value is large enough, increasing this value further provides little benefit. This phenomenon is also confirmed by

later simulation as shown in Figure 3.10.

Varying AN Figure 3.9 shows that increasing AN can reduce the average estimation error. With more deployed anchors, a sensor node can hear more anchors and generate more rings. As a result, the intersection area becomes smaller, resulting in more accurate estimation.

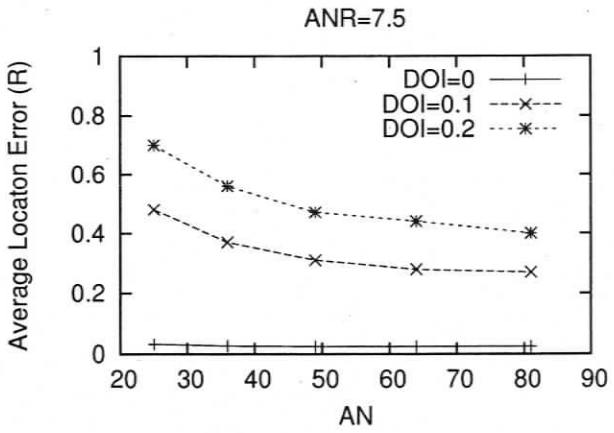

Figure 3.9. *Average Location Errors vs. AN with Anchors Uniformly Deployed inside Field*

Varying ANR Figure 3.10 illustrates that, with a perfectly circular radio propagation model (DOI= 0), increasing ANR reduces the average estimation error to the minimum possible with the grid granularity in use. The reason is that increasing ANR will increase the number of audible anchors per sensor node, and thus more rings will be generated. The circular radio propagation model guarantees that *each ring will certainly contain the sensor node* and can be used to narrow down the final intersection area, resulting in very accurate estimation.

Figure 3.10 also shows that, when radio propagation is irregular, with the increase of ANR, the average estimation error decreases and then oscillates around a fixed value. With an irregular radio propagation model, increasing ANR has two effects: (1) the number of

audible anchors per sensor node increases and (2) more incorrect rings might be generated by possible anchor pairs in different directions. When ANR is small, the first effect dominates, and thus the average estimation error can be reduced. When ANR is large enough, however, both factors can influence the final outcome, causing the oscillation.

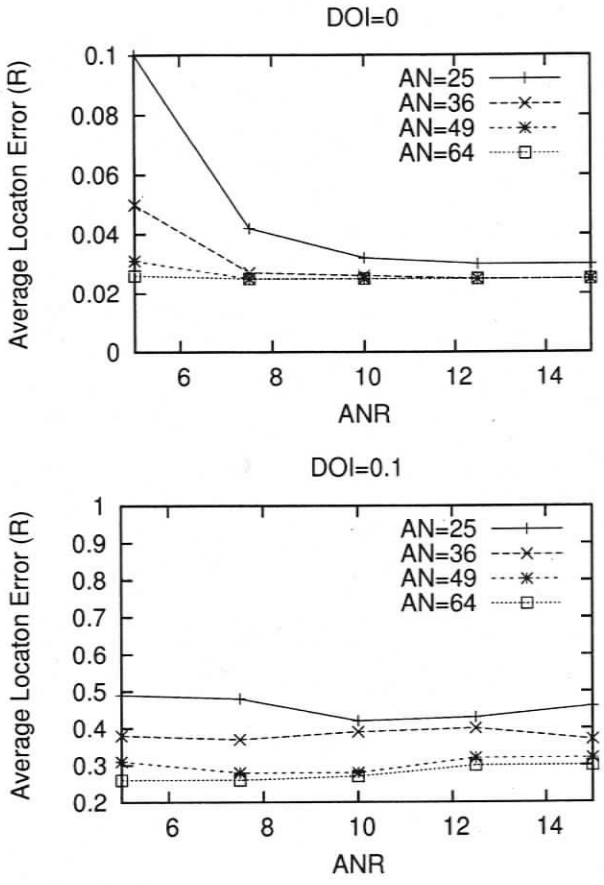

Figure 3.10. *Average Location Errors vs. ANR with Anchor Uniform Deployment inside Field*

3.6 Comparison between APIT and ROCRSSI

We were unable to find implementations for other range-free localization algorithms [23, 41, 42]. Consequently, simulation is the best available means for performance compari-

son between ROCRSSI and other range-free algorithms. We compare the performance of ROCRSSI and Approximate Point In Triangle (APIT) [23], because APIT [23] performs extremely well with randomly deployed wireless sensor networks and also because there are important similarities between APIT and ROCRSSI.

3.6.1 Introduction to APIT

APIT is a range-free localization method depending on anchor nodes. Because the number of anchor nodes is usually small, each node can use the Global Positioning System (GPS) to determine its location and can be equipped with a powerful radio transceiver. Using beacon messages from these anchors, APIT divides the neighborhood area of a sensor node into many overlapped triangles. The vertices of these triangles are anchors this sensor node can hear. The sensor node can narrow down the area in which it may reside by checking whether it falls within a triangle. Utilizing information from all anchors in range, the size of the intersection area of triangles can be reduced, and the center of gravity of this area is taken as the sensor's approximate location. As shown in Figure 3.11, the shadowed area is the intersection area of all perceived triangles and the center of gravity of the shadowed area is taken to be the sensor's location. A sensor node is considered "undetermined" if the node does not fall within any triangle.

Therefore, a sensor must have the ability to check whether it falls inside or outside a triangle. As shown in Figure 3.12, the perfect PIT test theorem [23] can be used for such a test:

> If there exists an adjacent point of point M, which is further from or closer to points A, B, and C than M simultaneously, then M is outside of $\triangle$ABC. Otherwise, M is inside $\triangle$ABC.

Since a node cannot move around to carry out the above test, neighboring sensor nodes exchange information to perform an approximate test, i.e., if a node has no neighboring nodes further or closer to three anchors simultaneously, this node is assumed to fall within

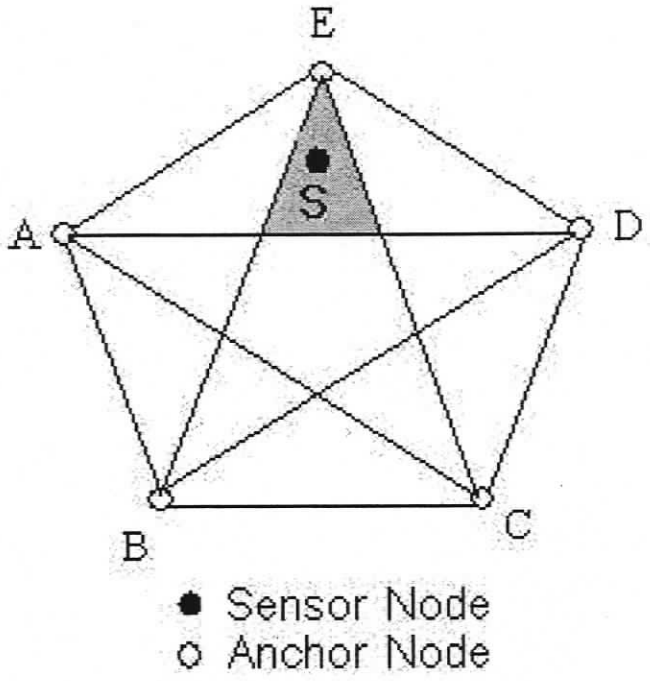

Figure 3.11. *An Example of Triangle Overlapping in APIT*

the triangle formed by the three anchors. Otherwise, it is assumed outside the triangle.

It is easy to see that the accuracy of the above test depends greatly on how many neighbors a node has. Fortunately, in realistic wireless sensor networks, sensor nodes are typically densely deployed. APIT's accuracy also requires that two sensor nodes can decide their relative distance to an anchor by comparing the received signal strength from the anchor.

APIT is sensitive to node density and requires a fairly high node density for a good performance [23]. In contrast, ROCRSSI does not perform message exchanges between neighboring sensors, making it insensitive to sensor radio transmission range and sensor density. While this is an advantage of ROCSSI over APIT, it makes it difficult to perform an absolutely fair comparison between APIT and ROCRSSI.

Nevertheless, from the results in [23], when sensor density is above 15 (i.e., a sensor node has 15 neighboring sensor nodes in average), APIT performs best and the estimation

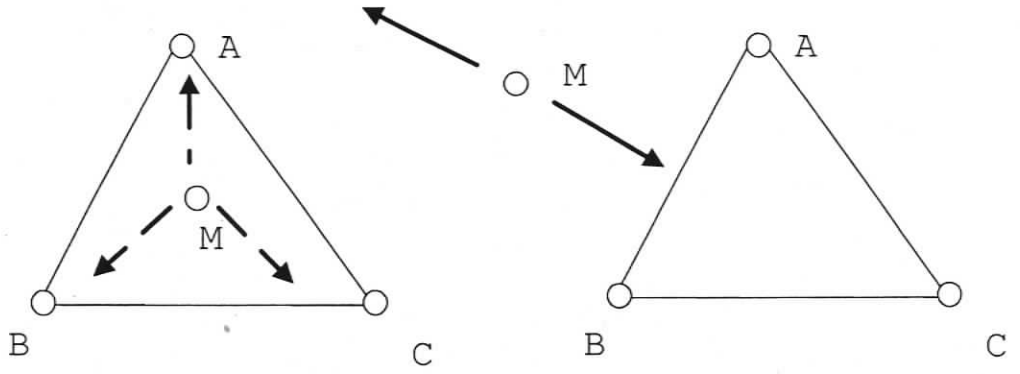

Figure 3.12. *Point In Triangle (PIT) Test*

accuracy is relatively stable. To get a reasonably fair comparison, we set sensor density to around 15 by adjusting maximum sensor radio range in the simulation. We use the same extended DOI radio model to test APIT. Moreover, we use the optimal anchor deployment strategy and ANR value for each method to make the two approaches achieve roughly the best performance with the same number of anchors.

3.6.2 Average Location Error

Figure 3.13 demonstrates that ROCRSSI always outperforms APIT in terms of average estimation error, whether the radio propagation is regular or irregular. This is because the intersection of rings *usually* has smaller size than the intersection of triangles. A simple example can be found in Figure 3.14, where it is easy to see that the size of the shadowed ring intersection area is smaller than the size of $\triangle$ABC.

The accuracy of APIT in our simulation is worse than that reported in [23] due to different radio models. In paper [23], the DOI radio model is used. In this model, radio irregularity has influence only on the number of immediate neighbors of each sensor node. If each pair of sensors can symmetrically communicate correctly, APIT assumes that they can always make a correct decision on whether one is closer to or further from a certain anchor. In the extended DOI model, however, radio irregularity can influence not only the

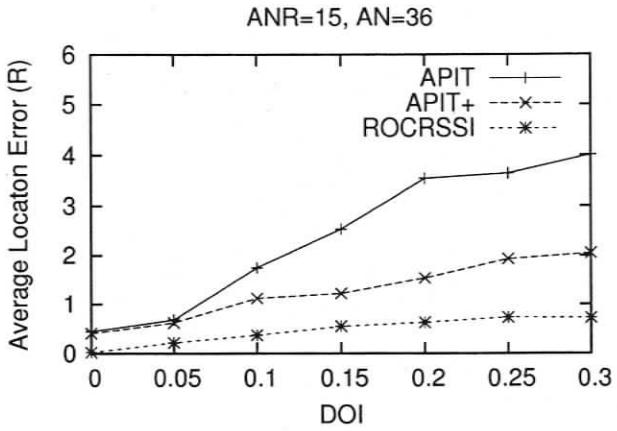

Figure 3.13. *The Approximate Comparison between APIT and ROCRSSI*

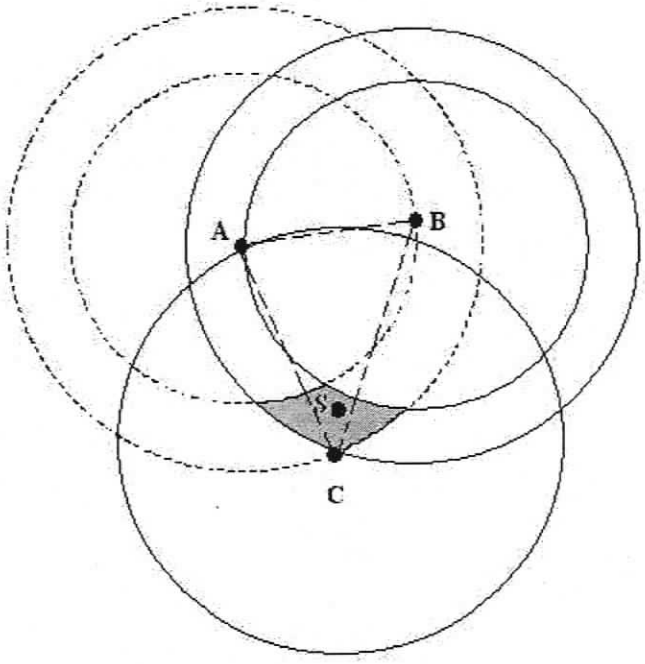

Figure 3.14. *An Example: Intersection of Rings is Smaller than Intersection of Triangles*

number of immediate neighbors of each sensor, but also the correctness of their decision on their relative distance to a certain anchor by the comparison of received signal strength. Therefore, in the extended DOI model, APIT will have more errors on the decision of whether a node is inside or outside a certain triangle. This factor can significantly decrease the accuracy of APIT, resulting in worse performance than that claimed in [23].

We also simulated an improved version of APIT, denoted as APIT+ in Figure 3.13. Using the grid-scan algorithm, when a triangle is picked up by the sensor node, only those grid elements within the maximum range of all anchors in range are added. This method improves the performance of APIT in face of radio irregularity with extra checking for each grid element.

3.6.3 The "Undetermined Nodes" Problem

Besides the improvement in accuracy, ROCRSSI can eliminate the inherent "undetermined nodes" problem in APIT, regardless of the anchor deployment strategy. We verify this claim by considering the worst cases in ROCRSSI.

We have studied all four deployment strategies with ROCRSSI and observed that random deployment of anchors inside field can yield more "undetermined nodes" than uniform deployment of anchors inside field, and random deployment of anchors at the edges of field can yield more "undetermined nodes" than even deployment of anchors at the edges. Therefore, we only present the results for two deployment strategies: random deployment of anchors inside field and random deployment of anchors at the edges of field.

Figure 3.15 shows the ratio of "undetermined nodes" under the above deployment strategies with different ANR and DOI values. The figures show that, if ANR is larger than a threshold value (for example, 7.5 in the simulation), there will be no "undetermined nodes."

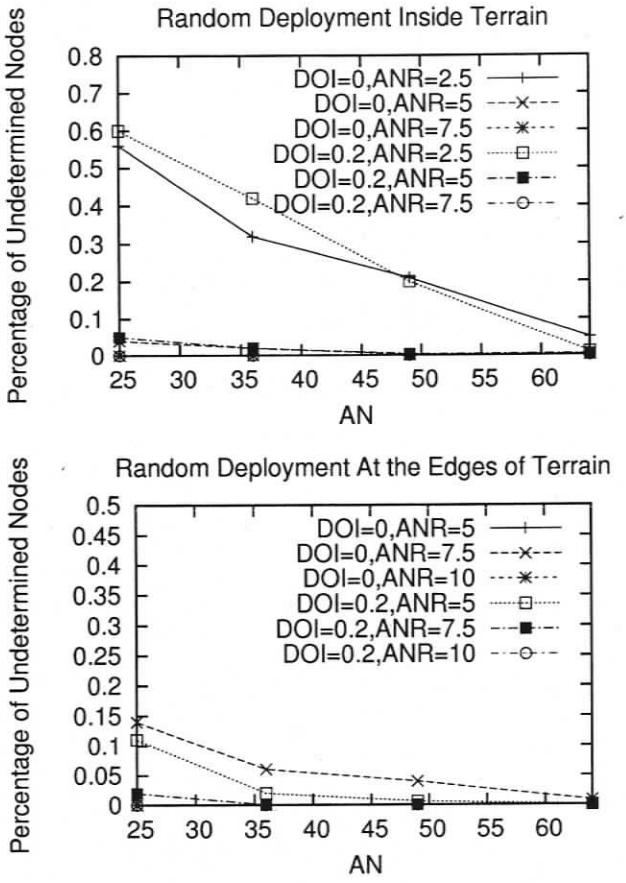

Figure 3.15. *Percentage of "Undetermined Nodes" ("missing" curves mean no "undetermined nodes.")*

3.6.4 Communication Overhead

Consider a sensor network with N anchors and M sensor nodes. Suppose that each broadcast from a sensor node costs 1 energy unit. Since the radio range of each anchor node is ANR times of that of a sensor node, each broadcast from anchors costs roughly $(\text{ANR})^2$ energy units if we assume a free-space radio propagation model. In APIT, all nodes broadcast at least once under a collision-free situation, so the total number of broadcast messages from sensor nodes is $\Theta(M)$ and that from anchor nodes is $\Theta(N)$, where Θ means asymptotically tight bound. Therefore, the total energy consumption is $\Theta(M) + \Theta(N) \times (\text{ANR})^2$

energy units. In ROCRSSI, only anchors need to broadcast and sensor nodes do not need to exchange any information for localization. So the total number of broadcast messages is $\Theta(N)$ and the total energy consumption is $\Theta(N)\times(\text{ANR})^2$ energy units. In cases where $M \gg N$ and ANR is not very large, ROCRSSI consumes much less energy than APIT with the same number of anchors deployed.

3.7 Summary

Range-free localization is promising in wireless sensor networks. In this chapter, we present a new range-free localization method, which is based on overlapping rings to estimate sensors' location. The rings are formed with the comparison of received signal strength from anchor nodes. We performed simulation study to investigate its performance in large-scale networks. Besides, our method was implemented on broadly-used MICA2 Motes and the in-field experiment was conducted in small-schale networks by Scott [49]. Both the experimental results and simulation results demonstrated that our range-free localization method has the following appealing features:

- It does not require sensor nodes to send out control messages and thus poses very small overhead on sensor nodes.
- It generates small intersection area, resulting in accurate location estimation.
- It can eliminate the "undetermined nodes" if anchors' radio range is large enough.
- It is resilient to irregular radio propagation patterns.
- It can achieve better performance with less communication overhead, compared to another well-known range-free method, APIT.

Chapter 4

Joint Scheduling Service for Sensing Coverage and Network Connectivity

4.1 Introduction

In previous chapter, we propose a method to make each sensor node obtain its *accurate* location. However, obtaining *accurate* locations is not free. In this chapter, we deal with a challenging task: Without *accurate* location information, how can we schedule sensor nodes to save energy and meet both constraints of sensing coverage and network connectivity? This problem should be solved in a lot of applications with WSNs, such as the detection of chemical attacks or the detection of forest fire. We use the following example to stress the importance of solving this problem.

Example: Imagine that a wireless sensor network is deployed to detect forest fire. The network should be able to detect the outbreaks of wild fire at any location within the monitored region with a high probability and report the outbreaks to the data collection center (also known as the sink node) with a small delay. In this example, sensing coverage, network connectivity, and energy efficiency are equally important: a large sensing coverage is to meet the users' requirement that an event can be detected with a high probability; network connectivity is to meet the users' requirement that the detected event can be delivered to the sink node; energy efficiency is to meet the users' requirement that the network should keep its operation as long as possible after the deployment. A good scheduling

scheme should achieve energy efficiency under the constraints of sensing coverage and network connectivity.

The difficulty in the above joint scheduling problem is that a sensor's sensing range is totally independent of its radio transmission range. It is generally hard to combine and solve the two problems together. This is the reason that although scheduling based on sensing coverage [2, 27, 36, 40, 54, 59, 62, 64] and scheduling based on network connectivity [8, 20, 60] have been studied extensively, very little work has been devoted to solving the joint problem. Since it is usually costly to obtain and maintain the location information of each sensor node, the joint scheduling problem is even more difficult if the location of each sensor node is unknown.

In this thesis work, we are motivated to provide a solution to solve the joint scheduling problem under both sensing coverage and network connectivity constraints without the availability of per-node location information. Specifically, we aim at designing a scheduling scheme that has the following features at any given time:

1. The sensing coverage is above a given requirement.
2. All the active sensor nodes are connected.
3. Each active sensor node knows at least one shortest or nearly shortest route to the sink node.

Our contributions to solving this problem are as follows. First, we propose a randomized scheduling algorithm and present the analytical results to illustrate the relationship among achievable coverage quality, event detection probability, event detection delay, energy saving, and node density. We also demonstrate that the randomized algorithm is resilient to time asynchrony if the network is sufficiently dense. Such a feature is indispensable for a practical scheduling algorithm since precise time synchronization is very hard for large sensor networks [17] and requires extra communication and energy consumptions. Second, we propose a rule to turn on extra sensors, so that the network connectivity is guaranteed. The rule permits each sensor node to decide whether and when it should turn

on in addition to the time slot scheduled by the randomized scheduling algorithm. Third, with our joint scheduling method, each active node knows at least one path to the sink. In this sense, the routing problem is simultaneously solved with the maintenance of network connectivity and no extra routing protocols are needed. Finally, we carry out performance evaluation and demonstrate that the joint scheduling method can achieve user-specified coverage quality with guaranteed network connectivity.

The rest of this chapter is organized as follows. In the next section, we introduce the network model. In Section 4.3, we introduce a randomized scheduling scheme that provides statistical sensing coverage. In Section 4.4, we propose a joint scheduling scheme that is based on the randomized scheduling algorithm and turns on extra sensor nodes, if necessary, for maintaining network connectivity. In Section 4.5, we analyze the relationship among coverage quality, detection delay, detection probability, energy saving, and node density. Section 4.6 presents the performance evaluation of our joint scheduling scheme. Finally we conclude this chapter in Section 4.7.

4.2 Network Model

There are many ways to organize the communication architecture of a sensor network. A sensor network could be in a hierarchical structure where each sensor communicates with a local cluster head and the cluster head communicates directly with the sink node. Alternatively, it could be in a flat communication structure as well, where each sensor has essentially the same role and relies on other sensors to relay its messages to the sink node via multi-hop radio communication. In this thesis work, we assume the flat communication architecture, where the joint problem of sensing coverage and network connectivity arises.

We consider *stationary* sensor networks in a two-dimensional field and assume that sensor nodes are randomly and independently deployed in a field. Compared to other sensor deployment strategies such as deployment in grids or in pre-defined positions, random deployment is much easier and cheaper [56]. Also, scheduling for a regular network topology

such as grids is simple and may not deserve further investigation.

We assume that a sensor node's radio transmission range is fixed and totally independent of its sensing range because of different hardware components involved. Unlike other work [51, 66] that assumes certain relationship between the radio range and the sensing range, our work makes no assumption on the relationship between them.

We do not assume accurate global time synchronization, which is an extremely hard task for large-scale sensor networks. Instead, our scheduling algorithm permits slight time asynchrony without performance degradation.

4.3 Randomized Scheduling for Coverage

We have designed a randomized scheduling algorithm for sensing coverage which has several prominent features [34, 36]. The algorithm does not assume the availability of any location or directional information. It is a purely distributed algorithm, thus scalable for large networks. It is also resilient to clock asynchrony and requires only a roughly synchronized clock, which significantly decreases the energy and communication overhead introduced by maintaining network-wide time synchronization. In the following we briefly summarize the basic idea of the randomized scheduling algorithm.

Assume that the sensor nodes constitute a set S. Given a number k, each sensor node randomly joins one of the k disjoint subsets of set S. Once the k subsets are determined, they work alternatively. At any given time, there is only one subset working, and all the sensor nodes belonging to this subset will turn on. The intuition is that when the network is sufficiently dense, each subset alone will cover most part of the field.

Figure 4.1 shows an example. Assume that we have eight sensor nodes (with IDs $0, 1, ..., 7$) randomly deployed in a rectangular area. Assume that the eight sensor nodes will be assigned into two disjoint subsets, S_0 and S_1. Each sensor node randomly selects a number (i.e., 0 or 1) and then joins the corresponding subset. Assume that sensor nodes $0, 2, 5, 6$ select number 0 and thus join subset S_0, and sensor nodes $1, 3, 4, 7$ select number

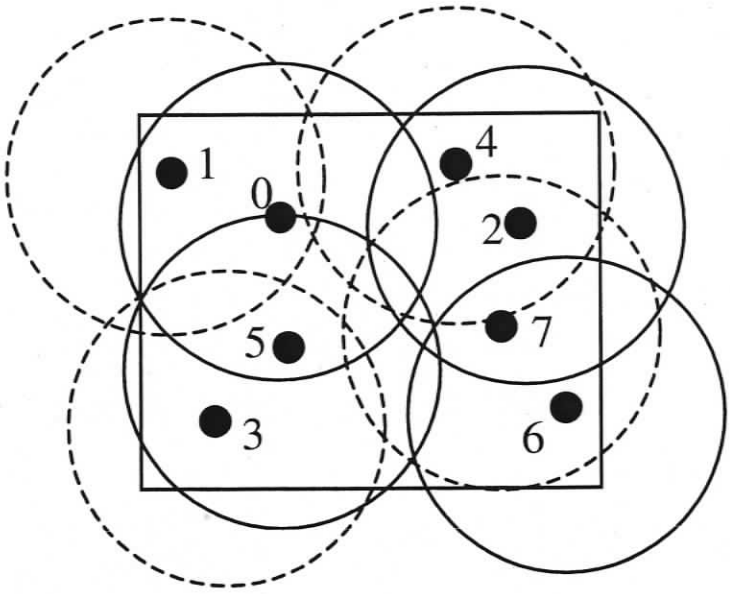

Figure 4.1. *An Example of the Randomized Coverage-Based Algorithm*

1 and thus join subset S_1. Then subsets S_0 and S_1 work alternatively, that is, when sensor nodes $0, 2, 5, 6$, whose sensing ranges are denoted as the solid circles, are active, sensor nodes $1, 3, 4, 7$, whose sensing ranges are denoted as the dashed circles, fall asleep, and vice versa.

4.4 Joint Scheduling: Random Coverage with Guaranteed Connectivity

4.4.1 Motivation

With the proposed randomized coverage-based scheduling scheme, the coverage quality can be guaranteed statistically by setting an appropriate subset number k. Yet, there is no guarantee on the network connectivity after scheduling. To operate successfully, a sensor network must be connected so that sensor nodes can report the detected events to the sink node. Therefore, in addition to sensing coverage, the sensor network must remain con-

nected, i.e., the active nodes should not be partitioned in any schedule of node duty cycles. We are hence motivated to enhance the above randomized scheduling algorithm such that both coverage quality and network connectivity can be met at any given time.

4.4.2 Extra-On Rule

Given that the total number of subsets is k, after the randomized coverage-based scheduling scheme, there are k sub-networks formed, each of which corresponds to a specific subset and consists of all the nodes assigned to that subset. However, there is no guarantee on the connectivity of each sub-network. The following extra-on rule ensures that each sub-network is connected, given that the original network is connected when all the nodes are on. Besides, it also guarantees that the path from any sensor node to the sink node has the global minimum hop count.

Assume that each sensor node knows its minimal hop count to the sink node when all the nodes are on. A sensor node A is called the upstream node of another sensor node B, if node A and node B are neighboring nodes and the minimal hop count of node A to the sink node is one less than that of node B. Node B is also called node A's downstream node.

Extra-on rule: If a sensor node A has a downstream node B, which is active in time slot i, and if none of node B's upstream nodes is active in that time slot, then node A should also work in time slot i. In other words, besides working in the duty cycles assigned by the randomized coverage-based scheduling, node A is required to work in extra time slots, e.g., time slot i in this case.

This rule requires each sensor node to maintain its minimum hop count to the sink node and the list of its upstream nodes. We stress that the minimum hop count is used to label the relative location information among sensor nodes. Since we focus on static sensor networks only and the failure of certain nodes does not influence such relative relationship, our joint scheduling based on the extra-on rule works correctly in face of network failure without requiring periodical update of the minimum hop count values. The method of collecting the hop count information and its energy cost will be addressed in detail in Section 4.4.4.

It is natural to realize that the extra-on rule may cause problems of synchronization and large overheads due to the dependency among nodes. These potential problems, however, have been carefully avoided in our protocol design. The details will be provided in the following sections.

4.4.3 The Correctness of the Extra-on Rule

Proposition 1: Given that the original network is connected. Applying the extra-on rule to each sub-network obtained with randomized coverage-based scheduling ensures that each sensor node has a shortest path to the sink node in the sub-network.

Proof: Let d_i denote the shortest hop distance from sensor node i to the sink node ($d_i \geq 1$). Let sensor node set S_l^i denote the sensor nodes that are on one of the shortest paths from sensor node i to the sink node and have a path to the sink with length l ($1 \leq l \leq d_i - 1$) in the original network. Since the original network is connected, none of the S_l^i ($1 \leq l \leq d_i - 1$) is empty. Also let p_l^i denote one sensor node in the set S_l^i.

If $d_i = 1$, sensor node i has a shortest path to the sink with one hop. If $d_i > 1$, with the extra-on rule, in any time slot, there exists at least one node p with hop distance $d_i - 1$ belonging to $S_{d_i-1}^i$ that is active and connected with sensor node i. Recursively, in any time slot, for any sensor node i, there is a path $\{p_{d_i-1}^i, p_{d_i-2}^i, \ldots, p_1^i\}$ connecting sensor node i and the sink node. Since the length of this path is equal to d_i, this is a shortest path. ■

Based on Proposition 1, after using the extra-on rule, each sensor has a path to the sink node. We can therefore get the following proposition:

Proposition 2: Given that the original network is connected. Applying the extra-on rule to each sub-network obtained with randomized coverage-based scheduling ensures the connectivity of each sub-network.

4.4.4 The Joint Scheduling Method in Detail

The joint scheduling method ensures the coverage quality and network connectivity simultaneously and has the following steps.

Step 1: Select a subset randomly

Initially, each sensor node generates a random number i between 0 to $k-1$ (0 and $k-1$ inclusive) and assigns itself to subset i. This is exactly the same as in the randomized coverage-based scheduling scheme.

Step 2: Propagate minimum hop count

This step starts from the sink node at the time when it broadcasts a HOP advertisement message to its immediate neighboring sensor nodes. Each HOP advertisement message contains the minimum hop count to the sink, the nodeID and its subset decision. In the packet broadcast from the sink, the minimum hop count is set to 0. Initially, the minimum hop count to the sink is set to infinity at each sensor node.

Each node, after receiving a HOP advertisement message, will put the message in its buffer. It will defer the transmission of the HOP message after a backoff time and only re-broadcasts the HOP message that has the minimum hop count. Before the re-broadcast of the HOP message, the hop count value in the HOP message is increased by 1. With this method, HOP message broadcasts with a non-minimal hop count will be suppressed if the HOP message with the actual minimal hop count arrives before the backoff time expires. The number of broadcasts from each sensor node depends on the length of backoff time. Increasing the backoff time will significantly decrease the number of broadcasts. Although a large backoff time value will increase the total time required for the completion of this step, we argue that it is an effective solution because this step is a one-time task for static sensor networks and the energy is the most precious resource for sensor nodes.

If no packets are lost, our method can guarantee that at the end of this step, each sensor node will obtain the minimum hop count to the sink node. In practice, packets may be lost due to collisions or poor channel quality. Nevertheless, packet losses will not impact the successful operation of our joint scheduling scheme, i.e., the network will still be connected

even if some nodes may have only a nearly shortest path to the sink node. Our simulation results in Section 4.6.2 demonstrate that the number of nodes without knowing the actual minimum hop count at the end of this step is negligible even in the presence of packet losses.

Step 3: Exchange information with local neighbors

Each sensor node locally broadcasts its minimum hop count, its nodeID, its subset decision, the nodeIDs of its upstream nodes and their subset decisions. The upstream nodes are the nodes from which the current node receives its minimum hop count. Each sensor node records and maintains all the information it receives from its immediate neighbors.

Step 4: Enforce the extra-on rule to draw up working schedule

Based on the extra-on rule and the information from Step 3, each sensor node decides the extra time slots it has to remain active to ensure network connectivity and updates its working schedule accordingly. Then the updated working schedule is broadcasted locally to neighboring sensor nodes.

It is easy to see that the update of a sensor node's working schedule can impact the working schedule of its upstream nodes and the neighboring nodes with the same minimum hop count to the sink. To minimize the number of broadcasts of working schedule updates, it is desirable that a sensor node updates its working schedule after it receives all of the latest working schedules from its downstream nodes. This is exactly the reverse process of Step 2. Therefore, a backoff-based broadcast scheme similar to that in Step 2 can be applied here.

As an example, assume that the network consists of one sink node and four sensor nodes, A, B, C, and D as shown in Figure 4.2. D is three hops away, B and C are two hops away, and A is one hop away from the sink node. Assume that at the end of Step 1, A, B, C and D pick up their working time slots as time slots 1, 2, 3 and 4, respectively. Assume that D broadcasts its updated working schedule first. B and C are its upstream nodes and in Step 3 they know that node D does not have an upstream node working in time slot 4. After they receive D's working schedule update, there are several possibilities.

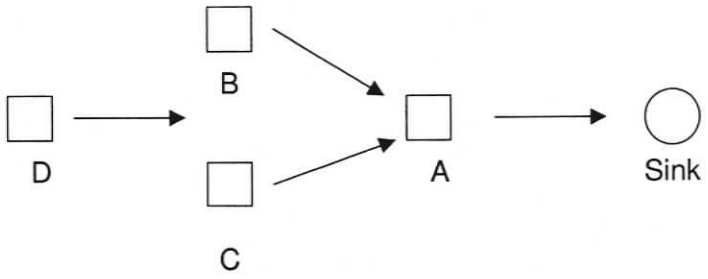

Figure 4.2. *An Example of the Extra-on Rule*

1. Case 1: Suppose that B and C can hear each other. If B broadcasts its working schedule prior to C, C can hear B's updated working schedule and knows that D has an upstream node working in time slot 4. So C will not schedule itself to work in time slot 4. In this case, B will work in time slots 2 and 4 and C will work in time slot 3 only. Likewise, if C broadcasts its working schedule prior to B, C will work in time slots 3 and 4 and B will work in time slot 2 only.
2. Case 2: Suppose that B and C cannot hear each other. B and C will both work in time slot 4 to ensure network connectivity, no matter which node broadcasts first. Therefore, B will work in time slots 2 and 4 and C will work in time slots 3 and 4.

In both cases, based on the latest working schedules received from nodes B and C, node A will know that it has to work in time slots 2, 3 and 4 to ensure network connectivity. Therefore, A will work in time slots 1 to 4.

At the end of this step, each sensor node obtains its working schedule and it works according to this schedule.

4.4.5 System Overhead

In this section, we evaluate the system overhead of joint scheduling in terms of the average number of broadcasts from individual sensor node.

Step 1: No broadcast is needed in this step.

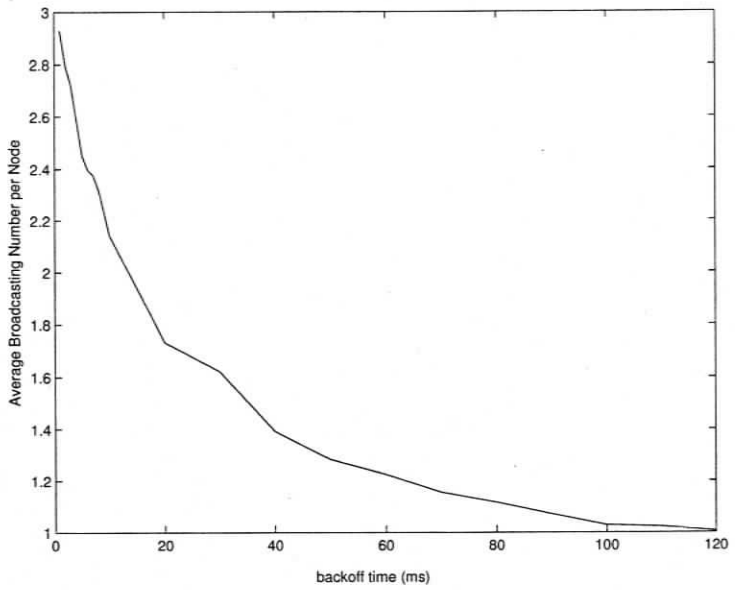

Figure 4.3. *Average Number of Broadcasts per Node in Step 2*

Step 2: The number of broadcasts from each sensor node depends on the length of the backoff time. We study their relationship via simulation. We deploy 1500 sensor nodes randomly in a 200 meters * 200 meters area and place the sink node at the center of the area. The radio range of each sensor node is fixed to 10 meters. We adopt the CSMA MAC layer protocol. To broadcast a HOP advertisement message, we assume that each sensor node has to capture the channel for 1 ms. Figure 4.3 illustrates the results with the backoff time from 1 ms to 120 ms.

From the figure, we can see that if the backoff time is large enough, each sensor node almost broadcasts only once in this step. The energy saving is at the cost of longer delay to complete this step, as shown in Figure 4.4. Nevertheless, since this is only a one-time task, the delay should not be tolerable. Actually, our simulation results indicate that if the backoff time is set to 120 ms, the time from the moment when the sink node broadcasts the HOP advertisement message to the moment when the last sensor node finishes the HOP advertisement message is below 2500 ms, which is acceptable. Figure 4.4 also indicates that using a too small backoff time does not necessarily reduce the total delay since a node

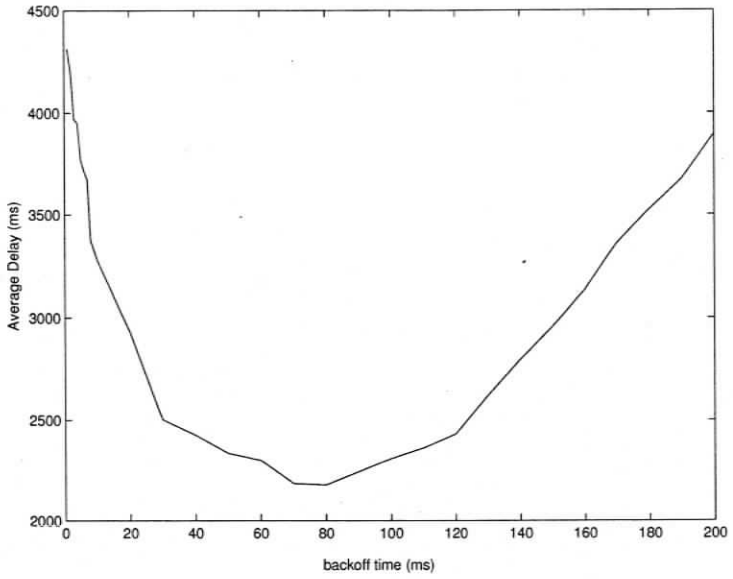

Figure 4.4. *Average Delay to Complete Step 2*

may need to broadcast multiple times if HOP messages with a smaller hop value arrive later.

Step 3: Each sensor node needs to broadcast only once to notify their neighbors.

Step 4: Since the propagation of the extra-on decisions is actually the reverse process of Step 2, we can expect that most of the sensor nodes will broadcast only once if the backoff time is appropriately set.

In conclusion, the overhead for most of the sensor nodes is three local broadcasts if the backoff time in Step 2 and Step 4 are large enough. The whole system setup process can be finished within several seconds, given a sensor network system similar to the one in our simulation.

4.4.6 Advantages of the Joint Scheduling Algorithm

First, the joint scheduling method can guarantee that the resulting coverage quality is above a given requirement, since the extra-on nodes actually increase the coverage quality pro-

vided by the randomized coverage-based scheme. Later performance evaluation demonstrates that the number of extra-on sensor nodes is not large in any time slot; hence good coverage with guaranteed connectivity is not at the cost of large energy waste.

Second, the route from each sensor node to the sink node has the minimum or nearly minimum hop count. This feature of our joint scheduling method roughly eliminates the extra energy consumption on data delivery with unnecessarily longer paths.

Third, with the joint scheduling method, the routing problem is simultaneously addressed with the connectivity problem. A sensor node not only has a route to the sink node, but also knows the upstream node in this route. Therefore, no additional routing protocols are needed.

Fourth, the overhead to set up the system is small. As demonstrated in the previous section, most sensor nodes need only three local broadcasts if the backoff times in Step 2 and Step 4 of the joint scheduling algorithm are set appropriately.

Finally, since each sensor node uses only local information to make its scheduling decision, the joint scheduling method is purely distributed and is scalable well to large and dense networks.

Note that our joint scheduling scheme exploits the redundancy in both sensing coverage and network connectivity. However, we cannot exclude the possibility that some nodes are critical nodes for connectivity, i.e., the network will be partitioned if these nodes are turned off. Therefore, to meet the requirement of network connectivity, our joint scheduling method has to turn on these critical nodes all the time. Therefore, these critical nodes might die sooner than other nodes due to their heavier workload. We point out that there is no method to solve this problem unless more nodes are deployed nearby these critical nodes to increase redundancy. Also, note that the joint scheduling is decoupled from the MAC layer protocol. Although the network topology is different from time slot to time slot, within a single time slot, the network topology is fixed and any MAC layer protocol can be used.

4.5 Performance Analysis

4.5.1 Performance Metric

There is a clear trade-off in the randomized scheduling algorithm. Generally, a larger k value means more subsets, and thus a subset can wait longer until its next turn to work. As such, the network can last longer. But a larger k value means smaller number of sensors in each subset and thus potentially worse network coverage. We need to select a proper k value so that the energy can be saved with desirable network coverage. For this, we need to clearly define the network coverage.

Definition 1: Coverage Intensity for a Specific Point. For a given point p in the field, we define the coverage intensity for this point as

$$C_p = \frac{T_c}{T_a}$$

where T_a is any given long time period and T_c is the total time during T_a when point p is covered by at least one active sensor.

It is obvious that C_p depends on both the number of sensor nodes deployed in the neighborhood of p and the scheduling scheme. Due to the randomness in the sensor deployment strategy and the scheduling scheme, C_p is a random variable. Hence, the expectation of C_p reflects the *average* time fraction when point p can be monitored. Notably, the expectation of C_p for any point inside the field is equal because sensors are independently and uniformly distributed in the field (For simplicity, we ignore the edge effect for large deployment area). Because of this reason, the expectation of C_p is a network-wide consistent metric and could be used to evaluate the coverage quality of the whole network.

Definition 2: Network Coverage Intensity. We define the network coverage intensity, C_n, as the expectation of C_p. That is, $C_n = E[C_p]$.

Since the main task of wireless sensor networks is to detect and report interesting events within the monitored field and the coverage intensity C_n reflects the probability that an event can be detected, C_n can be considered as the coverage measurement of sensor net-

works. The ideal value of C_n is 1, which indicates that with the probability 1 every point in the field is covered by at least one active sensor at any given time. But achieving this ideal value may require very dense deployment and is extremely expensive. Since different applications have different requirements on acceptable coverage intensity, a good scheduling scheme should set the number of simultaneous working sensor nodes merely enough to fulfill a given coverage requirement. In the sequel, we will investigate the relationship among achievable coverage, energy saving, and the minimum number of required sensor nodes.

4.5.2 Analysis on Coverage Intensity

For easy reference, all notations used in our probabilistic analysis are listed in Table 4.1.

Theorem 1: Without considering network connectivity, $C_n = 1 - \left(1 - \frac{q}{k}\right)^n$, where $q = \frac{r}{a}$ is the probability that each sensor covers a given point.

Proof: Suppose that a given point inside the monitored field is covered by s sensor nodes, which construct a set denoted as set S. The randomized algorithm will assign each sensor node in S to one of the k disjoint subsets randomly. Let's consider the question of how many subsets do not include any sensor in S. For the first subset, denoted as subset 0, it must miss all the s sensor nodes to let the above event happen. Since each sensor node hits the first subset independently with same probability of $\frac{1}{k}$,

$$\Pr\{S_0 \textit{ is empty}\} = \left(1 - \frac{1}{k}\right)^s$$

and thus

$$\Pr\{S_0 \textit{ is not empty}\} = 1 - \left(1 - \frac{1}{k}\right)^s$$

This probability is the same for all subsets by symmetry.

We define a random variable X_j. $X_j = 0$ if S_j is empty and $X_j = 1$ otherwise. Let $X = \sum_{j=0}^{k-1} X_j$ denote the total number of nonempty $S_j, (0 \leq j \leq k-1)$. Then

Table 4.1. *Notations*

Symbol	Description
n	the total number of deployed sensor nodes
T	the time duration of each time slot
a	the size of the whole field
r	the size of sensing area of each sensor
k	the number of disjoint subsets
s	the number of sensor nodes that cover a specific point inside the field
S	the set of sensor nodes that cover a specific point inside the field
s_i	the number of sensor nodes that belong to subset i and cover a specific point inside the field
S_i	the set of sensor nodes that belong to subset i and cover a specific point inside the field
C_p	coverage intensity for a specific point
C_n	network coverage intensity

$$E[X] = \sum_{j=0}^{k-1} E[X_j] = k \times \left[1 - \left(1 - \frac{1}{k}\right)^s\right]$$

According to the definition of C_p, the coverage intensity for point p, which is covered by s sensor nodes, is

$$C_p = \frac{E[X] \times T}{k \times T} = 1 - \left(1 - \frac{1}{k}\right)^s$$

Here s is a binomial random variable, and

$$\Pr\{s = j\} = \binom{n}{j} \times q^j \times (1-q)^{n-j}$$

where $q = \frac{r}{a}$ is the probability that each sensor covers a given point.

Therefore, using conditional expectation, the network coverage intensity C_n, which is the expectation of C_p, can be calculated as

$$\begin{aligned}
C_n &= E[C_p] \\
&= E\left[1 - \left(1 - \frac{1}{k}\right)^s\right] \\
&= 1 - \sum_{j=0}^{n} \left(1 - \frac{1}{k}\right)^j \times \binom{n}{j} \times q^j \times (1-q)^{n-j} \\
&= 1 - \sum_{j=0}^{n} \binom{n}{j} \times \left(q - \frac{q}{k}\right)^j \times (1-q)^{n-j} \\
&= 1 - \left(1 - \frac{q}{k}\right)^n \blacksquare
\end{aligned}$$

Corollary 1: For a given k, the lower bound on the number of sensor nodes required in the whole network to provide a network coverage intensity of at least t is

$$\left\lceil \frac{\ln(1-t)}{\ln(1-\frac{q}{k})} \right\rceil$$

where $q = \frac{r}{a}$.

Proof: Based on Theorem 1, if we predefine the value of k and we require that the network coverage intensity is no less than a threshold value t, we can compute the lower bound on the number of sensor nodes required to fulfill the task, by solving the inequality

$$1 - \left(1 - \frac{q}{k}\right)^n \geq t.$$

It is easy to see that

$$n \geq \left\lceil \frac{\ln(1-t)}{\ln(1-\frac{q}{k})} \right\rceil . \blacksquare$$

Theorem 1 and Corollary 1 illustrate clearly the relationship among the coverage, energy saving, and the minimum number of sensor nodes. For example, if we set $t = 0.9, q = \frac{\pi}{400}$, and $k = 3$, we can get $n \geq 878$ with Corollary 1, which means at least 878 sensor nodes are required in the whole network to achieve a network coverage intensity of no less than 0.9. This example also implies that, if we use 3-disjoint subsets and the ratio of each sensor's sensing range over the size of the whole monitored field is $\frac{\pi}{400}$, the average number of sensor nodes that cover a specific point is around 6 to meet a network coverage intensity of 0.9. Note that the value 6 is estimated by calculating $\frac{878\times\pi}{400} - 1$. From this example, it is really exciting that the randomized algorithm can achieve a reasonably high coverage requirement with a moderate network density.

Based on Theorem 1, we can easily get the following corollary:

Corollary 2: For a given n, the upper bound on the number of disjoint subsets to provide a network coverage intensity of at least t is

$$\frac{q}{1 - e^{\frac{\ln(1-t)}{n}}} = \frac{q}{1 - \sqrt[n]{1-t}}$$

where $q = \frac{r}{a}$.

Corollary 2 is very useful in dynamically adjusting the coverage of a sensor network after it is deployed. When the total number of sensor nodes is fixed, the network coverage intensity can be adjusted by changing the number of disjoint subsets k. This feature is extremely useful for practical sensor networks requiring adjustable measurement quality for energy saving. Using Corollary 2, we can easily map the coverage requirement to an

appropriate k value so that a perfect balance between energy conservation and coverage can be achieved. The coverage can be adjusted by a simple message flooding to inform all sensor nodes about the new k value.

4.5.3 Analysis on Detection Delay and Detection Probability

Besides coverage intensity, users sometimes may be interested in the detection probability and the average detection delay. The detection probability is defined as the probability that the occurrence of an event can be detected by one or more sensor nodes. The average detection delay is defined as the expectation of the time elapsed from the occurrence of an event to the time when the event is detected by some sensor nodes.

For the event lasting for a duration larger than $(k-1) \times T$, if there is at least one sensor node covering the occurrence location of the event, it will be detected with probability 1. Here k denotes the number of total subsets and T denotes the duration of each time slot. In this case, a short average detection delay is desirable. In the following, we investigate the average detection delay for events lasting longer than $(k-1) \times T$, as well as the detection probability for other shorter events.

4.5.3.1 Average Detection Delay

With random deployment, it is possible that there exist some blind points in the field, which cannot be covered by any sensor nodes. For these blind points, the detection delay is infinite. In the following analysis, we do not consider blind points since it is meaningless for such evaluation if a point cannot be covered all the time.

Theorem 2: Assume that an event arrives at any time slot with equal probability and lasts for a duration longer than $(k-1) \times T$. With the randomized coverage-based scheduling algorithm, the average detection delay for an event occurring at a point covered by s sensor nodes, is equal to $\frac{T}{2}\left[\left(\frac{k-1}{k}\right)^s + 2\sum_{i=2}^{k-1}\left(\frac{k-i}{k}\right)^s\right]$

Proof: Without lose of generality, we assume that the event arrives at time slot 0,

followed by time slots $1, 2, ..., k-1$. Time slot i $(0 \leq i \leq k-1)$ is associated with the working shift of subset i. Therefore, time slot 0 to time slot $k-1$ consist of a scheduling cycle. Let H_i be the event that none of the s covering sensors belongs to subset i and $\overline{H_i}$ be the event that at least one of the s covering sensors belongs to subset i. Therefore, the average detection delay, $delay_s$, can be calculated as

$$
\begin{aligned}
delay_s &= \sum_{i=1}^{k-1} \int_0^T \frac{1}{T} * Pr\left(H_0 \bigcap H_1 \bigcap \ldots \bigcap \overline{H_i}\right) * \\
&\quad (i * T - t)\, dt \\
&= \sum_{i=1}^{k-1} \int_0^T \frac{1}{T} * \left(1 - \frac{1}{k}\right)^s * \left(1 - \frac{1}{k-1}\right)^s * \ldots * \\
&\quad \left[1 - \left(1 - \frac{1}{k-i}\right)^s\right] * (i * T - t)\, dt \\
&= \sum_{i=1}^{k-1} \frac{(2i-1) * T}{2} \left[\left(\frac{k-i}{k}\right)^s - \left(\frac{k-i-1}{k}\right)^s\right] \\
&= \frac{T}{2}\left[\left(\frac{k-1}{k}\right)^s + 2\sum_{i=2}^{k-1}\left(\frac{k-i}{k}\right)^s\right] \blacksquare
\end{aligned}
$$

From the above expression, it is easy to see that the average detection delay for a specific point covered by s sensor nodes is influenced by three factors.

1. T: the time duration of each time slot, i.e., the working duration of each subset in one round of scheduling. The average detection delay increases with the increase of T. This is because a larger T will lead to a larger waiting time for the event to be detected by the active sensor nodes of the next working subset, if no node in the current subset can detect the event.
2. k: the number of total disjoint subsets. The average detection delay increases with the increase of k (the derivative of $delay_s$ regarding k is always positive when $k > 0$ and thus $delay_s$ monotonically increases with the increase of k). This is because increasing k will potentially increase the probability that no node in a subset can detect the event and hence prolongs the event detection delay.

3. s: the number of sensor nodes that cover point p. The average detection delay decreases with the increase of s. This is because a larger s value results in a less chance of generating a subset such that none of its nodes can cover p.

4.5.3.2 Detection Probability

For events lasting less than $(k-1) \times T$ and occurring at a point covered by s sensor nodes, we calculate its detection probability, dp_s.

Theorem 3: Let l $(< (k-1)T)$ denote the duration of an event occurring at a point covered by s sensor nodes. With the randomized coverage-based scheduling algorithm, the detection probability of the event, dp_s, is equal to $1-(1-\gamma_1)\beta_1 - \gamma_1\beta_2$, where $\gamma_1 = \frac{l}{T} - \lfloor \frac{l}{T} \rfloor$, $\beta_1 = (1 - \frac{\lceil l/T \rceil}{k})^s$, and $\beta_2 = (1 - \frac{\lceil l/T \rceil + 1}{k})^s$.

Proof: For an event with duration l $(< (k-1)T)$, it can span either $\lceil \frac{l}{T} \rceil$ or $\lceil \frac{l}{T} \rceil + 1$ time slots. Therefore, dp_s is equal to the probability that either case occurs, conditioned on the probability that in the corresponding case there are some nodes detecting the event.

$$
\begin{aligned}
dp_s &= (1 - (\frac{l}{T} - \left\lfloor \frac{l}{T} \right\rfloor)) \times \left[1 - (1 - \frac{\lceil l/T \rceil}{k})^s\right] + \\
&(\frac{l}{T} - \left\lfloor \frac{l}{T} \right\rfloor) \times \left[1 - (1 - \frac{\lceil l/T \rceil + 1}{k})^s\right] \\
&= 1 - (1-\gamma_1)\beta_1 - \gamma_1\beta_2,
\end{aligned}
$$

where $\gamma_1 = \frac{l}{T} - \lfloor \frac{l}{T} \rfloor$, $\beta_1 = (1 - \frac{\lceil l/T \rceil}{k})^s$, and $\beta_2 = (1 - \frac{\lceil l/T \rceil + 1}{k})^s$. Note that γ_1 is the probability that the event spans $\lceil \frac{l}{T} \rceil + 1$ time slots. From the above expression, the detection probability for a specific point covered by s sensor nodes, dp_s, is influenced by three factors.

1. T: The detection probability decreases with the increase of T. It is not apparent from the formula given above, but can be easily observed with numerical results. This is because a larger T will make the event span fewer time slots and hence decrease the detection probability.

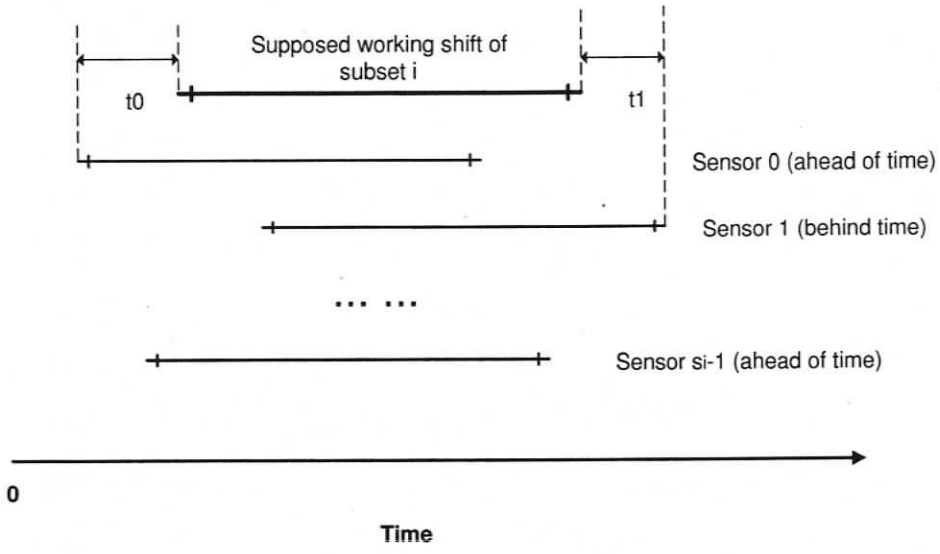

Figure 4.5. *A Point p Monitored by s_i Sensor Nodes in Subset i*

2. k: The detection probability decreases with the increase of k. This is because a large k value will increase the number of subsets that does not include a node to cover the point.
3. s: The detection probability increases with the increase of s. This is because a larger s value results in a smaller chance that a subset does not include any sensor to cover the point.

4.5.4 Analysis on the Impact of Clock Asynchrony on Coverage Quality

4.5.4.1 A Glance at Clock Asynchrony

Intuitively, the randomized scheduling algorithm should work well without requiring strict time synchronization. Let's check the example shown in Figure 4.5. A point p in the monitored region is covered by s_i sensor nodes in the subset i. Assume that among the s_i sensor nodes, some sensor nodes (e.g., sensor node 0) are ahead of the supposed starting

time while some (e.g., sensor node 1) are behind the time. In this example, point p can be monitored during the whole working shift of subset i even if the sensor nodes are not synchronized very well.

There are only three possibilities that point p may not be monitored during the working shift of subset i:

1. All the s_i sensor nodes are ahead of the starting time of subset i.
2. All the s_i sensor nodes are behind the starting time of subset i.
3. Some sensor nodes in S_i are ahead of the starting time of subset i while some in S_i are behind the time, and there is a gap period when no sensor node in S_i can monitor point p during the working shift of subset i.

4.5.4.2 Analysis on the Impact of Clock Asynchrony

To facilitate analysis, we make the following assumptions:

1. We assume that the internal time ticking frequency of each sensor node is accurate but may not be synchronized precisely to the standard time.
2. We assume that the clock drift of each sensor node from the standard time, Δt, is a random variable following a normal distribution with parameters $(0, \sigma)$.
3. If we use T to normalize Δt, we assume $\Delta t \geq \frac{T}{2}$ is an extremely rare case and could be ignored.

For a point p in the region, we suppose that there are s_i sensor nodes assigned to subset i $(1 \leq i \leq k)$ covering p. Let Δt_j denote the deviation of the clock of the j-th sensor node from the standard clock $(0 \leq j \leq s_i - 1)$. Δt_j is a random variable following a normal distribution with parameters $(0, \sigma)$. If $\Delta t_j \leq 0$ holds for all j $(0 \leq j \leq s_i - 1)$, indicating that all the clocks of these s_i sensor nodes are ahead of time, there will be a period of unmonitored time at the end of working duration of subset i with the length of $\min\{-\Delta t_j, 0 \leq j \leq s_i - 1\}$. Likewise, if $\Delta t_j \geq 0$ holds for all j $(0 \leq j \leq s_i - 1)$, indicating that all the clocks of these s_i sensor nodes are behind time, there will be a period

of unmonitored time at the beginning of working duration of subset i with the length of $\min\{\Delta t_j, 0 \leq j \leq s_i - 1\}$.

Note that the sensor nodes with an ahead-of-time clock in subset $i+1$ and the sensor nodes with a behind-time clock in subset $i-1$ could help decrease the unmonitored time length during the working duration of subset i. Nevertheless, considering these cases will greatly increase the analysis complexity by introducing correlation between neighboring subsets, we ignore these cases when calculating the network coverage intensity. Therefore, the calculated network coverage intensity is the lower bound of its actual value.

We now calculate the expectation of unmonitored-time fraction (the time when p is not covered by any of these s_i sensor nodes) during the working shift of subset i. We denote this expectation as E_{s_i}.

When $s_i = 0$, it is obvious that $E_0 = 1$. When $s_i \geq 0$,

$$E_{s_i} = \int_0^{\infty} x f_1(x)\, dx + \int_{-\infty}^{0} -y f_2(y)\, dy$$

where $x = \min\{\Delta t_j, 0 \leq j \leq s_i - 1\}$, $y = \max\{\Delta t_j, 0 \leq j \leq s_i - 1\}$, $f_1(x)$ and $f_2(y)$ are the p.d.f. (probability density function) of x and y, respectively.

Since $\Delta t_0, \Delta t_1, \cdots, \Delta t_{s_i - 1}$ are independent random variables following a normal distribution, we can get

$$Pr\{x \geq a\} = [1 - \Phi(a)]^{s_i}$$

where $\Phi(a)$ is c.d.f. (cumulative distribution function) of normal distribution. Therefore,

$$f_1(x) = s_i \phi(x) [1 - \Phi(x)]^{s_i - 1}$$

where $\phi(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}}$ and $\Phi(x) = \int_{-\infty}^{x} \phi(x)\, dx$.

By symmetry, we have

$$\int_{-\infty}^{0} -y f_2(y)\, dy = \int_0^{\infty} x f_1(x)\, dx$$

Therefore,

$$E_{s_i} = 2\int_0^{\infty} s_i x \left[1 - \Phi(x)\right]^{s_i - 1} \phi(x)\, dx.$$

Since x follows a normal distribution with parameters $(0, \sigma)$, when $x \geq 0$, we have $\Phi(x) \geq \frac{1}{2}$, and thus $1 - \Phi(x) \leq \frac{1}{2}$.

So we get

$$E_{s_i} \leq 2\int_0^{\infty} s_i x \left(\frac{1}{2}\right)^{s_i - 1} \phi(x)\, dx = \frac{s_i \sigma}{\sqrt{2\pi}} \left(\frac{1}{2}\right)^{s_i - 2}$$

Here, E_{s_i} is the expectation of the unmonitored time fraction during the working-shift of subset i, which includes exactly s_i sensor nodes covering the point p. Suppose that the total number of sensor nodes in the network that cover point p is s. Any subset may contain j sensor nodes to cover p, where j varies from 0 to s. Therefore, we can calculate the expectation of unmonitored time fraction for any subset (denoted as $\bar{E}_s$):

$$\begin{aligned}
\bar{E}_s &= \sum_{j=0}^{s} E_j \times Pr\{\textit{the subset contains } j \textit{ nodes to cover } p\} \\
&\leq 1 \times \left(1 - \frac{1}{k}\right)^s + \sum_{j=1}^{s} \frac{j\sigma}{\sqrt{2\pi}} \left(\frac{1}{2}\right)^{j-2} \left(1 - \frac{1}{k}\right)^{s-j} \left(\frac{1}{k}\right)^j \binom{s}{j} \\
&= \left(1 - \frac{1}{k}\right)^s + \frac{2s\sigma}{\sqrt{2\pi k}} \left(1 - \frac{1}{2k}\right)^{s-1}
\end{aligned}$$

Thus, for any point covered by s sensor nodes, the expectation of the monitored time fraction of the working-shift of any subset

$$E_s = 1 - \bar{E}_s \geq 1 - \left(1 - \frac{1}{k}\right)^s - \frac{2s\sigma}{\sqrt{2\pi k}} \left(1 - \frac{1}{2k}\right)^{s-1}$$

We next calculate E, the expectation of E_s.

$$
\begin{aligned}
E &= \sum_{s=0}^{n} E_s \times Pr\{\textit{the point is covered by } s \textit{ sensor nodes}\} \\
&= \sum_{s=0}^{n} E_s \binom{n}{s} q^s (1-q)^{n-s}, \ \textit{where } q = \frac{r}{a} \\
&\geq 1 - \sum_{s=0}^{n} \left(1 - \frac{1}{k}\right)^s \binom{n}{s} q^s (1-q)^{n-s} \\
&- \frac{2s\sigma}{\sqrt{2\pi}k} \sum_{s=0}^{n} \left(1 - \frac{1}{2k}\right)^{s-1} \binom{n}{s} q^s (1-q)^{n-s} \\
&= 1 - \left(1 - \frac{q}{k}\right)^n - \frac{2nq\sigma}{\sqrt{2\pi}k\,(1-q)} \left(1 - \frac{q}{2k}\right)^{n-1}
\end{aligned}
$$

For any point p, by symmetry, each subset has the same E value, so the expectation of monitored time fraction, which is the network coverage intensity C_n, according to the definition, can be calculated as

$$C_n = \frac{k \times E}{k} = E$$

Observing the expression of C_n above, we find that the term $1 - \left(1 - \frac{q}{k}\right)^n$ is equal to the C_n in Section 4.5.2, where all the clocks are assumed well-synchronized. Thus, the last term $\Delta = \frac{2nq\sigma}{\sqrt{2\pi}k(1-q)} \left(1 - \frac{q}{2k}\right)^{n-1}$ indicates the impact of time asynchrony on network coverage intensity. The numeric results in Figure 4.6 show the weight of Δ over C_n when $\sigma = 0.1613$ (which makes $(Pr(\Delta t \geq \frac{T}{2})$ is less than 0.1% and hence in the third assumption in the beginning of this subsection holds). The small values of the weight indicate that the impact of clock asynchrony on coverage is negligible and that the randomized scheduling scheme is resilient to time asynchrony.

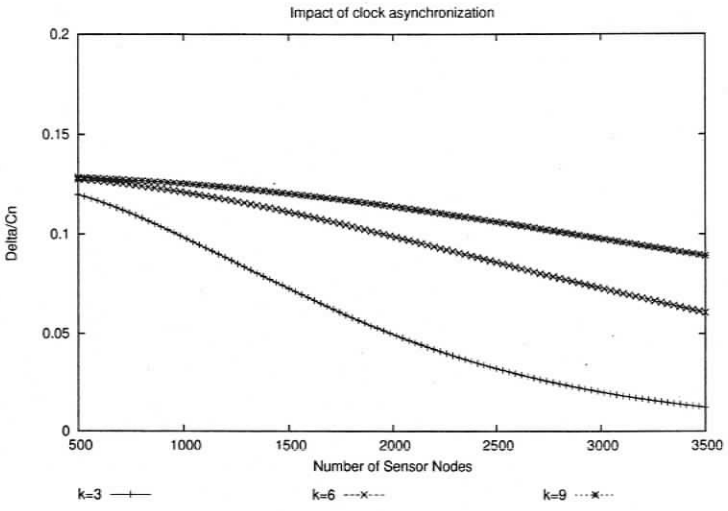

Figure 4.6. *Impact of Clock Asynchrony*

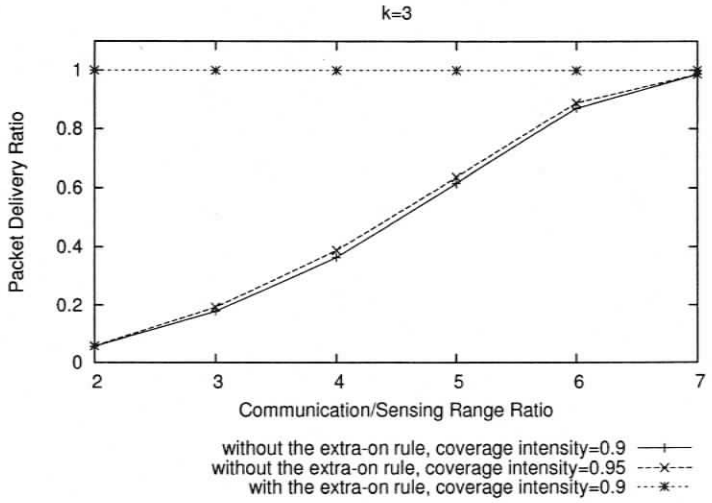

Figure 4.7. *The Packet Delivery Ratio*

4.6 Simulation Evaluation

4.6.1 Simulation Settings

We evaluate the performance of the joint scheduling algorithm on a simulator we implemented in Java. We use the CSMA MAC layer protocol. In our simulation, we deploy sensor nodes randomly in a 200 meters * 200 meters square region. The sink node is located at the center of the region. The total number of sensor nodes is selected to meet any given network coverage intensity, according to Corollary 1. The sensing range of each sensor node is fixed to 10 meters. We normalize the communication range with the sensing range and use the ratio of the communication range over the sensing range as the measure of the communication range. The traffic load is very light such that packet losses are mainly caused by network partition or channel errors. Under each simulation scenario, 100 runs with different random seeds are executed.

We use the following metrics to evaluate the joint scheduling algorithm:

Packet Delivery Ratio: It is defined as the ratio of total number of packets received at the sink node over the total number of transmitted packets from sensor nodes. Because the traffic load is very light, this metric is an indicator of network connectivity.

The Ratio of Nodes Having the Shortest Path: It is defined as the number of nodes that have the shortest path to the sink node over the total number of nodes. It is an indicator of path optimality.

The Ratio of Extra-on Sensor Nodes: It is defined as the ratio of the number of sensor nodes, which must remain active beyond their regular working shifts assigned by the randomized coverage-based scheduling algorithm, to the total number of deployed sensor nodes. A small ratio of extra-on sensor nodes indicates that small extra energy is required to maintain connectivity after the coverage requirement is granted.

Network Coverage Intensity: It is defined in Section 4.5.1 and is a measure of coverage quality.

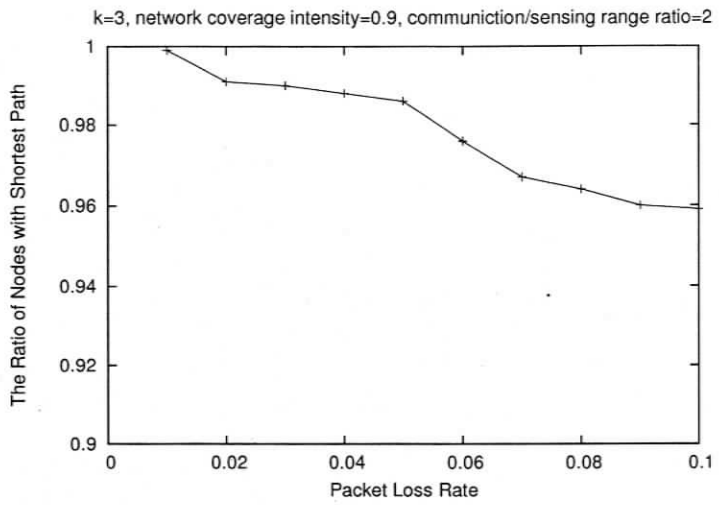

Figure 4.8. *The Ratio of Nodes Having the Shortest Path*

4.6.2 Network Connectivity and Path Optimality

To demonstrate that the extra-on rule can assure network connectivity, we compare the packet delivery ratio with and without the extra-on rule applied to the network. The number of deployed sensor nodes is determined by the network coverage intensity and k. In this test, to preclude the packet losses due to broadcast collision or channel errors, we adopt a perfect radio channel without medium contention. In all the simulated scenarios, the networks are connected if all the deployed sensor nodes are active. After randomly turning off redundant sensor nodes without applying the extra-on rule, the networks are partitioned. This is evident from the fact that the packet delivery ratio cannot achieve 100% without using the extra-on rule as shown in Figure 4.7. However, with the extra-on rule, the networks can always achieve a 100% packet delivery rate, indicating that the extra-on rule provides guaranteed network connectivity.

As discussed in Section 4.4.4, if there exists packet losses, some nodes may not have the shortest path to the sink node. Nevertheless, from Figure 4.8, we can see that at the end of Step 2 of the joint scheduling scheme, even if the packet loss rate is as high as 10%, the ratio of nodes having the shortest path to the sink node is no less than 95%.

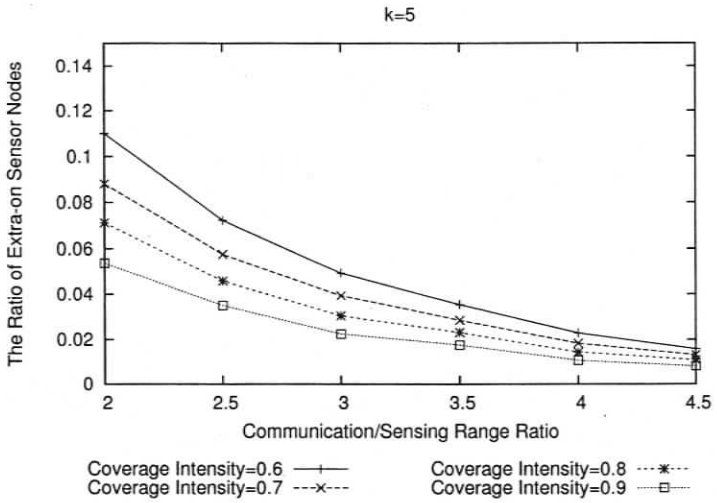

Figure 4.9. *Influential Factors of Average Number of Extra-On Nodes (K is fixed)*

4.6.3 Ratio of Extra-on Sensor Nodes

Apparently, there are three factors influencing the ratio of extra-on sensor nodes: network coverage intensity, the number of subsets, and the ratio of the communication range over the sensing range.

To investigate the influence of network coverage intensity, we fix the number of subsets and vary the communication range and the network coverage intensity. As shown in Figure 4.9, the ratio of extra-on sensor nodes drops with the increase of the coverage intensity and the communication range. This is because when the coverage intensity increases, more sensor nodes will remain active for coverage, hence few extra nodes are needed for network connectivity. In addition, the increase of communication range enhances the connectivity of the original networks, resulting in the decrease of the number of extra-on sensor nodes.

Similarly, to investigate the influence of the number of subsets, we fix the network coverage intensity and vary the communication range and the number of subsets. As shown in Figure 4.10, increasing communication range decreases the ratio of extra-on sensor nodes, due to the same reason mentioned above.

From Figure 4.10, we can see that the ratio of extra-on nodes decreases with the in-

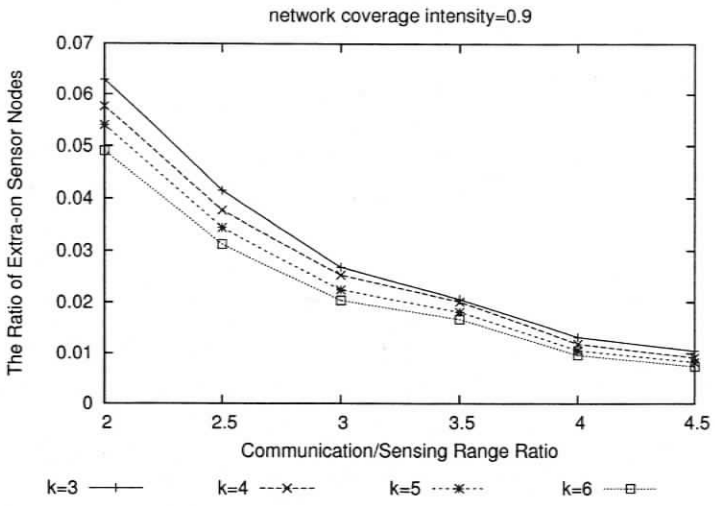

Figure 4.10. *Average Number of Extra-On Nodes (Coverage Intensity is fixed)*

crease of k. Given fixed coverage intensity, the number of simultaneous active sensor nodes scheduled by the randomized scheduling algorithm is roughly the same and is independent of the value k. That is, the density of the network after the randomized scheduling only depends on the coverage intensity. Since network connectivity is mainly determined by network density, the number of extra-on nodes should be roughly the same in order to maintain network connectivity. Since for a given coverage intensity a larger k value means a larger total number of deployed sensor nodes, the ratio of extra-on sensors decreases with the increase of k.

In Figure 4.9 and Figure 4.10, when the coverage intensity is sufficiently high and the communication range is sufficiently large, the number of extra nodes needed to turn on for connectivity maintenance is very small, compared to the total number of active nodes for coverage. Therefore, with our joint scheduling approach, the extra energy consumption on connectivity maintenance is small and the network coverage intensity does not unnecessarily exceed a given requirement too much.

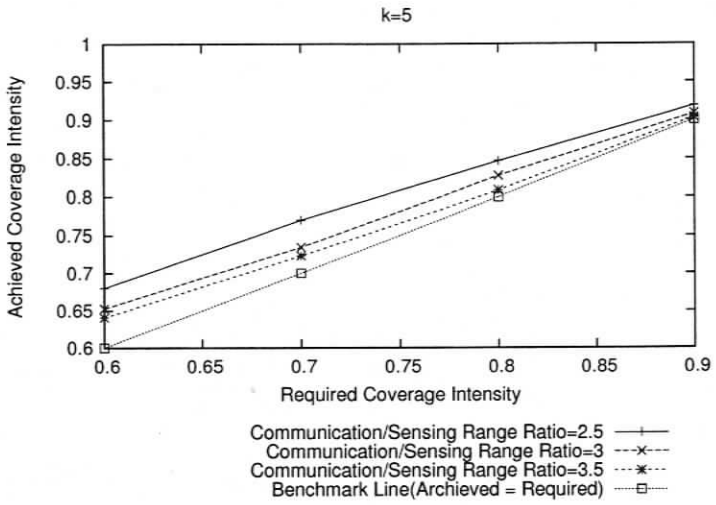

Figure 4.11. *Achieved Coverage Intensity Vs Required Coverage Intensity*

4.6.4 Network Coverage Intensity

The achieved network coverage intensity by our joint scheduling method is illustrated in Figure 4.11. As expected, the achieved coverage intensity is always slightly higher than the required coverage due to the fact that extra sensor nodes are needed to turn on to remain network connectivity.

4.7 Summary

Sensor scheduling plays an essential role for energy efficiency of wireless sensor networks. Traditional sensor scheduling methods usually use sensing coverage or network connectivity, but rarely both. Some research [51, 58, 66] has dealt with the joint problem of sensing coverage as well as network connectivity, but requires specific network topology such as grid or strong constraints on the relationship of sensing range and radio transmission range. In this thesis work, we take a different approach to solving the joint scheduling problem. We use a randomized scheduling method to provide statistical sensing coverage and then switch on extra sensors, if necessary, for network connectivity. Analytical and simulation

results demonstrate the effectiveness of our joint scheduling method. More specifically, our joint scheduling method has the following good features:

1. It can achieve substantial energy saving and meet both constraints of coverage and connectivity without relying on location information or any assumptions on the relationship between the sensing range and the radio range.
2. It is totally distributed and thus easy to implement.
3. Each active sensor node knows at least one route with minimum hop count to the sink node. In this sense, our joint scheduling method solves the routing problem and eliminates the system cost incurred by routing protocols.
4. The coverage intensity could be dynamically adjusted by a simple broadcast of the k value.
5. The scheduling method is resilient to time asynchrony.

Chapter 5

Adaptive Sampling Service

5.1 Introduction

An application domain where wireless sensor networks are broadly used is environmental data collection. Imagine that scientists want to collect information about the evolving process of a particular phenomenon over a given region. They can deploy a wireless sensor network in the region, consisting of a large number of geographically distributed sensor nodes. Each sensor node periodically samples certain measures characterizing the phenomenon, such as temperature and humidity, and transmits them back to the sink node. Each sampling value is associated with the sensor's location. In the sink node, a snapshot of the region is obtained by assembling the received sampling values in the corresponding locations. Snapshots may be periodically generated, one for each time instant at a given time granularity. These snapshot sequences constitute a SpatioTemporal Data Map (STDMap) [11]. Building up STDMap is actually the target of a lot of realistic applications, such as Geographic Information System (GIS).

STDMaps could be considered as analogous to video recording of the monitored region. The playback quality of a piece of video clip is determined by its time resolution (frame rate) and spatial resolution (the number of pixels). Similarly, the observation fidelity of STDMap is decided by its spatiotemporal resolution as well. For a given requirement on the observation fidelity, certain spatiotemporal resolution might suffice and hence it is unnecessary to ask all the sensor nodes to sample and report data as fast as they can.

Since sampling and wireless communication consume much energy, energy consumption can be significantly reduced by turning off redundant sensor nodes. Fortunately, due to the pervasive existence of spatiotemporal correlation in the sampling data, data transmission can be reduced aggressively for energy saving without a large degradation of observation fidelity.

Spatial correlation usually exists among the readings of neighboring sensor nodes, meaning that the measures from a sensor node can be predicted from that of its neighboring sensor nodes with high confidence. Therefore, given a certain requirement on spatial accuracy, only part of the sensor nodes should be required to work for sampling and data transmission, in order to save energy.

Unlike spatial correlation, temporal correlation exists in the time series from a single sensor node, meaning that the future readings of a sensor node can be predicted based on the previous readings from the same node. The correlation can be captured by mathematical models such as the linear model or the Auto Regressive Integrated Moving Average (ARIMA) model [16]. Therefore, the time series can be approximated by suitable mathematical models and the number of model parameters is usually significantly less than the length of the whole series. Transferring the model parameters, instead of the raw time series, can significantly decrease the energy consumption on communication.

This chapter aims to save energy in constructing STDMap by exploiting the spatiotemporal correlation in wireless sensor networks. To exploit the spatial correlation, we cluster the sensor nodes with similar observations into a cluster. Within each cluster, the reading of any sensor node can be approximated by any other sensor nodes within an error bound. Therefore, the sensor nodes within a cluster can be scheduled to work alternatively in order to save energy. We model this clustering problem as a clique covering problem and propose a greedy algorithm to solve it. Also, we propose a randomized scheduling algorithm to balance the energy consumption.

To exploit temporal correlation, we adopt piecewise linear approximation technique, i.e. approximating the time series with a sequence of line segments. In order to minimize

the energy consumption on data transmission with certain accuracy at the sink node, we model the problem as an optimization problem, named the PLAMLiS (Piecewise Linear Approximation with Minimum number of Line Segments) problem. The PLAMLiS problem can be solved in polynomial time by being converted to a shortest path problem [59]. However, the optimal solution requires excessive computing and memory resources, which might be prohibitive for sensor nodes. Therefore, we propose a lightweight, greedy algorithm to solve it. The greedy algorithm has the time complexity of $O(n^2 \log n)$, where n is the length of the time series, and can be easily integrated to our Energy Efficient Data Collection (EEDC) framework and work together with the dynamical clustering and randomized scheduling algorithms, leading to further energy saving.

Our EEDC framework is aware of the spatiotemporal correlation and can achieve *adaptive* sampling in both time and spatial domains. Adaptive sampling can dynamically adjust the temporal and spatial sampling rate aligning with the phenomenon changes while a given accuracy requirement is always satisfied. When a phenomenon varies slowly in time domain, the number of line segments required for approximation with certain accuracy decreases and hence less sampling data is transmitted to the sink node. Therefore, the actual temporal sampling rate decreases and over sampling is avoided during the slow-changing period. Similarly, when a phenomenon varies slowly in spatial domain and the observations are similar in a larger area, the size of each cluster increases and hence the total number of cluster decreases. Since the number of simultaneous working node is proportional to the number of the clusters, the actual spatial sampling rate decreases in this case.

The rest of the chapter is organized as follows: In Section 5.2, we propose the Energy Efficient Data Collection (EEDC) framework. In Section 5.3, we explore spatial correlation, and in Section 5.4 we propose a sensor scheduling scheme based on spatial correlation. In Section 5.5, we explore temporal correlation for further energy saving. In Section 5.6, we perform comprehensive performance evaluation of the EEDC framework based on both real test bed and a large synthetic dataset. Finally, we conclude this chapter in Section 5.7.

5.2 The Energy Efficient Data Collection (EEDC) Framework

In this chapter, we assume *a single-hop network model*, i.e., all the sensor nodes are within single-hop radio transmission to the sink node, or to a local center. With this setting, we do not need to consider network partitioning due to sensor scheduling. As can be seen, the sink node may become a bottleneck for large-scale networks. When a network becomes very large, the EEDC framework can be easily extended to a hierarchical architecture as shown in Fig. 5.1. A commonly used strategy is to divide the network into several sub-networks, with each depending on a local center for local data collection and long-distance radio transmission to the sink node [39]. In this case, the EEDC framework should be implemented in the local centers, and a local center could be considered as a local sink node. In the following, the sink node should be understood as a local center if the hierarchical network architecture is assumed.

Compared to other existing sensor scheduling schemes that rely on sensing coverage and the static geographic distribution of sensors [27, 54, 64], EEDC distinguishes itself in that it is data-aware and makes scheduling decisions according to the spatiotemporal correlation of phenomenon.

Compared to sensors nodes, the sink node usually has much larger memory and much powerful computing capability. Such an asymmetry between the sink node and the sensor nodes requires that a good design for data collection should not put heavy burdens on sensor nodes. Instead, the heavy duties should be assigned to the powerful sink node. Our EEDC framework follows this design principle and is shown in Fig. 5.2. As we can see, the functionalities in sensor nodes are much simpler than those in the sink node.

In a sensor node, the scheduler module simply extracts the working schedules received from the sink node and makes the sensor node work/sleep according to the schedule. In its working shift, it obtains the sensor readings periodically and puts the reading to the buffer. When the buffer is full, the PLAMLiS module takes the time-ordered readings in the buffer

as a time-series, approximates it with the PLAMLiS algorithm and sends the calculated line segments parameters to the output interface.

In contrast, the sink node takes most workloads, including four main functional modules as shown in Fig. 5.2.

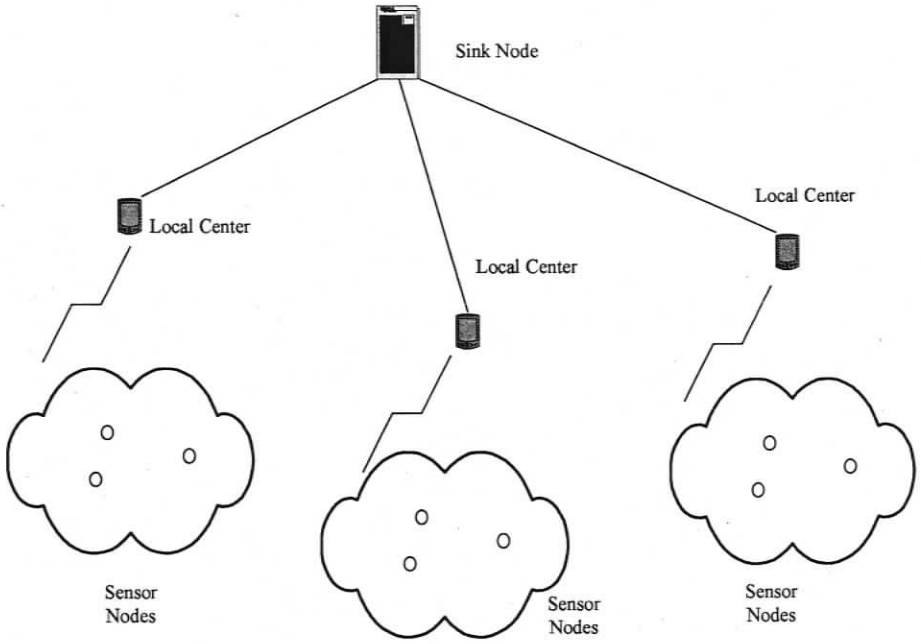

Figure 5.1. *Energy Efficient Data Collection (EEDC) Framework for Hierarchical Architecture*

1. *The data storage module*. It stores all sampling data received from the sensor nodes. This module records a time series for each sensor, which is fed into the dissimilarity measure module as input data.
2. *The dissimilarity measure module*. It calculates the pairwise dissimilarity measure of time series. Dissimilarity measure is application specific, and it is impossible to use a common dissimilarity measure to accommodate all application scenarios. As such, this module assumes that the dissimilarity measure is provided by the user for a specific application scenario. We will introduce the details of dissimilarity measure used in our experimental study in Section 5.3.2.

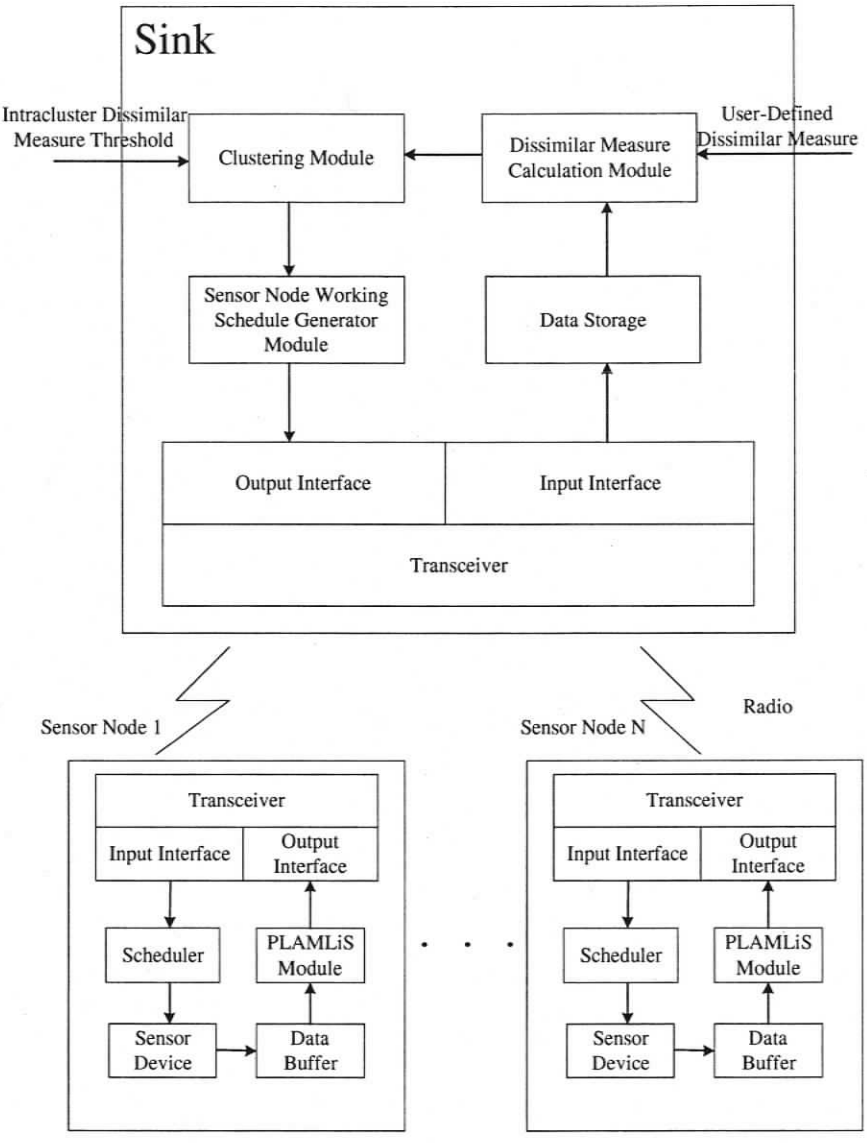

Figure 5.2. *Energy Efficient Data Collection (EEDC) Framework*

3. *The clustering module.* Given the dissimilarity computed by the dissimilarity measure module and a maximal dissimilarity threshold value max_dst, this module divides the sensor nodes into clusters, such that the dissimilarity of any two sensor nodes within a cluster is less than max_dst. The details of the clustering algorithm will be discussed in Section 5.3.3.
4. *The sensor node working schedule generator.* It generates a working schedule for each sensor node based on the clusters obtained from the clustering module. The details of sensor scheduling will be discussed in Section 5.4.

With the EEDC framework, the data collection procedure in a sensor network could be divided into the following three phases:

1. *Data accumulation.* In this phase, each sensor node keeps sampling, processes the samples with the PLAMLiS algorithm when the data buffer is full, then transmits the approximated time series to the sink node. The sink node records the received time series for each sensor node. After collecting enough data, the sink node calculates the dissimilarity measure between any two time series. It terminates this phase whenever the dissimilarity measure among most of the collected time series remains roughly stable. Note that in some applications the dissimilarity measure among some sensors may never become stable due to environmental noise or fast-changing phenomenon. In this situation, it may be very hard, if not impossible, to explore data correlation for energy saving for these nodes. In other words, the sensors that do not exhibit any similarity to other sensors cannot use spatial correlation for energy saving and will not be considered in the following clustering phase.
2. *Clustering.* In this phase, the sink node uses a clustering algorithm to partition sensor nodes according to the dissimilarity measure calculated in the first phase. The output of the clustering algorithm is a set of clusters, and inside each cluster the dissimilarity measure between two arbitrary sensor nodes is smaller than a given threshold value. Consequently, the whole field is divided into pieces of smooth sub-regions, each of

which is covered by a corresponding cluster of sensor nodes. The observation at any point in this sub-region can be approximated by the observation of any sensor node within the cluster covering this sub-region.

3. *Saving and dynamic clustering*. In this phase, the sink node sends out the decision of clusters to all sensor nodes and requires the sensor nodes within the same cluster to work alternatively to save energy. In the mean time, the sink node monitors large variations within a cluster and dynamically adjusts the cluster.

Note that after clustering, the environment may change and thus the previous clusters may not be valid anymore. It is desirable to adaptively change the clusters in response to the changes of the spatial correlation. The details of re-clustering will be discussed in Section 5.4 when we introduce the scheduling algorithm.

In terms of data restoration quality, if the approximation error is bounded by ϵ_1 in the PLAMLiS algorithm and by ϵ_2 in the clustering algorithm, the final approximation error of each reading at any point in the restored STDMap is upper bounded by $\epsilon_1 + \epsilon_2$. Our experimental results show that the approximation error is actually far below this upper bound in the specific experimental scenario. The details will be disclosed in Section 5.6.7. Note that the order of performing PLAMLiS algorithm and clustering algorithm is interchangeable since we exploit spatial correlation and temporal correlation orthogonally and the final approximation error bound is the same regardless of the order. However, executing PLAMLiS first can save energy in the data accumulation phase.

5.3 Exploiting Spatial Correlation

5.3.1 Motivation and Methodology

Most existing research [27, 54, 64] adopts the *coverage-based scheduling methods* to save energy. That is, they check the redundancy based on the sensing coverage of sensor nodes. In this case, the pairwise geographic distance between sensor nodes is the only factor con-

sidered in sensor scheduling, and the output of the scheduling algorithm is a scheduling scheme that samples the field uniformly in space, given that the sensing range of individual sensor node is equal network wide. The coverage-based scheduling methods may work well for event-driven applications, such as the detection of fire, in which sensor nodes remain silent most of the time and send alarms only when the critical event (fire) is detected.

Unlike event-driven applications, some applications require continuous data collection to construct the STDMap by restoring the spatiotemporal evolving process of phenomenon in the field for further online or offline analysis. The restoration is based on a sequence of snapshots, each of which consists of the local measures from geographically distributed sensor nodes. Therefore, in such applications, sensor nodes must report their local measures of interest, such as temperature, light intensity, air pressure, etc., to the sink node *continuously*. Also, the density of the measure points in a field should be sufficiently high so that the spatial distribution of the monitored measure can be restored with high quality.

We stress that the coverage-based scheduling methods may not work well for applications requiring continuous data collection. First, the environment might be heterogeneous, meaning that the monitored measure changes quickly in some sub-regions but slowly in others. It is also possible that the readings of two geographically proximate sensor nodes are dramatically different due to a boundary that separates the two sensor nodes into two sub-regions with distinguished features. For example, if the temperature is monitored, two proximate sensor nodes can report very different values if one is directly under sunshine and the other one is in the shadow. An even spatial sampling rate resulted from a coverage-based scheduling method is obviously not appropriate in this case, where adaptive sampling rate is required according to the environmental heterogeneity. Second, it is often very difficult to accurately estimate the sensing range of a sensor node, since the sensing range is highly related to the intensity of event source and the physical propagation features of the local environment in which the sensor node is deployed.

As a result, scheduling purely based on the sensing coverage may not be efficient or effective for continuous data-collection applications. The inherent problem of the coverage-

based scheduling methods is that *they only rely on the static structure of the sensor networks but are unaware of the data reported by the sensor nodes.* Intuitively, the correlation between the data reported by the sensor nodes may help to reduce the spatial sampling rate of the sensor nodes substantially.

Example: The readings reported by a sensor node over time form a time series. Suppose the time series of sensor nodes x, y, and z are very similar in the past. Thus, we may conjecture that the readings of x, y, and z would also likely be similar in the future. Therefore, instead of scheduling the three sensor nodes reporting data simultaneously, we can let two out of the three sensor nodes report at a time and all the three take turn to report. Such a schedule has the following two advantages.

- *Energy saving.* Each sensor node saves 33.3% energy on reporting data.
- *Quality guarantee.* When the three sensors are still correlated, we can obtain the data with high quality and also save energy. On the other hand, even if the readings of one sensor node become not similar to the other two, we still can detect the divergence with a minor delay. Then, an updated schedule can be made based on the change. ■

Motivated by the intuition in the example above, we develop *a dynamic clustering and scheduling approach to solve the typical data collection problem in continuous-collection applications with wireless sensor networks.*

Our solution is to dynamically group sensor nodes into a set of disjoint clusters such that the sensor nodes within a single cluster have current and strong spatial correlation, hence great similarity in observations. Therefore, all the sensor nodes in a cluster can be treated equally, and at any time instant only a small fraction of sensor nodes are needed to be active, serving as the representatives for the whole cluster. All the rest of sensor nodes can sleep without much degradation of observation fidelity. To balance the energy consumption, the sensor nodes within a cluster can share the workload equally.

The clustering operation is based on the dissimilarity measure of time series consisting of historical observations from individual sensor nodes. The degree of spatial correlation

can be evaluated by the dissimilarity measure. For two locations with high spatial correlation, their corresponding time series are usually associated with a low dissimilarity measure. Therefore, in a very smooth sub-region, the observed measure has only small changes within the sub-region, that is, the difference between observations at any two locations within the sub-region may be quite small. Hence, the working sensor nodes within this sub-region could be sparse without losing the observation fidelity. In contrast, in a fast-changing sub-region, the working sensor nodes should be dense. By setting an appropriate dissimilarity measure threshold value to distinguish similar nodes from dissimilar nodes, the spatial sampling rate will match the spatial variation of the observed physical phenomenon. A smaller threshold value increases the spatial sampling rate and the observation fidelity, but it requires more energy consumption. In this sense, the dissimilarity measure threshold value constitutes a degree of freedom that could be tuned to balance the trade-off between observation fidelity and energy consumption.

Another advantage of our method is that *the difficulty of estimating a sensor's sensing range is avoided* because the dissimilarity measure relies solely on the data reported.

5.3.2 Dissimilarity Measure

As described above, the time-ordered data sequence at each sensor node forms a time series, and the clustering algorithm is based on the dissimilarity between the time series of the sensor nodes. Nevertheless, the notion of dissimilarity of complex objects such as time-series is application specific and task oriented, and defining dissimilarity is non-trivial [28]. However, in the context of continuous-collection applications with wireless sensor networks, we believe that the magnitude and the trend of time series are of most concern, since they determine the general shape of the phenomenon's evolving curve. Therefore, we define the following two metrics to evaluate the difference of two time series.

Definition 1: magnitude m-similar. Two time series $X\{x_1, x_2, ..., x_q\}$ and $Y\{y_1, y_2, ..., y_q\}$ are magnitude m-similar if

$$|x_i - y_i| \leq m, 1 \leq i \leq q$$

Definition 2: trend t-similar. Two time series $X\{x_1, x_2, ..., x_q\}$ and $Y\{y_1, y_2, ..., y_q\}$ are trend t-similar if

$$\frac{q_1}{q} \geq t,$$

where q_1 is the total number of pairs (x_i,y_i) in the time series that satisfy $\nabla x_i \times \nabla y_i \geq 0, \nabla x_i = x_i - x_{i-1}, \nabla y_i = y_i - y_{i-1}, 1 \leq i \leq q-1$.

If we put geographical constraints on the similarity of two sensor nodes, for instance, any two sensor nodes whose geographical distance is larger than a threshold value are considered to be dissimilar, the computational complexity of calculating pairwise dissimilarity can be greatly reduced. This geographical constraint is realistic, because generally the observation correlation of two location decreases with the increase of the distance between locations [29]. Thus, we constrain that the geographic distance between any two sensor nodes that are considered similar must be at most $gmax\_dist$, where $gmax\_dist$ is a given maximal distance threshold.

In general, we want the dissimilarity measure to have a straightforward and practical meaning. Assume that $gdst(S_x, S_y)$ denotes the geographical distance between sensor node S_x and sensor node S_y, and assume that $gmax\_dst$ is the user-defined threshold value for geographical distance. In our later experiments, we put two time series X and Y into different groups if (a) they are not magnitude m-similar, or (b) they are not trend t-similar, or (c) $gdst(S_x, S_y)$ is greater than $gmax\_dst$.

5.3.3 Clustering Sensor Nodes

Given the pairwise dissimilarity values, we need a clustering algorithm to partition the sensor nodes into exclusive clusters such that within each cluster, the pairwise dissimilarity measure of the sensor nodes is below a given intra-cluster dissimilarity threshold max_dst, which is defined by the tuple $(m, t, gmax\_dst)$ in our example. All sensor nodes in the same cluster are correlated. In each cluster, only as few as one sensor node needs to work at any time instant. Because of this reason, it is desirable to minimize the number of clusters

to maximize the energy saving.

Interestingly, the above problem could be modeled as a clique-covering problem. We construct a graph G such that each sensor node is a vertex in the graph. An edge (u, v) is drawn if the dissimilarity measure between vertex u and vertex v is less than or equal to the given intra-cluster dissimilarity measure threshold max_dst. Clearly, a cluster is a clique in the graph. Then, the clustering problem is to use the minimum number of cliques to cover all vertices in the graph.

The clique-covering problem is proven to be NP-complete and even does not allow constant approximation [37, 68]. Hence we adopt a greedy algorithm as described in Fig. 5.3 to obtain a rough approximation. The basic idea of the algorithm is to heuristically find cliques that cover more vertices that have not been clustered. Heuristically, the vertices with larger degrees may have a better chance of appearing in larger cliques. Thus, the search starts from the vertex with the largest degree, until all vertices are covered. The output of this algorithm is a set of cliques that cover all vertices.

5.4 Sensor Scheduling Based on Spatial Correlation

5.4.1 Randomized Intracluster Scheduling Method

Unlike the round-robin scheduling method which maintains a single active sensor node within each cluster [34], we propose using multiple active sensor nodes per cluster at an instant for the following considerations. First, multiple active sensor nodes can improve data reliability. When multiple active sensor nodes per cluster are used, the data can be restored as long as at least one packet out of the multiple active sensor nodes reaches the sink node. Second, it is possible that a cluster needs to be split to two or more clusters due to sudden spatial correlation changes caused by the environmental changes. Multiple active sensor nodes can help to shorten the delay of cluster split detection and make the system quickly respond to the spatial correlation changes among sensor nodes within a

```
Input: a graph G;
Output: a set of cliques covering the graph G;
Algorithm Description:
  Label all vertices in the graph G as uncovered;
  while (there are vertices uncovered in the graph G){
      Pick up the vertex v with the highest node degree among the uncovered vertices;
      Pick up all the vertices adjacent to v and put them into a list of S;
      Construct a graph Gtmp, consisting of only the vertices in S;
      Calculate the node degree of each vertices in Gtmp;
      Sort the vertices in S according to the decreasing order of node degree in Gtmp;
      (To break a tie, the vertex with lower degree in the original graph G precedes);
      Construct a clique C containing only v;
      while (there are vertices available in S){
            Pick up next vertex s from S;
            If s is adjacent to all vertices in C thus far, put s into the clique C;
      }
      Output clique C;
      Remove all vertices covered by C from the graph G;
  }
```

Figure 5.3. *The Greedy Clique-Covering Algorithm*

single cluster.

Randomized Intracluster Scheduling Method: The time is divided into a sequence of time slots each with length of T. T is the time needed to collect q samples (This is because we need q samples to evaluate the similarity between two sensor nodes). For each time slot, the sensor node goes into work state with a probability λ. Note that λ varies among clusters. A large cluster should have a small λ and a small cluster should have a large λ, in order to obtain same detection delay for cluster split. Note that the detection delay is the interval between the time instant when an event occurs and the time instant when it is detected by an active sensor node.

Randomization balances the workload distribution among all sensor nodes without introducing any communication overhead for coordination. Besides, by adjusting the value of λ, we can easily control the trade off between energy saving and detection delay for cluster split.

The sink node calculates the pairwise dissimilarity of active sensor nodes within each cluster at the end of each time slot, after it successfully collects the latest q samples from each active sensor node. The current cluster should be split if the sink node finds that there is at least one active sensor node reporting significantly different data.

5.4.2 Analysis on Detection Delay for Cluster Split

Theorem 1: With the randomized intracluster scheduling method, the expectation of the detection delay for cluster split, T_d, is no greater than $T\left(\frac{1}{\lambda[1-(1-\lambda)^{n-1}]}-\frac{1}{2}\right)$, where n is the size of the cluster.

Proof: We only consider the case that the environment changes make current cluster split into two clusters. If we can detect the case that the current cluster must be split into two clusters, we can certainly detect the case of splitting the cluster into more clusters, since the latter case is simply the recurrence of the former case.

Suppose n sensor nodes are split into two clusters C_1 and C_2 of size n_1 and n_2 ($n_1 + n_2 = n$), respectively. Without losing generality, we assume that the split occurs at slot 1.

Therefore,

$$
\begin{aligned}
&Pr\{\textit{split is detected at slot } 1\} \\
&= Pr\{\textit{at least one node is active in } C_1\} * Pr\{\textit{at least one node is active in } C_2\} \\
&= [1-(1-\lambda)^{n_1}] \times [1-(1-\lambda)^{n_2}] = p
\end{aligned}
$$

$$
\begin{aligned}
&Pr\{\textit{split is detected at slot } 2\} \\
&= Pr\{\textit{split is not detected at slot } 1\} * Pr\{\textit{splitting is detected at slot } 2\} \\
&= (1-p) \times p
\end{aligned}
$$

Similarly,

$$Pr\{\textit{split is detected at slot } i\} = (1-p)^{i-1} \times p$$

So,

$$T_d = \sum_{i=1}^{\infty} \int_{(i-1)\times T}^{iT} \frac{t \times Pr\{\textit{splitting is detected at slot } i\}}{T} dt = T\left(\frac{1}{p} - \frac{1}{2}\right)$$

T_d reaches its upper bound when p is minimum. Conditioned on $n_1 > 0, n_2 > 0$, and $0 < \lambda \leq 1$, p is minimum when n_1 or n_2 equals 1. That is, it is hardest to detect when only one sensor node deviates from the rest of sensor nodes in a cluster.

Therefore,

$$T_d \leq T\left(\frac{1}{\lambda\left[1-(1-\lambda)^{n-1}\right]} - \frac{1}{2}\right)$$

■

Since the actual working time of each sensor node is proportional to λ, based on the Theorem 1, we can set λ to the minimum value such that the specified split-detection delay can be met while the actual working time of each sensor node is minimized.

Note that with the randomized intracluster scheduling scheme proposed above, it is possible that in a particular time slot, all the sensor nodes in a cluster happen to sleep,

and hence the sink node can not restore the readings in this time slot. A solution to this problem is to use the round-robin scheduling method in our previous work [34], along with the randomized intracluster scheduling method. The round-robin scheduling method evenly distributes the workload among the nodes in a cluster and guarantees there is at least one active node in any time slot. In the working time slot assigned by the round-robin scheduling, the sensor node must go into work state. For the rest of time slots, the sensor node randomly goes into the sleep/work state with the probability of λ. In this case, the number of working nodes in any time slot is no less than that in the purely randomized intracluster scheduling. Therefore, the upper bound of the expectation of the detection delay for cluster split still holds for the joint scheduling scheme.

5.4.3 Energy Saving

With joint round-robin and randomized scheduling, it is obviously that, after the formation of clusters, for any sensor node within a cluster of size n, the average amount of data it reports to the sink node is only a fraction $\frac{\lambda(n-1)+1}{n}$ of the amount of data without scheduling.

Note that the amount of energy saving varies with the size of the cluster. For sensor nodes within a larger cluster, a longer lifetime can be expected due to a lower workload. For sensor nodes within a smaller cluster, their workload is higher and thus the lifetime is shorter, due to the lower data redundancy in vicinity. In other words, they are critical nodes, and the lack of their observations results in a large information loss. In this case, there is no room for scheduling algorithm to exploit spatial correlation for energy saving. The only solution to prolong the lifetime of these critical nodes is to deploy more sensor nodes in their vicinity to share their workload.

5.4.4 Dynamic Adjustment

Once the sink node detects that a cluster should be split, it asks all sensor nodes in the corresponding cluster to work simultaneously. Then the clustering algorithm will be executed

to re-group these sensor nodes into several clusters in response to local spatial correlation changes. It is obvious that the number of clusters will keep increasing, since there are only splitting operations in the above adjustment. In the worst case, most sensors in the network will be waken up to work simultaneously. To avoid this situation, the sink node can re-cluster the whole network when the current number of clusters becomes significantly larger than the number of clusters at the previous network-wide clustering.

When the spatial correlation within a sub-region remains stable, the frequency of dynamic adjustment of clusters should be very low. We stress that *spatial correlation is quite stable for many applications even if the monitored phenomenon changes dramatically*. For instance, in the experimental results presented in Section 5.6.2, we changed light strength very quickly by tuning the dimmer of a desk lamp. The sampling data from the sensors within the same box remained similar no matter how fast we changed the light.

5.4.5 Data Restoration at the Sink

Based on the joint round-robin scheduling and randomized scheduling, for a cluster of size n, the number of working sensor nodes in any time slot varies between 1 and n with an expectation of $1 + (n-1) \times \lambda$. The sink node uses the following restoration rule to restore the observation of sleeping nodes.

Restoration Rule: For any sleeping node, the observation at instant i, denoted as v_i, is restored as $\frac{min_i + max_i}{2}$, where max_i and min_i denote the maximum and minimum observation of the working sensor nodes within the same cluster at instant i, respectively.

Theorem 2: If the cluster is not split, with the restoration rule given above, the upper bound of the restoration error at instant i is $m - \frac{max_i - min_i}{2}$, where m is the similarity threshold value.

Proof: If the cluster is not split, the difference of two observations from any pair of sensor nodes, no matter they are working or sleeping, should be less than m. Therefore, the upper bound and lower bound of observation from sleeping nodes at instant i is $min_i + m$ and $max_i - m$, respectively. The length of this feasible region for the observation of

sleeping node is $min_i + m - (max_i - m) = 2 \times m + min_i - max_i$. Since we take the middle point of this feasible region as the restoration value, the difference between the actual value and restoration value is no more than the half length of the feasible region, that is, $\frac{2 \times m + min_i - max_i}{2} = m - \frac{max_i - min_i}{2}$. ■

Note that another way to restore the data is to use the average of all working nodes, but in the worst case the error bound could be as large as m. We ignore the proof for brevity.

5.5 Exploiting Temporal Correlation

5.5.1 Motivation and Methodology

Typically, each sensor node samples the environment at a fixed interval and the time-ordered sequence of sampling constitutes a time series. However, transmitting all data from each sensor node back to the sink node is often prohibitive due to limited bandwidth and heavy energy consumption on data transmission. In order to reduce the volume of transmitted data from each individual sensor node, we approximate the observed time series within a given error bound. Our technique builds on the fact that for most time series consisting of environmental measures such as temperature and humidity, linear approximation is simple and works well enough in a small time period. Therefore, we use the piecewise linear approximation technique to approximate the time series at each individual sensor node. If computational capacity permits, it is possible to adopt other approximation and compression techniques, such as wavelet transformation. We did not explore complex approximation techniques in this thesis work since it is still unrealistic to perform complex calculation with current sensor node hardware.

The sensor node maintains a fixed size of buffer to temporally store the latest sampling values. Once the buffer is full, we calculate the line segments approximating the original time series. Then only the end points from every line segments, rather than the whole time series, will be transmitted to the sink node. Usually, each line segment represents more

than two measures, so we can expect a significant decrease of data volume.

5.5.2 Problem Modeling

Piecewise linear approximation is a well-investigated research area [15, 31, 30, 44, 51, 58]. The idea is to use a sequence of line segments to represent the time series with a bounded approximation error. Since a line segment can be determined by only two end points, piecewise linear approximation can make the storage, transmission and computation of time series more efficient.

Most of the existing piecewise linear approximation algorithms use standard linear regression technique to calculate a line segment with the minimal total squared error between line segment and the represented points. Unlike these methods, we use a simpler method to reduce the computation complexity. In our method, the end points of each line segment must be the points in the time series and hence the linear fitting procedure is not necessary. For the observations that are not the end points of the selected line segments in the time series, their values are approximated by the corresponding points in the line segments. To facilitate understanding, Fig. 5.4 shows a possible scenario for the piecewise linear approximation, where three line segments(4 values), (x_1, x_2), (x_2, x_6), and (x_6, x_7) could be used to approximate the time series.

The problem: Given a time series and an error bound ϵ, how to choose the minimum number of line segments to approximate the time series, such that the difference between any approximation value and its actual value is less than ϵ. The end points of the line segments must be the points in the time series.

5.5.3 The PLAMLiS Algorithm

In [35], it is disclosed that the PLAMLiS problem can be solved in polynomial time. Assume that the time series consists of n points $\{x_1, x_2, ..., x_n\}$. The basic idea is to build a graph with the data points as the vertices. An edge is established between x_i and $x_j (i < j)$

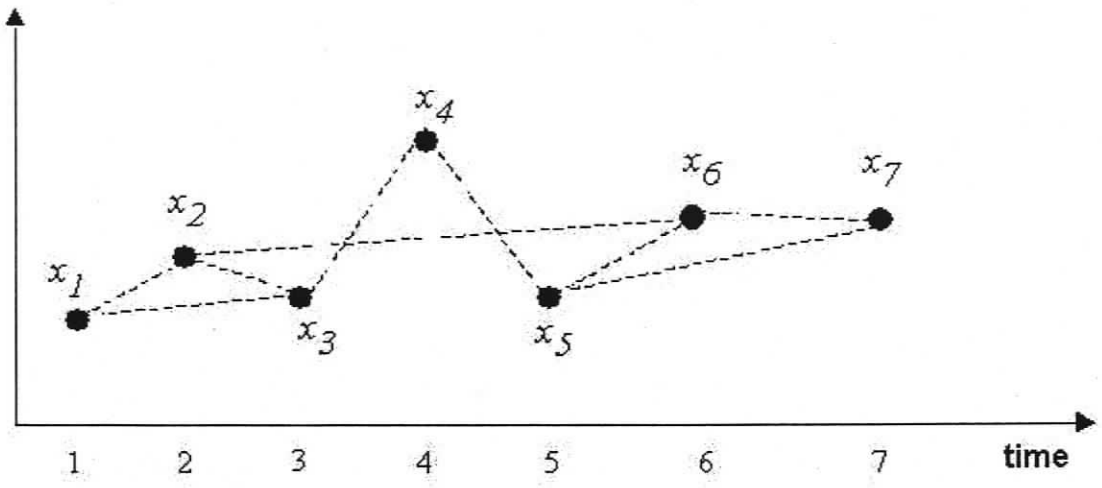

Figure 5.4. *An Example of Piecewise Linear Approximation*

if the line segment (x_i, x_j) meets the error bound, that is, the difference between the approximation value of $x_k (i < k < j)$ and the actual value of x_k is not larger than the given error bound. Note that the approximation value of x_k is the intersection point of the line segment (x_i, x_j) and the vertical line $x = x_k$. The PLAMLiS problem is solved by searching the shortest path from x_1 to x_n in the above graph. Fig. 5.4 shows the graph and the shortest path (x_1, x_2, x_6, x_7) that constitutes the optimal solution.

In the worst case, the above algorithm has the time complexity of $O(n^3)$ and requires space cost of $O(n^2)$. The calculation and storage complexity of the solution might be too high for the resource stringent sensor nodes. Therefore, we propose a greedy algorithm to solve it. If the time series consists of n points, the greedy PLAMLiS algorithm can run in $O(n^2 \log n)$ time and takes $O(n)$ space.

The greedy algorithm converts the PLAMLiS problem into a set covering problem as follows. Suppose that there is a time series X, consisting of n points $\{x_1, x_2, ..., x_n\}$. For each point x_i in the time series, associate it with point $x_j (j > i)$, which is furthest away from point x_i and the line segment (x_i, x_j) meets the given error bound, that is, the difference between the approximation value of $x_k (i < k < j)$ and the actual value of $x_k (i < k < j)$ is not larger than the given error bound. Let F_i denote the subset consisting

of all the points on this line segment $\{x_i, x_{i+1}, ..., x_j\}$. Eventually, we have a set F with n subsets ($\{F_1, F_2, ..., F_n\}$). The running time to construct the set F is $O(n^2 logn)$ and the memory needed is $O(n)$.

Now, the PLAMLiS problem is converted to the problem of picking up the least number of subsets from F which covers all the elements in set X. This is the minimum set cover problem, which is proved NP-complete. We use the following well-known greedy algorithm to solve this problem. The basic idea is that at each stage, pick the set that covers the largest number of remaining elements that are uncovered. The algorithm ends whenever all the elements are covered. This greedy algorithm can run in time $O(\sum_{S \in F} |S|)$ with a P(n)-approximation rate, where $P(n) = H(max|S| : S \in F)$ ($H(d) = \sum_{i=1}^{d} \frac{1}{i}$, the d-th harmonic number). The space complexity is $O(n)$. In the PLAMLiS problem, since $|F| = n$ and $|S| \leq n$ ($S \in F$), the running time is bounded by $O(n^2)$.

In the whole, the greedy PLAMLiS algorithm can run in $O(n^2 \log n)$ time and takes $O(n)$ memory space.

5.6 Comprehensive Performance Evaluation

5.6.1 Evaluation Methodology

In the following sections, we first evaluate the performance of EEDC when only spatial correlation is exploited. The evaluation is based on both a real test bed as well as a large-scale synthetic dataset. We then evaluate the performance of the PLAMLiS algorithm in exploring temporal correlation. Finally, we investigate the performance when spatial correlation and temporal correlation are jointly considered to further reduce energy consumption.

5.6.2 Experiment Setup of Testing Spatial Correlation

We experimentally tested the EEDC framework based on MICA2 sensor nodes [1]. As illustrated in Fig. 5.5, we deployed 18 MICA2 sensor nodes in a 3 × 6 grid layout on a

large table to sample the light intensity. The unit length of each grid is 1 foot. A desk lamp with a dimmer was the only light source in the room. We used several boxes with different sizes of holes on the top to divide the area into sub-regions with different light intensity. The deployment of sensor nodes and the boxes are illustrated in Fig. 5.6. The monitored phenomenon was generated by varying the light intensity of the lamp. The onboard ADC translates a light intensity raw reading into an integer value between 0 and 1024. We implemented the EEDC framework on the sink node and the MICA2 sensor nodes to collect the light intensity data. The sampling rate was 2 samples per second at the working sensors, and the data collection time was 10 minutes.

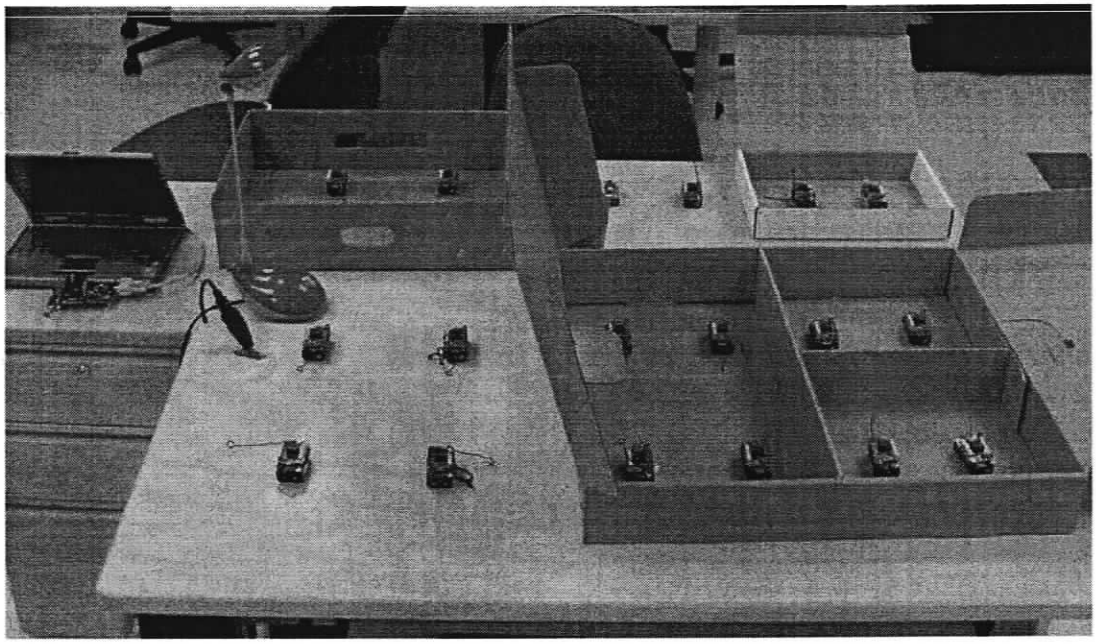

Figure 5.5. *The Test Bed*

The purpose of this experiment is twofold. First, we need to verify the correctness of the proposed clustering algorithm. Second, since the energy saving should not be achieved at the cost of observation fidelity, we need to verify that the observation fidelity with EEDC is acceptable.

Although this experiment is not a real application, we remark that the experimental

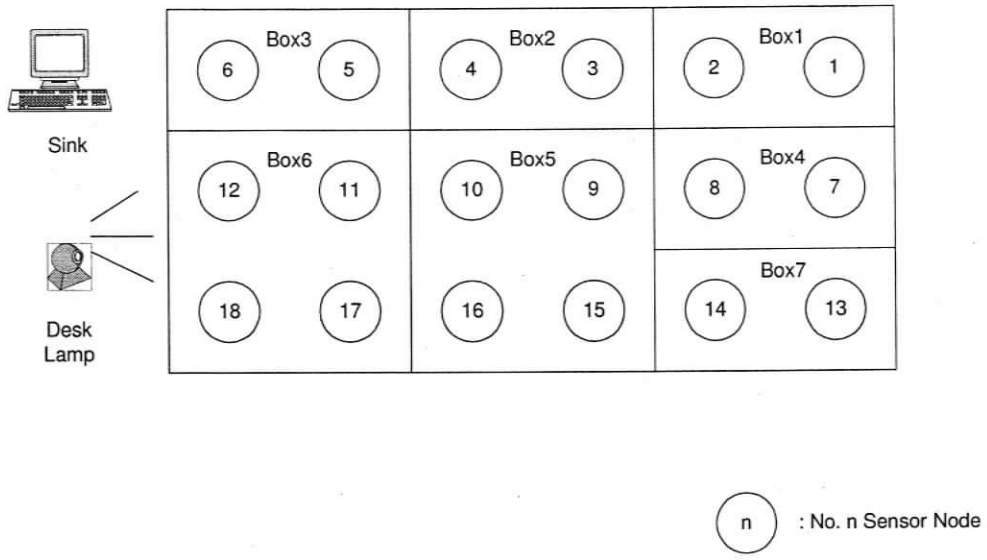

Figure 5.6. *The Sensor Nodes and the Boxes*

design may be representative for some real applications with sensor networks, for instance, the monitoring system for storage rooms in a grocery warehouse.

5.6.3 Experimental Results of Exploring Spatial Correlation

5.6.3.1 The Correctness of Clustering with EEDC

Although the clique-covering problem for a general graph is NP-complete, it is easy to know the optimal clique covers in our experiment since the knowledge of which sensor node belongs to which sub-region is known as a priori. We use the criteria in Section 5.3.2 to check the dissimilarity and set $m = 30, t = 95\%, gmax\_dst = 3$ feet. By calculating the pairwise dissimilarity measure, we obtain a graph as shown in Fig. 5.7, where a link between two nodes indicates that they are similar according to the above criteria. From this figure we can see that all cliques consist of the sensor nodes in the same box, which validates the effectiveness of the dissimilarity measure. The output of the clustering algorithm

is illustrated in Fig. 5.8, which is apparently the optimal solution for this specific simple graph.

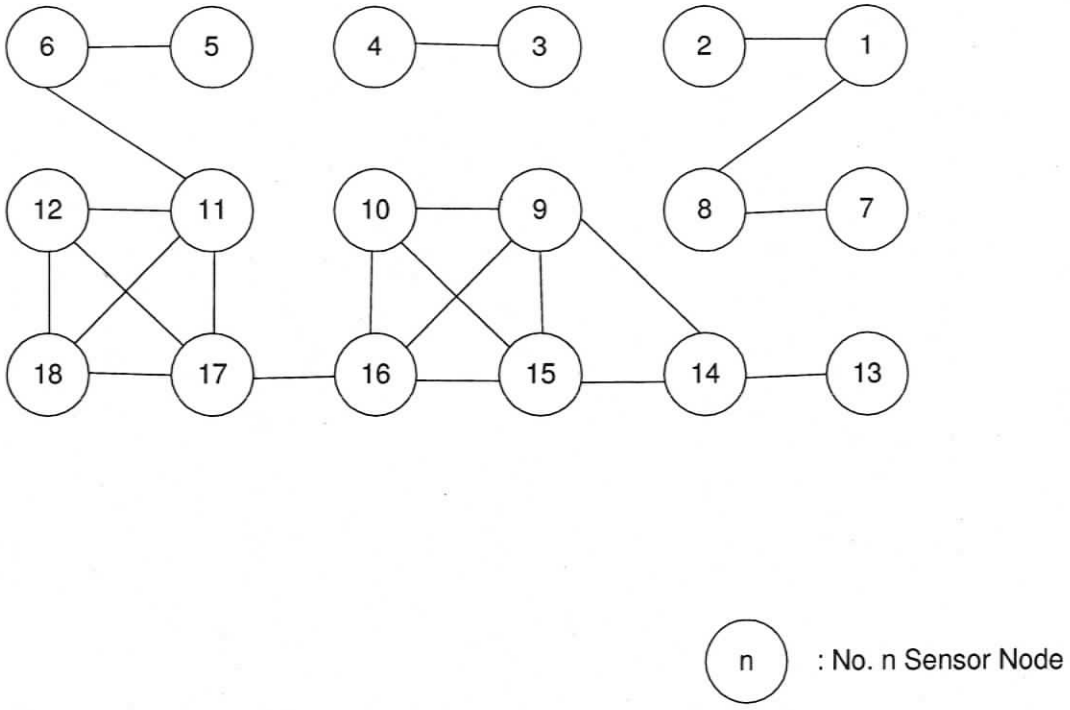

Figure 5.7. *The Generated Graph*

5.6.3.2 The Observation Fidelity with EEDC

The difference distortion measure has been broadly used in image compression to evaluate the fidelity of a re-constructed image against the original image [47]. The difference distortion measure σ^2 is given by

$$\sigma^2 = \frac{\sum_{j=1}^{M} \sum_{i=1}^{N} (X_{ij} - Y_{ij})^2}{M \times N}$$

where X_{ij} is the j-th actual sampling value from the i-th sensor node, Y_{ij} is the j-th restoration value of the i-th sensor node at the sink, N is the total number of sensor nodes, and M is the total number of samples from each sensor node.

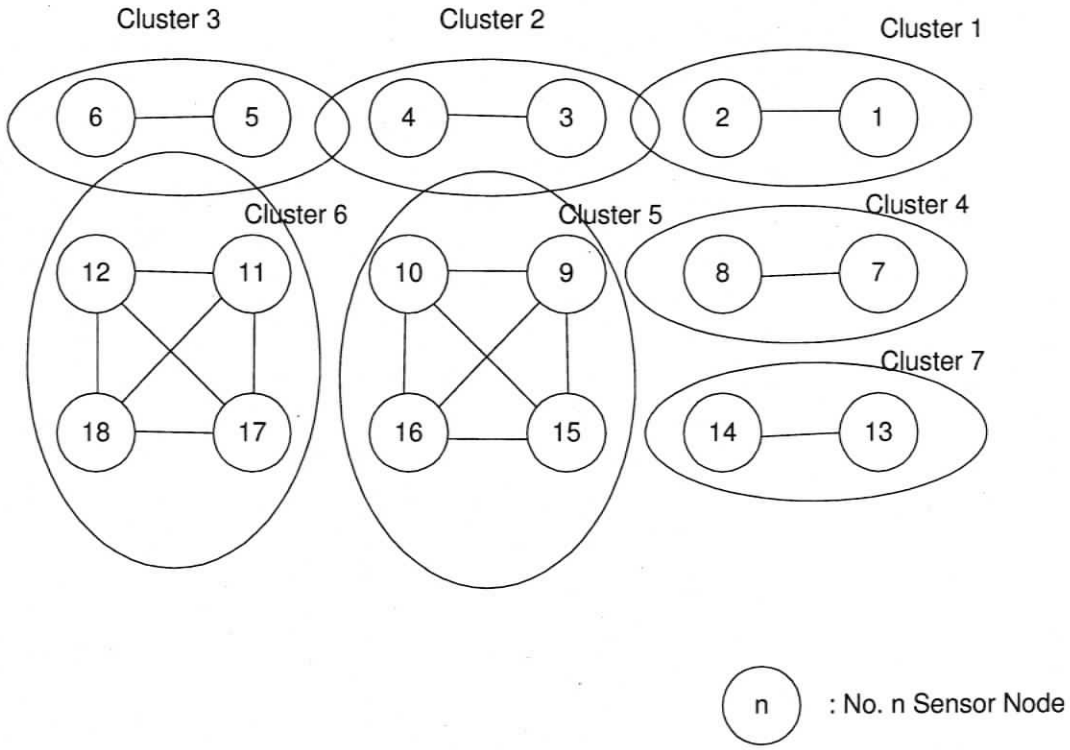

Figure 5.8. *The Clustering Result with the Real Dataset*

The absolute value of σ^2 is not meaningful without considering the degree of variation in magnitude. So we normalized the difference distortion measure by the average variation of samples and used it as the metric of observation fidelity. Formally,

$$\sigma^2_{norm} = \frac{\sigma^2}{\frac{\sum_{i=1}^{N} Var(X_i)}{N}} = \frac{N\sigma^2}{\sum_{i=1}^{N} Var(X_i)},$$

where $Var(X_i)$ is the observation variation of the i-th sensor node. Generally, a smaller σ^2_{norm} value indicates a higher observation fidelity. When we set $m = 30, t = 95\%$, and $gmax\_dst = 3$ feet, the σ^2_{norm} of the light intensity in our test bed is equal to 0.0093, indicating that the data collected with EEDC has high fidelity.

5.6.3.3 Energy Saving

In this case study, at any time instant, only one sensor node in a cluster is scheduled to work. The eighteen sensor nodes were grouped into seven clusters with EEDC as shown in Fig. 5.8. By calculating $\frac{2*2*5+4*4*2}{18} \approx 3$, we can see that without using EEDC, on average each sensor will spend three times more energy in sampling and data transmission.

5.6.4 Large-scale Synthetic Data Generation

Due to the high system cost, we cannot afford an experiment with hundreds of sensor nodes. In order to further investigate the performance of EEDC with large-scale networks, we synthetically generated large traces of spatially correlated dataset based on a mathematical model proposed in [29]. We utilized the software toolkit provided by [29] to extract the model parameters from small-scale real datasets and generate large-scale synthetic datasets based on the model parameters. The toolkit has been validated by comparing the statistical features of the synthetic dataset and the experimental dataset [29].

Initially, we used our test bed in Section 5.6.2 to collect a small-size real dataset. Then we utilized the synthetic data generation toolkit [29] on the dataset from each individual sub-region to generate a larger dataset for each individual sub-region. As a result, a field

consisting of nine distinguished sub-regions with 100 sensor nodes in a 10×10 grid layout was generated, as shown in Fig. 5.9.

5.6.5 Performance Results on Large-scale Synthetic Data

5.6.5.1 The Correctness of Clustering with EEDC

Since we know which sensor node belongs to which sub-region as a priori, it is easy to verify the correctness of the clustering algorithm. We set $m = 20, t = 95\%, gmax\_dst = 8$ distance units. The distant unit is defined as the distance between the two neighboring sensor nodes in a row. By calculating the pairwise dissimilarity measure and performing the clustering algorithm, we obtained nine clusters, each for a sub-region.

5.6.5.2 The Observation Fidelity and Energy Saving with EEDC

By varying the value m in the magnitude m-similarity and applying different intracluster scheduling methods, we collected a set of performance data, based on which Fig. 5.10, Fig. 5.11 and Fig. 5.12 are drawn. Fig. 5.10 demonstrates that with the decrease of m, the number of cliques increases. This conforms to intuition, because a lower m value corresponds to a higher data resolution requirement. Note that the number of cliques is irrelevant to the intracluster scheduling methods. Therefore Fig. 5.10 will not change under different scheduling methods.

Fig. 5.11 and Fig. 5.12 compare the performance of two different intracluster scheduling methods discussed in previous sections, in terms of observation fidelity and energy saving. With the round-robin scheduling, only one sensor node from a cluster is working at a time, while with the joint round-robin and randomized scheduling, multiple sensor nodes are working simultaneously with each cluster. Note that in our experiment, the wakeup probability λ is chosen cluster by cluster, such that the expectation of the split detection delay of a specific cluster $T_d \leq 4T$. The data restoration rule discussed in Section 5.4.5 is applied to both cases to restore the observations of sleeping nodes. Note that since with

the round-robin scheduling, there is only one working node in each cluster at any given time, the sink node simply assumes that the sampling values of all sleeping sensor nodes in the same cluster are equal to that of the working node. Fig. 5.11 and Fig. 5.12 clearly demonstrate the tradeoff between energy saving and observation fidelity: compared with the round-robin scheduling, the joint round-robin and randomized scheduling can significantly improve the observation fidelity but with a smaller energy saving.

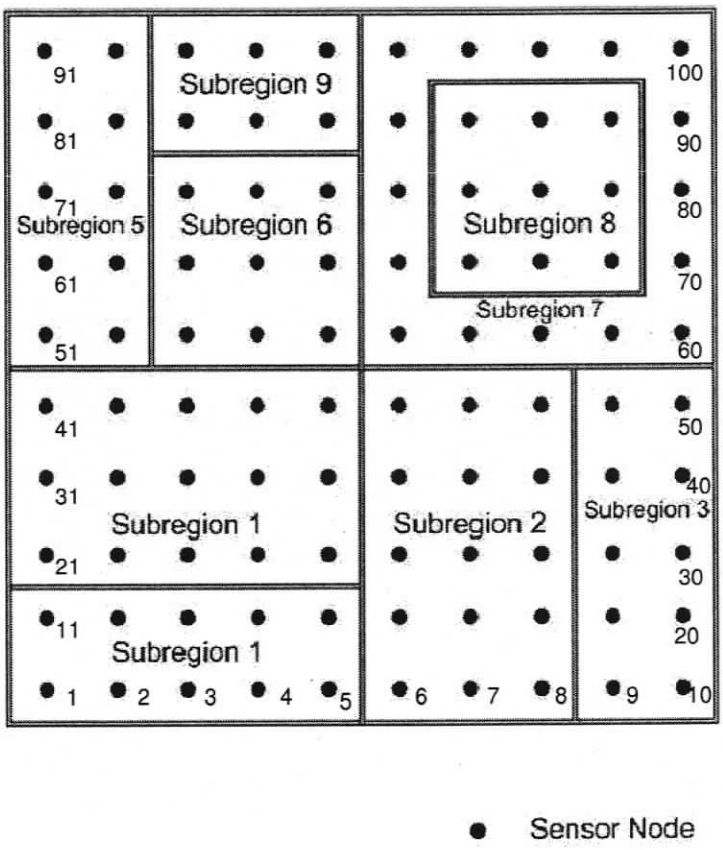

Figure 5.9. *The Field with Nine Distinguished Sub-regions*

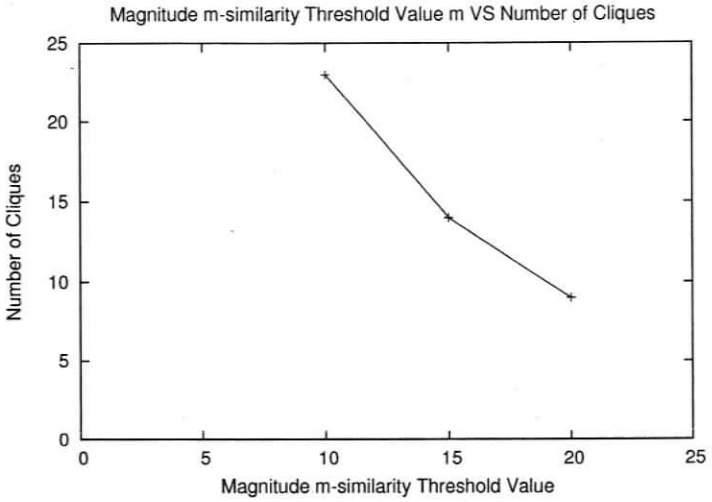

Figure 5.10. *Magnitude M-similarity Threshold vs Number of Cliques*

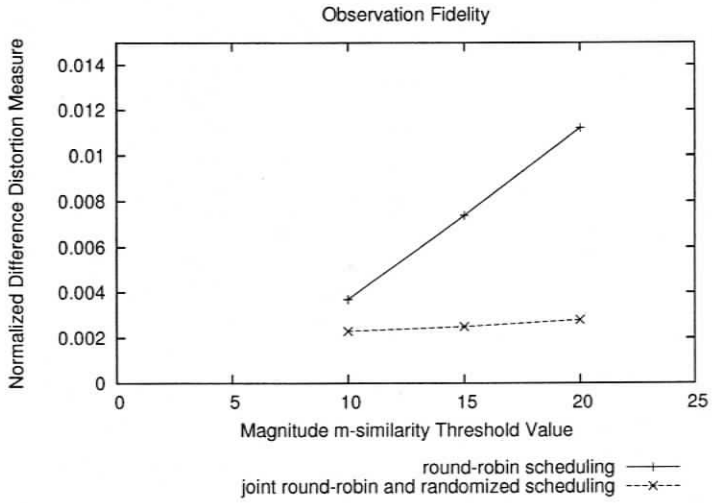

Figure 5.11. *Observation Fidelity*

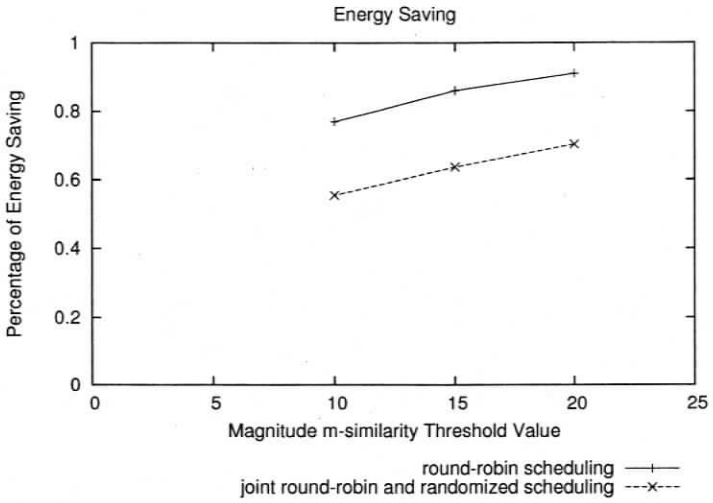

Figure 5.12. *Energy Saving*

5.6.5.3 Energy Saving

At any time instant, only one sensor node in a cluster is required to work. Based on the clustering result, the 100 sensor nodes were grouped into 9 clusters with EEDC. By calculating

$$\frac{16 * 16 + 9 * 9 * 2 + 6 * 6 + 10 * 10 * 3 + 15 * 15 * 2}{100}$$

$$\approx 12,$$

we can see that without using EEDC, on average each sensor will spend 12 times more energy in sampling and data transmission.

5.6.5.4 Response to Spatial Correlation Changes

In order to verify that the spatial correlation changes can be detected by EEDC and demonstrate how EEDC responds to the changes, we create a scenario where an opaque object suddenly covers sensor node 10, 20, 30, 40, 50 in subregion 3, and makes the covered area totally dark. Therefore, the readings from those covered sensor nodes should be totally different from those from the rest of the nodes and a cluster split action is expected in response

to the significant change in the spatial correlation.

In this scenario, the joint round-robin and randomized scheduling is used and the wakeup probability λ is set such that the expectation of the split detection delay of every cluster $T_d \leq 4T$. An active sensor node sends 10 packets in a working shift with a length of T. We monitor the number of packets sent back to the sink node from the 10 sensor nodes in Subregion 3 before and after this period, which is demonstrated in Fig. 5.13. Originally, the number of packets sent to the sink node per work shift oscillates around 30 packets. The lid comes in at time instant 3.5T. The sink detects the changes in spatial correlation between 4T and 5T and instructs all the sensor nodes to wake up and send samples at 5T. The sink node gets enough data for re-clustering at 6T, executes the clustering algorithm to split the sensor nodes in Subregion 3 to 2 clusters and sends the updated working schedules to the sensor nodes at 7T. Afterwards, the sensor nodes work according to the updated schedule and the number of packets sent to the sink node oscillates around 40 packets.

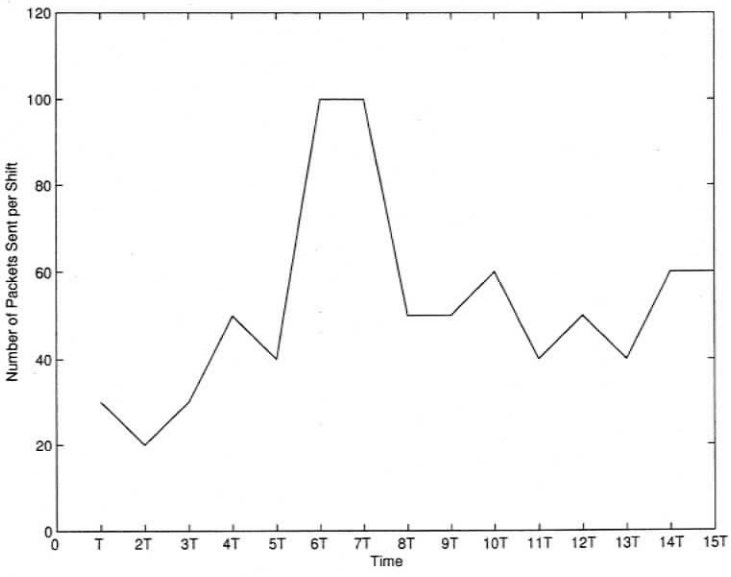

Figure 5.13. *Response to Spatial Correlation Changes*

5.6.6 Performance Evaluation of Exploring Temporal Correlation

To evaluate the performance of the proposed PLAMLiS algorithm, we first applied the algorithm to an indoor temperature dataset we collected on the Crossbow Mica2 platform. The sampling interval is set to 2 minutes and 877 samples were taken in total.

By varying the error bound from 0.5 Celsius degrees to 0.05 Celsius degrees, we got Fig. 5.14, Fig. 5.15, Fig. 5.16 and Fig. 5.17. From these figures we can see that the number of line segments generated by the PLAMLiS algorithm to fit the original data curve is significantly less than the number of points in the original data curve. When the error bound is set to 0.5, only 12 line segments are needed to transmit. Considering each line segments is presented by two end points, only 24 points are transmitted with PLAMLiS algorithm against 877 points are transmitted without using PLAMLiS algorithm. This translates to a 97.3% data reduction in transmission.

With the PLAMLiS algorithm, the transmitted data increases with the decrease of error bound. However, as shown in Fig. 5.17, even the error bound is as small as 0.05 Celsius, only 158 line segments needs to be transmitted, which translates to a 64.0% data reduction in transmission.

We also applied the PLAMLiS to the light intensity dataset we collected in our experimental test bed. Unlike the indoor temperature dataset, the light intensity dataset is not smooth at all. However, even in this case, fairly amount energy could be saved with our PLAMLiS algorithm. Fig. 5.18 illustrates the scenario when error bound is set to 20. Fifty-eight line segments are needed to approximate the time series, which translates to a 42% reduction in transmission.

5.6.7 Performance of Jointly Exploring Spatial Correlation and Temporal Correlation

In this phase, both the PLAMLiS algorithm and the dynamical clustering algorithm are integrated into EEDC, as illustrated in Fig. 5.2.

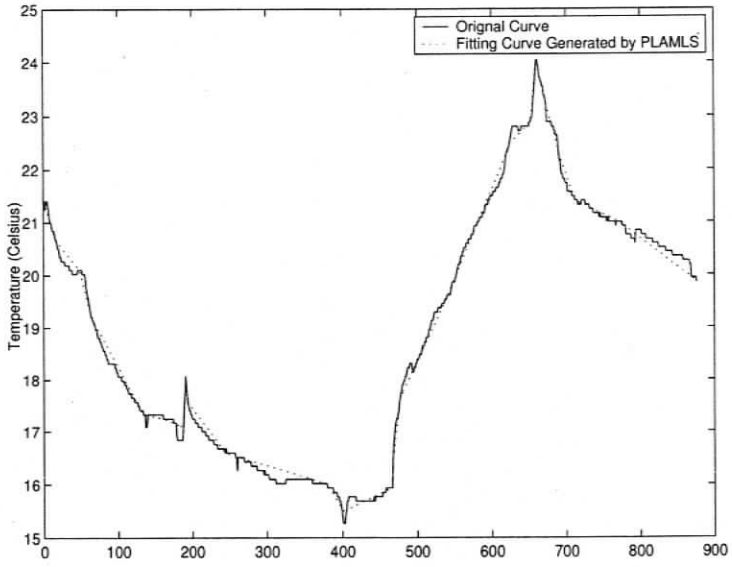

Figure 5.14. *Error Bound=0.5, Number of Line Segments with PLAMLiS=12*

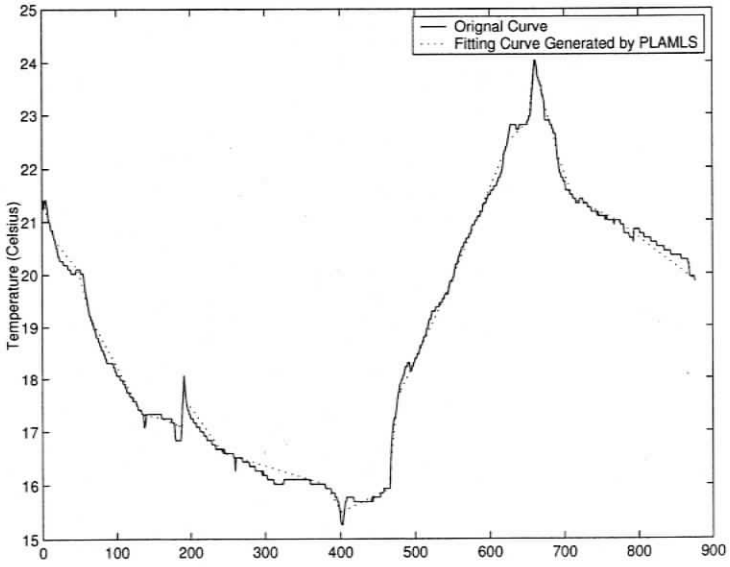

Figure 5.15. *Error Bound=0.3, Number of Line Segments with PLAMLiS=23*

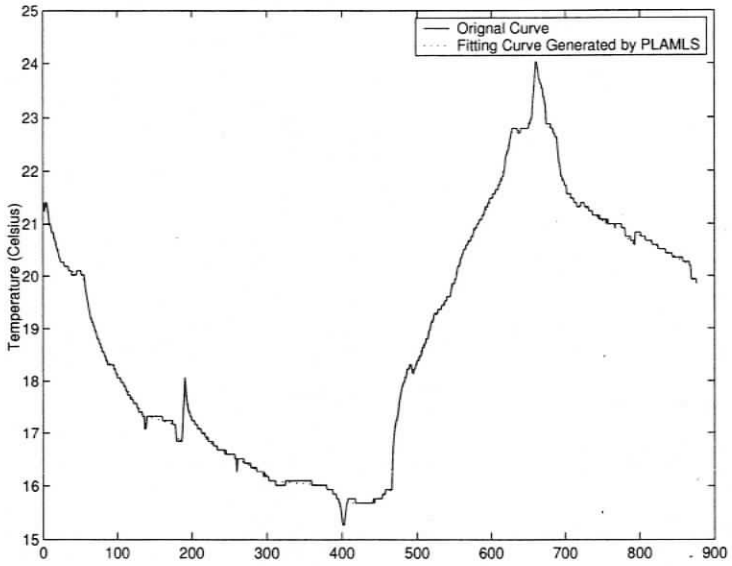

Figure 5.16. *Error Bound=0.1, Number of Line Segments with PLAMLiS=56*

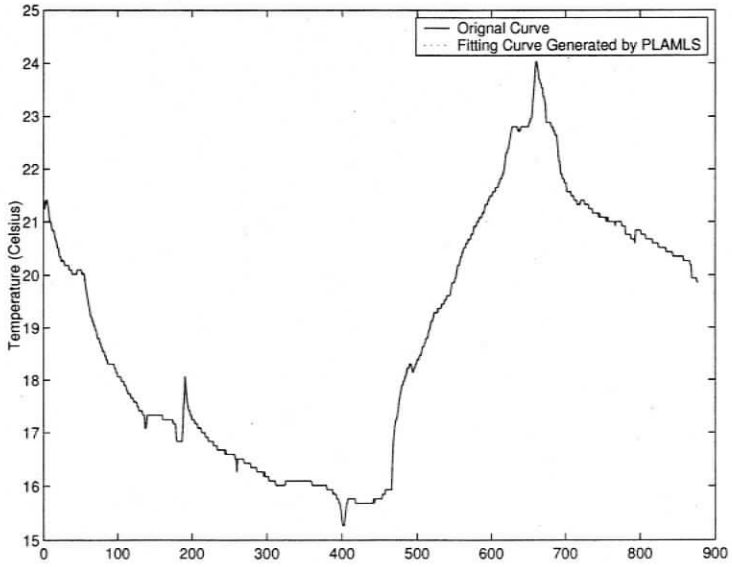

Figure 5.17. *Error Bound=0.05, Number of Line Segments with PLAMLiS=158*

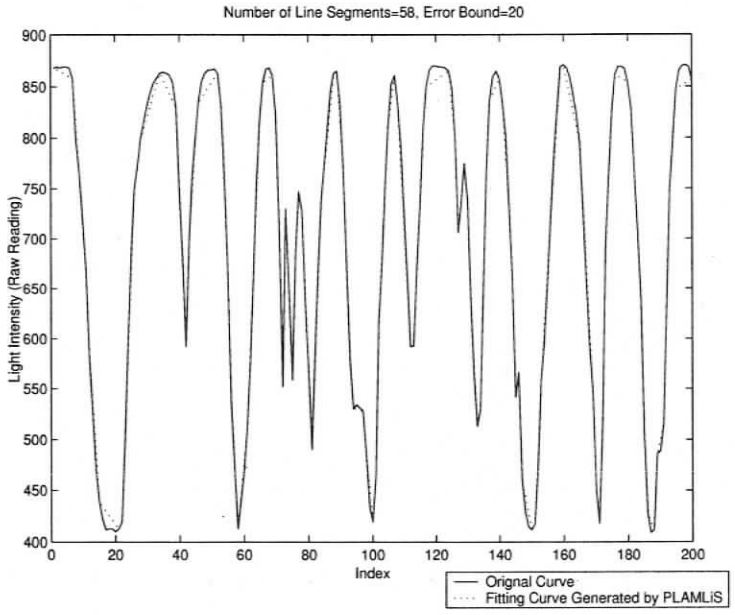

Figure 5.18. *Error Bound =20, Number of Line Segments with PLAMLiS=58*

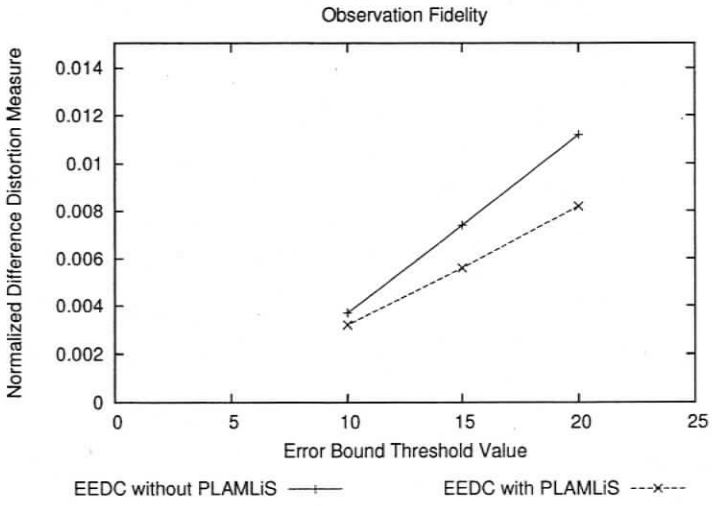

Figure 5.19. *Observation Fidelity*

5.6.7.1 Restoration Data Quality

Suppose the error bound of the dynamical clustering algorithm is set to ϵ_1 (i.e., $m = \epsilon_1$ in the measure of magnitude m-similar) and the error bound of the PLAMLiS algorithm is set to ϵ_2. Since the temporal correlation and spatial correlation are exploited orthogonally, it is easily to see that in the enhanced EEDC framework, the error bound of data restoration $\epsilon = \epsilon_1 + \epsilon_2$. That is to say, any restoration value in the sink node will not differ from its actual value more than $\epsilon_1 + \epsilon_2$.

However, given an error bound ϵ with the enhanced EEDC framework, we face a question of assigning appropriate values to ϵ_1 and ϵ_2, such that $\epsilon_1 + \epsilon_2 = \epsilon$ and data transmission is minimized. The heuristic is that for the sensor nodes having strong temporal correlation in their readings, ϵ_2 can be set larger. For the sensor nodes having strong spatial correlation in their neighborhood, ϵ_1 can be set larger. The method of obtaining the optimal values and dynamically adjusting these values is left as our future work.

5.6.7.2 Performance Evaluation

To illustrate the benefit of using the PLAMLiS algorithm in the EEDC framework, we evaluate the performance of EEDC without PLAMLiS as a benchmark. In all tests, round robin scheduling is used for simplicity. In the EEDC framework without PLAMLiS, the error bound threshold value ϵ is equal to m in the magnitude m-similar measure. In the EEDC framework with PLAMLiS, the error bound threshold value $\epsilon = \epsilon_1 + \epsilon_2$, as discussed above. In all our test cases, ϵ_1 takes 75% of ϵ and ϵ_2 takes 25%.

Fig. 5.19 and Fig. 5.20 clearly demonstrate that after introducing the PLAMLiS algorithm, the data restoration accuracy is improved and the energy consumption decreases. This is because given the same error bound ϵ, the actual error bound used for dynamical clustering ϵ_2 in the enhanced EEDC framework is smaller than that in the EEDC framework without PLAMLiS. Therefore more cliques are generated, which is illustrated in Fig. 5.21. This actually increases the spatial sampling rate and helps improve the data restoration ac-

curacy. The increase in energy consumption by using more clusters is offset by the energy saving with the PLAMLiS algorithm. As a result, the enhanced EEDC framework achieves data accuracy improvement and energy saving at the same time.

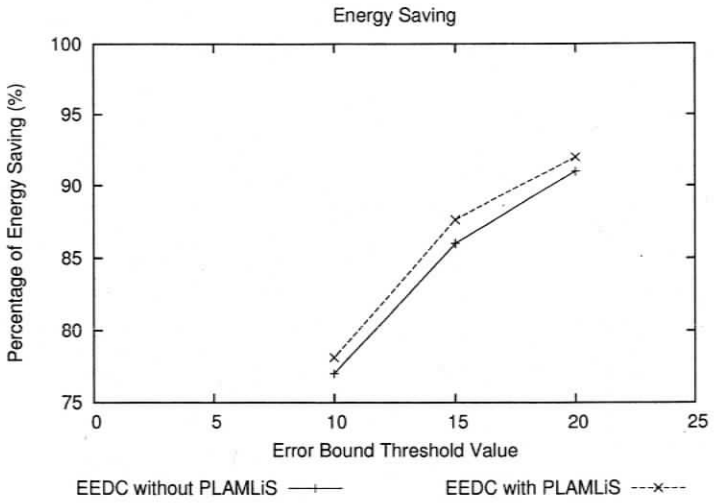

Figure 5.20. *Energy Saving*

5.7 Summary

In this chapter, we design an Energy Efficient Data Collection (EEDC) framework that is aware of the spatiotemporal correlation among the sensing data. Spatial correlation is exploited by dynamically grouping sensor nodes into clusters based on the dissimilarity measure of sampling data. With this method, the whole network is divided into several sub-regions, with each covered by a cluster of sensor nodes. Since the clusters are based on the features of sampling data, scheduling based on the clusters is much more accurate than scheduling based purely on the sensing range of sensor nodes. Temporal correlation is exploited by using the piecewise linear approximation technique to represent the sampling data and minimize the data transmission under a given bound on approximation accuracy. We discuss the details of all major components in the EEDC framework, including the

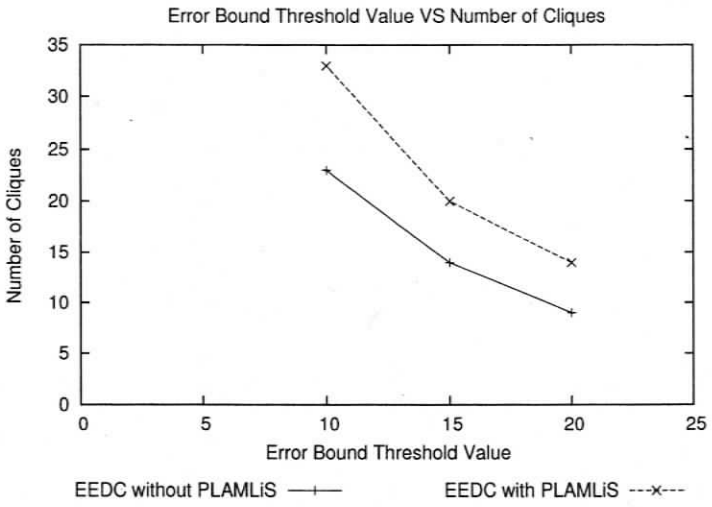

Figure 5.21. *Error Bound Threshold vs Number of Cliques*

calculation of dissimilarity, sensor clustering, sensor scheduling, data restoration, and the PLAMLiS algorithm.

We thoroughly evaluate the performance of the EEDC framework with a real experiment based on MICA2 motes [1] and with a large-scale synthetic dataset. Experimental results demonstrate that the EEDC framework can effectively save energy without losing observation fidelity.

Chapter 6

Conclusions and Future Work

6.1 Summary

With recent advance in wireless communication and Micro-Electro-Mechanical Systems (MEMS) technology, tiny sensor nodes have been constructed to integrate sensors, processors, memory, wireless communication, and power supply within the size of several cubic millimeters. Large quantities of such sensor nodes can be simply dropped in place to monitor a wide variety of real-world phenomena. Through short-range wireless communication, sensor nodes can form WSNs to coordinate their behavior, collect and relay observed data, and process the data in a distributed fashion. Therefore, WSNs foster a lot of novel and promising applications.

However, WSNs have their unique characteristics, which make it impossible to directly apply existing algorithms and protocols in wired and wireless networks to WSNs as effective solutions. These distinguished characteristics impose a set of *requirements* on the underlying network service, which are significantly different from all existing networks. Despite the diversity of WSN applications, some services of underlying WSN are so fundamental that they are required by most WSN applications. We call them *the fundamental services*, as we believe these services are most important and indispensable for a wide variety of WSNs applications and can serve as the building blocks for building up WSN applications.

This thesis work provides the solutions to three fundamental services, namely localiza-

tion service, joint scheduling service for sensing coverage and network connectivity and adaptive sampling service.

In the first part of thesis, we propose a range-free localization method ROCRSSI as our solution to the localization service. It is a range-free method and it does not require any extra dedicated hardware for localization. We perform simulation study to investigate its performance in large-scale networks. Besides, our method has been implemented on broadly-used MICA2 Motes and the in-field experiment has been conducted in small-schale networks by Scott [49]. Compared to one of the best existing range-free localization algorithm, APIT [23], ROCRSSI exhibits better localization accuracy with lower communication cost and other good features.

In the second part of thesis, we propose a joint scheduling method that provides statistical sensing coverage and guaranteed network connectivity. We use randomized scheduling for sensing coverage and then turn on extra sensor nodes, if necessary, for network connectivity. Our method is totally distributed, is able to dynamically adjust sensing coverage with guaranteed network connectivity, and is resilient to time asynchrony. We present analytical results to disclose the relationship among node density, scheduling parameters, coverage quality, detection probability, and detection delay. Analytical and simulation results demonstrate the effectiveness of our joint scheduling method.

In the last part of thesis, we propose an adaptive sampling method that is based on a careful analysis of the surveillance data reported by the sensors. By exploring the spatial correlation of sensing data, we dynamically partition the sensor nodes into clusters so that the sensors in the same cluster have similar surveillance time series. They can share the workload of data collection in the future since their future readings may still likely be correlated and hence predictable. Furthermore, during a short time period, a sensor may report predictable readings. Such a correlation in the data reported from the same sensor is called temporal correlation, which can be explored to further save energy. We develop a generic framework called EEDC to address several important technical challenges, including how to partition the sensors into clusters, how to dynamically maintain the clusters in response

to environmental changes, how to schedule the sensors in a cluster, how to explore temporal correlation, and how to restore the data in the sink with high fidelity. We conduct an extensive empirical study to test our method using both a real test bed system and a large-scale synthetic dataset.

We believe these services we proposed are most important and indispensable for a wide variety of WSNs applications. On one hand, they are orthogonal and can work independently with each other. On the other hand, these services are complementary to each other in order to form an integrated solution for a scalable, robust and energy-efficient wireless sensor network system.

6.2 Future Work

There are many research challenges in wireless sensor networks. Close to this thesis, the following problems deserve further investigation: security, in-network data aggregation, and application specific design.

In this thesis, we assume that every node is reliable and honest. In certain hostile environment such as a battlefield, sensor nodes may be captured and may inject bogus information into the networks. For instance, if an anchor node is compromised and sends out fake location information, sensors using this fake information will create large localization errors; if a sensor node sends fake emergency events such as very high temperature readings to the sink node, the end user may make wrong decisions based on the false information. Due to the constraints on sensors' energy supply and calculation capability, traditional cryptographic operations may not be suitable for wireless sensor networks. It is important and also challenging to make the fundamental services resilient to malicious attacks.

In this thesis, we focus on sensor scheduling and adaptive sampling for energy efficiency. This method is to reduce the data traffic generated by each sensor. Another broadly used method is to reduce the data traffic delivered within the network via in-network data

aggregation. Many research questions have to be answered to explore the advantage of in-network aggregation, e.g., How to organize the aggregation tree? How to integrate in-network data aggregation, sensor scheduling, and adaptive sampling? How to keep a malicious sensor from injecting a bogus aggregation value?

In this thesis, we aim at generic wireless sensor networks and do not investigate specific applications. In practice, a sensor network may be constructed dedicated for a specific task. In this case, the application context is very helpful in simplifying the network architecture. For instance, if sensor nodes are deployed according to pre-defined locations or if we know roughly the network topology, sensor localization could be much easier. How to utilize the application context to simplify network design is a very important and practical research problem.

Bibliography

[1] "Crossbow Technology Inc." Accessed in February 2005. [Online]. Available: http://www.xbow.com/Products/products.htm

[2] Z. Abrams, A. Goel, and S. Plotkin, "Set k-cover algorithms for energy efficient monitoring in wireless sensor networks," in *Proceedings of the Third International Symposium On Information Processing In Sensor Networks (IPSN 2004)*, Berkeley, California, April 2004.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, August 2002.

[4] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proceedings of the IEEE INFOCOM 2000*, Telaviv, Israel, March 2000.

[5] M. Batalin, G. Sukhatme, Y. Yu, M. Rahimi, G. Pottie, W. Kaiser, and D. Estrin, "Call and response: Experiments in sampling the environment," in *Proceedings of ACM SenSys 2004*, Baltimore, MD, November 2004.

[6] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices," *IEEE Personal Communications Magazine*, vol. 8, no. 5, pp. 28–34, October 2000.

[7] A. Cerpa and D. Estrin, "Ascent: Adaptive self-configuring sensor networks topologies," in *Proceedings of the IEEE INFOCOM 2002*, New York, NY, June 2002.

[8] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *Proceedings of ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 2001)*, Rome, Italy, July 2001, pp. 85–96.

[9] J. Chou, D. Petrovic, and K. Ramchandran, "A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks," in *Proceedings of IEEE INFOCOM 2003*, San Francisco, CA, March 2003.

[10] L. Clare, G. Pottie, and J. Agre, "Self-organizing distributed sensor networks," in *Proceedings of SPIE Conference of unattended ground sensor technologies and applictions*, Orlando, FL, April 1999.

[11] A. Coman, M. Nascimento, and J. Sander, "Exploiting redundancy in sensor networks for energy efficient processing of spatiotemporal region queries," in *Proceedings of the 14th ACM Conference on Information and Knowledge Management (CIKM)*, Bremen, Germany, November 2005.

[12] R. Cristescu, B. Beferull-Lonzano, and M. Vetterli, "On network correlated data gathering," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong, China, March 2004.

[13] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *Proceedings of the Thirtieth International Conference on Very Large Data Bases (VLDB 2004)*, Toronto, Canada, August 2004.

[14] D. Doolin and N. Sitar, "Wireless sensors for wild fire monitoring," in *Proceedings of SPIE Symposium on Smart Structures and Materials/NDE 2005)*, San Diego, California, March 2005.

[15] D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Canadian Cartographer*, vol. 10, no. 2, pp. 112–122, December 1973.

[16] G. Edward, P. Box, and G. Jenkins, *Time Series Analysis: Forecasting and Control.* Prentice Hall, 1994.

[17] E. Elson and K. Romer, "Wireless sensor networks: A new regime for time synchronization," in *Proceedings of First Workshop on Hot Topics in Networks*, Princeton, New Jersey, October 2002.

[18] F. Emekci, S. E. Tuna, D. Agrawal, and A. E. Abbadi, "Binocular: A system monitoring framework," in *Proceedings of First Workshop on Data Management for Sensor Networks (DMSN 2004)*, Toronto, Canada, August 2004.

[19] D. Ganesan, D. Estrin, A. Woo, and D. Culler, "Complex behaviour at scale: An experimental study of low-power wireless sensor networks," in *Technical Report UCLA/CSD-TR 02-0013*, 2002.

[20] P. Godfrey and D. Ratajczak, "Robust topology management in wireless ad hoc networks," in *Proceedings of the Third International Symposium On Information Processing in Sensor Networks (IPSN 2004)*, Berkeley, California, April 2004.

[21] H. Gupta, S. Das, and Q. Gu, "Connected sensor cover: Self-organization of sensor networks for efficient query execution," in *Proceeding of the ACM MobiHoc 2003*, Annapolis, Maryland, June 2003.

[22] P. Gupta and P. Kumar, "Critical power for asymptotic connectivity in wireless networks," in *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming,*, Birkhauser, Boston, 1998.

[23] T. He, C. Huang, B. Blum, J. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," in *Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking (MobiCom 2003)*, San Diego, California, September 2003.

[24] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocols for wireless microsensor networks," in *Proceedings of Hawaii International Conference on Systems Science*, Hawaii, January 2000.

[25] J. Hightower, G. Boriello, and R. Want, "Spoton: An indoor 3d location sensing technology based on rf signal strength," in *University of Washington CSE Report 2000-02-02*, 2002.

[26] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *Global Positioning System: Theory and Practice, Fourth Edition*. Springer-Verlag, 1997.

[27] C. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: Random & coordinated sleep algorithm," in *Proceedings of the Third International Symposium On Information Processing In Sensor Networks (IPSN 2004)*, Berkeley, California, April 2004.

[28] H. Jagadish, A. O. Mendelzon, and T. Milo, "Similarity-based queries," in *Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'95)*, San Jose, CA, May 1995.

[29] A. Jindal and K. Psounis, "Modeling spatially-correlated sensor network data," in *Proceedings of Sensor and Ad Hoc Communications and Networks 2004 (SECON2004)*, Santa Clara, CA, October 2004.

[30] E. Keogh and M. Pazzani, "An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback," in *Proceedings of the 4th International Conference of Knowledge Discovery and Data Mining*, New York City, NY, August 1998.

[31] E. Keogh and P. Smyth, "A probabilistic approach to fast pattern matching in time series databases," in *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining*, Newport Beach, CA, August 1997.

[32] L. Kleinrock and J. Silvester, "Optimum transmission radii for packet radio networks or why six is a magic number," in *Proceedings of National Telecomm Conference*, 1978, pp. 4.3.1–4.3.5.

[33] Y. Ko and N. Vaidya, "Location-aided routing (LAR) mobile ad hoc networks," in *Proceedings of the Fourth Annual International Conference on Mobile Computing and Networking (MobiCom 98)*, Dallas, Texas, October 1998.

[34] C. Liu, "Randomized scheduling algorithm for wireless sensor networks," in *Project Report of Randomized Algorithm*, University of Victoria, March 2004.

[35] C. Liu and K. Wu, "Optimal and approximate solutions to plamlis problem in wireless sensor networks," in *Technical Report DCS-313-IR5*, Computer Science Department, University of Victoria, September 2005.

[36] C. Liu, K. Wu, and V. King, "Randomized coverage-preserving scheduling schemes for wireless sensor networks," in *Proceedings of IFIP Networking Conference 2005*, Waterloo, Canada, May 2005.

[37] C. Lund and M. Yannakakis, "On the hardness of approximating minimization problems," *Journal of the ACM*, vol. 41, no. 5, September 1994.

[38] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: Tiny aggregation service for ad-hoc sensor networks," in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI'02)*, Boston, MA, December 2002.

[39] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the WSNA 2002*, Atlanta, GA, September 2002.

[40] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Coverage problems in wireless ad-hoc sensor neworks," in *Proceedings of IEEE Infocom 2001*, Ankorange, Alaska, April 2001.

[41] R. Nagpal, "Organizing a global coordinate system from local information on an amorphous computer," in *A.I. Memo1666, MIT A.I. Laboratory*, August 1999.

[42] D. Niculescu and B. Nath, "DV-based positioning in ad hoc networks," *Telecommunication Systems*, vol. 22, no. 1-4, pp. 267–280, July 2003.

[43] S. Pattem, B. Krishnamachari, and B. Govindan, "The impact of spatial correlation on routing with compression in wireless sensor networks," in *Proceedings of the Third International Symposium On Information Processing In Sensor Networks (IPSN 2004)*, Berkeley, California, April 2004.

[44] Y. Qu, C. Wang, and S. Wang, "Supporting fast search in time series for movement patterns in multiples scales," in *Proceedings of the 7th International Conference on Information and Knowledge Management*, Bethesda, Maryland, November 1998.

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

[46] A. Savvides, C. Han, and M. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom 01)*, Rome, Italy, July 2001.

[47] K. Sayood, *Introduction to Data Compression*. Morgan Kaufmann, 1996.

[48] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Strivastava, "Topology management for sensor networks: Exploiting latency and density," in *Proceedings of the ACM MobiHoc 2002*, Lausanne, Switzerland, June 2002.

[49] T. Scott, "Experimental study of a range-free localization method for wireless sensor networks," in *Master Project Report*, Computer Science Department, University of Victoria, March 2006.

[50] T. Scott, K. Wu, and D. Hoffman, "Radio propagation patterns in wireless sensor networks: New experimental results," in *Proceedings of International Wireless Communications and Mobile Computing Conference*, Vancouver, Canada, July 2006.

[51] S. Shakkottai, R. Srikant, and N. Shroff, "Unreliable sensor grids: Coverage, connectivity and diameter," in *Proceedings of IEEE INFOCOM 2003*, San Francisco, CA, April 2003.

[52] Y. Shang, W. Ruml, and Y. Zhang, "Localization from mere connectivity," in *Proceedings of the ACM MobiHoc 2003*, Annapolis, Maryland, June 2003.

[53] H. Shatkay and S. Zdonik, "Approximate queries and representations for large data sequences," in *Proceedings of the 12th IEEE International Conference on Data Engineering*, New Orleans, Louisiana, Februry 1996.

[54] D. Tian and N. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," in *Proceedings of the ACM Workshop on Wireless Sensor Networks and Applicationss*, Atlanta, GA, October 2002.

[55] D. Tian and N. Georganas, "Connectivity maintenance and coverage preservation in wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 6, pp. 744–761, November 2005.

[56] S. Tilak, N. Abu-Ghazaleh, and H. W., "Infrastructure tradeoffs for sensor networks," in *Proceedings of First International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, Atlanta, GA, September 2002.

[57] C. Wang and S. Wang, "Supporting content-based searches on time series via approximation," in *Proceedings of the 12th International Conference on Scientific and Statistical Database Management*, Berlin, Germany, July 2000.

[58] 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 ACM Sensys, 2003*, Los Angeles, CA, November 2003.

[59] K. Wu, Y. Gao, F. Li, and Y. Xiao, "Lightweight deployment-aware scheduling for wireless sensor networks," *ACM/Kluwer MONET Journal, Special Issue on Energy Constraints and Lifetime Performance in Wireless Sensor Networks*, vol. 10, no. 6, pp. 837–852, December 2005.

[60] Y. Xu, S. Bien, Y. Mori, J. Heidemann, and D. Estrin, "Topology control protocols to conserve energy in wireless ad hoc networks," in *CENS Technical Report 0006*, January 2003.

[61] F. Xue and P. Kumar, "The number of neighbors needed for connectivity of wireless networks," *Wireless Networks*, vol. 10, no. 2, pp. 169–181, 2004.

[62] T. Yan, T. He, and J. Stankovic, "Differentiated surveillance for sensor networks," in *Proceedings of the First International Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, November 2003.

[63] Y. Yao and J. Gehrke, "Query processing for sensor networks," in *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003)*, Asilomar, CA, January 2003.

[64] 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 10th IEEE International Conference on Network Protocols*, Paris, France, November 2002.

[65] S. Yoo and C. Shahabi, "Exploiting spatial correlation towards an energy efficient clustered aggregation technique (CAG)," in *Proceedings of the IEEE International Conference on Communications 2005*, Seoul, Korea, May 2005.

[66] H. Zhang and J. Hou, "Maintaining coverage and connectivity in large sensor networks," in *Proceedings of International Workshop on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless and Peer-to-Peer Networks (invited paper)*, Florida, U.S.A., February 2004.

[67] G. Zhou, T. He, and J. Stankovic, "Impact of radio asymmetry on wireless sensor networks," in *Proceedings of the AcM MobiSys 2004*, Boston, Massachusetts, June 2004.

[68] D. Zuckerman, "NP-complete problems have a version that's hard to approximate," in *Proceedings of the 8th IEEE Annual Structure in Complexity Theory Conference*, San Diego, CA, May 1993.