universität
wien

DISSERTATION

Titel der Dissertation

"Analysis of the Energy Latency Trade-off in Wireless Sensor Networks"

Verfasser

Majid Iqbal Khan

angestrebter akademischer Grad

Doktor der Technischen Wissenschaften (Dr. techn.)

Wien, im July 2009

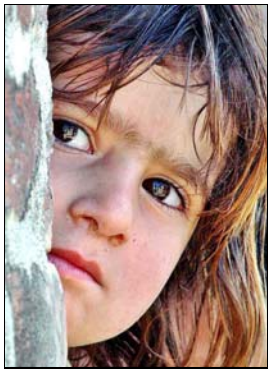
Studienkennzahl lt. Studienblatt:	A 786 881
Dissertationsgebiet lt. Studienblatt:	Informatik
Betreuerin / Betreuer:	o. Univ. Prof. Dr. Günter Haring

ANALYSIS OF THE ENERGY LATENCY TRADE-OFF IN WIRELESS SENSOR NETWORKS

Approved by:

o. Univ. Prof. Dr. Günter Haring
Department of Distributed and Multimedia Systems
University of Vienna

Priv.-Doz. Dr. Wilfried Gansterer
Research Lab Computational Technologies and Applications
University of Vienna



I dedicate my thesis to more than 7 million illiterate children of Pakistan.

ACKNOWLEDGEMENTS

I am grateful to the help of many individuals, without whom this dissertation would not have been possible. First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Günter Haring. He taught me the methodologies and approaches for academic research, and the essential skills of presenting research works. He also gave me freedom in making my own decisions and supported me along the way. During my PhD study, I have always been amazed and inspired by his strive for excellence, passion for research, and his innovative mind. He is truly my role model.

My special thanks, gratitude and appreciation to Dr. Wilfried N. Gansterer for his ever-lasting support during my degree. He has been a source of guidance and motivation all the way till the very end. I also like to thank all the people with whom I worked at the department of the distributed systems for their help in curricular and extra-curricular activities.

I would also like to thank Higher Education Commission of Pakistan (HEC) and government of Pakistan for providing me financial support to complete my degree at a prestigious institute in Vienna, Austria. I also owe thanks to all my friends in Vienna with whom I had a good time while away from home.

Last but not the least I am very grateful to my parents, grandparents, brother and sister for always being there for me. Your encouragement and advices helped me through good and bad times. Your care and support will always accompany me through my adventures.

ABSTRACT

Wireless Sensor Networks (WSN) have gained a considerable attention over the last decade. These networks are characterized by limited amount of energy supply at sensor node. Hence, energy efficiency is an important issue in system design and operation of WSN. This thesis focuses on large-scale applications of WSN, such as environment or habitat monitoring that usually requires ad-hoc deployment of the nodes in large numbers. Ad-hoc deployment and budget constraints restrict developers from programming the nodes with information like routing tables, position coordinates of the node, boundary of the network. In order to acquire this information, state-of-the-art is to program nodes with various initialization schemes that are heavy both from WSN's (energy consumption) and programmer's perspectives (programming effort). In view of these particular constraints, we require a new paradigm for WSN initialization and operation, which should be easy to deploy and have minimal energy demands.

In this thesis, we exploit sink mobility to reduce the WSN initialization and operational overhead. Our first major contribution is a boundary identification scheme for WSN, named *"Mobile Sink based Boundary detection" (MoSBoD)*. It exploits the sink mobility to remove the communication overhead from the sensor nodes, which leads to an increase in the lifetime of the WSN. Furthermore, it does not impose any restrictions on node placement, communication model, or location information of the nodes. The second major contribution is *Congestion avoidance low Latency and Energy efficient (CaLEe)* routing protocol for WSN. *CaLEe* is based on virtual partitioning of a sensor field into sectors and discrete mobility of the sink in the WSN. Our simulation results showed that *CaLEe* not only achieve considerable reduction in average energy dissipation per node compared to current state-of-the-art routing protocols but also accomplish lesser average end-to-end data latency under realistic scenarios. Furthermore, we observe that no single protocol is capable of providing best-case solution *(minium data latency and minimum energy dissipation)* under varying network configurations, which can be defined using communication range of the nodes, node density, throughput of the sensor field etc. Therefore, the third major contribution of this thesis is the identification of operational regions (based on varying network configurations) where one protocol performs better than the other.

In summary, this thesis revisits the classic energy efficiency problem of a WSN (that have resource-limited nodes) while keeping end-to-end data latency under acceptable bounds.

ZUSAMMENFASSUNG

Wireless Sensor Networks (WSNs) haben im letzten Jahrzehnt eine erhebliche Aufmerksamkeit erlangt. Diese Netzwerke zeichnen sich durch begrenzte Energieressourcen der Sensorknoten aus. Daher ist Energieeffizienz ein wichtiges Thema in Systemdesign und -betrieb von WSNs. Diese Arbeit konzentriert sich auf großflächige Anwendungen von WSNs wie Umwelt- oder Lebensraumüberwachung, die in der Regel den Ad-hoc-Einsatz von Knoten in großen Anzahl erfordern. Ad-hoc-Einsatz und Budgetbeschränkungen hindern Entwickler an der Programmierung der Knoten mit zusätzlichen Informationen wie beispielsweise Routingtabellen, Positionskoordinaten, oder Netzwerkgrenzen. Um diese Informationen zu beschaffen, ist es üblich verschiedene Initialisierungsschemen mit erheblichen Auswirkungen auf den Energieverbrauch und den Programmieraufwand zu implementieren. In Anbetracht dieser Beschränkungen ist ein neues Paradigma für die Initialisierung und den Betrieb von WSNs notwendig, das sich durch einfachen Einsatz und minimalen Energieaufwand auszeichnet.

In dieser Arbeit nutzen wir Sink-Mobilität, um den Initialisierungsoverhead und den operativen Overhead zu reduzieren. Unser erster großer Beitrag ist ein Boundary Identification Schema für WSNs mit dem Namen "Mobile Sink based Boundary Detection" (MoSBoD). Es nutzt die Sink-Mobilität um den Kommunikationsoverhead der Sensorknoten zu reduzieren, was zu einer Erhöhung der Laufzeit des WSN führt. Außerdem entstehen durch das Schema keine Einschränkungen in Bezug auf Nodeplacement, Kommunikationsmodell, oder Ortsinformationen der Knoten. Der zweite große Beitrag ist das Congestion avoidance low Latency and Energy efficient (CaLEe) Routingprotokoll für WSNs. *CaLEe* basiert auf der virtuellen Partitionierung eines Sensorsbereich in Sektoren und der diskreten Mobilität der Sink im WSN. Unsere Simulationsergebnisse zeigen, dass *CaLEe*, im Vergleich zum derzeitigen State-of-the-art, nicht nur eine erhebliche Reduzierung der durchschnittlichen Energy Dissipation per Node erzielt, sondern auch eine geringere durchschnittliche End-to-End Data Latency in realistischen Szenarien erreicht. Darüber hinaus haben wir festgestellt, dass kein einziges Protokoll in der Lage ist, eine Best-Case-Lösung (minimale Data Latency und minimale Energy Dissipation) für variierende Netzwerkkonfigurationen, die beispielsweise mithilfe der Parameter Kommunikationsbereich der Nodes, Nodedichte, Durchsatz des Sensorfelds definiert werden können, bieten. Daher ist der dritte Hauptbeitrag dieser Arbeit die Identifikation von (auf unterschiedlichen Netzwerkkonfigurationen basierenden) „Operational Regions“, in denen einzelne Protokolle besser arbeiten als andere.

Zusammenfassend kann man sagen, dass diese Dissertation das klassische Energieeffizienzproblem der WSNs (Ressource-begrenzte Knoten) aufgreift und gleichzeitig die End-to-End Data Latency auf einen annehmbaren Rahmen eingrenzt.

TABLE OF CONTENTS

DEDICATION .... **v**
ACKNOWLEDGEMENTS .... **vii**
ABSTRACT .... **ix**
ZUSAMMENFASSUNG .... **xi**
Chapter 1. INTRODUCTION **1**
1.1 Objective of the PhD research **1**
1.2 Overview of the thesis organization **3**
Chapter 2. BACKGROUND AND MOTIVATION **5**
2.1 Overview of wireless sensor networks **5**
2.1.1 Sensor node 5
2.1.2 Base station / sink 7
2.1.3 Four main types of applications of a WSN 8
2.1.4 Real world examples 10
2.1.5 Factors influencing sensor network design 15
2.2 Focus of the thesis **21**
2.2.1 WSN initialization schemes 21
2.2.2 Route establishment and data routing in WSN 23
2.3 Problem identification **23**
2.3.1 Evaluation metrics 24
2.4 Role of the mobile sink **25**
2.4.1 Effect of sink mobility on end-to-end data latency 26
2.4.2 Conclusion – effects of using a mobile sink in a WSN 27
Chapter 3. BOUNDARY IDENTIFICATION OF A WSN USING A MOBILE SINK **29**
3.1 Research contributions **29**
3.2 Related work **30**
3.2.1 Communication-based methods 30
3.2.2 Hardware-based methods 32
3.2.3 Perimeter-based methods 33
3.2.4 Summary 34
3.3 Network model and terminology **34**
3.3.1 Network model 34
3.3.2 Terminology 35
3.3.3 Sensor node position estimation procedure 36
3.4 MObile Sink based BOundary Detection algorithm (MoSBoD) **37**
3.4.1 Bootstrapping phase 37
3.4.2 Edge node identification and boundary traversal 39
3.5 Shortcomings in the MoSBoD algorithm **43**
3.5.1 Quality of the identified boundary 43
3.5.2 Mobility speed of the sink 43
3.6 MODIFIED MoSBoD (M-MoSBoD) **44**
3.6.1 Improved definition of the boundary 44

3.6.2 Reducing completion time of MoSBoD 46

3.7 Performance evaluation 48

3.7.1 Analyzing effects of refined boundary 48

3.7.2 Analyzing effects of increased duty cycle 49

3.8 Summary 56

Chapter 4. CONGESTION AVOIDANCE FOR LOW LATENCY AND ENERGY EFFICIENCY IN WSN (CaLEe) 59

4.1 Research contributions 59

4.2 Introduction 60

4.3 Related work 61

4.3.1 Static sink based WSNs 61

4.3.2 Mobile sink based WSNs 61

4.3.3 Summary 68

4.4 CaLEe routing protocol 68

4.4.1 Assumptions 68

4.4.2 Algorithm 70

4.4.3 Partitioning and route establishment 71

4.4.4 Open issues in CaLEe routing protocol 73

4.5 Data congestion and data loss in WSN 74

4.5.1 State-of-the-art congestion avoidance and control techniques 74

4.5.2 In-network storage model for data persistence in WSN 77

4.6 Summary 80

Chapter 5. PERFORMANCE EVALUATION OF THE CaLEe ROUTING PROTOCOL 81

5.1 Research contribution 81

5.2 System description and assumptions 81

5.2.1 Sensor behavior 82

5.2.2 Data routing 83

5.2.3 Channel access 84

5.3 Evaluation model 84

5.3.1 Evaluation metrics 85

5.3.2 Evaluation parameters 86

5.3.3 Evaluation environment 87

5.3.4 Evaluated scenarios 88

5.4 Performance results 89

5.4.1 Varying the mobility trajectory of the sink 95

5.4.2 Varying the number of sectors 97

5.4.3 Varying the communication range of the nodes 99

5.4.4 Varying the sensor node density 101

5.4.5 Varying the throughput of the sensor field 103

5.4.6 Varying the size of a sensor field 105

5.5 Summary **107**
Chapter 6: THE SOLUTION: AN ADAPTIVE ROUTING PROTOCOL **109**
6.1 Research contribution **109**
6.2 When to use *SS, MS* and *CaLEe* **109**
6.2.1 Throughput of the WSN and sensor node density 110
6.2.2 Throughput of the WSN and communication range of the nodes 112
6.2.3 Sensor node density and communication range of the nodes 115
6.2.4 Mobility radius of the sink and WSN dynamics 117
6.3 Combined objective function **119**
6.4 Implications of the results in real world scenarios **122**
6.5 Summary **124**
Chapter 7: CONCLUSION **127**
7.1 Future work **128**
APPENDIX – I: Table of figures **131**
APPENDIX – II: List of variables **133**
References **135**

CHAPTER 1. INTRODUCTION

1.1 OBJECTIVE OF THE PHD RESEARCH

Recent advances in the development of low cost sensing devices and microminiaturization have enabled the emergence of a new technology called Wireless Sensor Networks (WSN). WSN based solutions have been designed and implemented in diverse areas including environment and habitat monitoring, building automation, disaster and waste management, infrastructure monitoring etc. [1]. Sensor nodes used in these applications are characterized by limited resources in terms of memory, computation power, and energy [2]. Thus, efficient utilization of the energy resource is one of the main design considerations in many of these applications.

Current literature provides numerous efforts at developing efficient operational and communication schemes for WSNs. Primary concern in all of them is the energy efficiency. This consideration potentially affects many aspects of the system design: hardware, physical layer, MAC layer, addressing and routing, topology control, synchronization, naming scheme, security mechanisms, etc. [3-8].

In our research work, we focus on the primary operation of a WSN, that is, on the collection of sensor data from source nodes at the sink (the data routing process). Our aim is to improve the energy efficiency of this process while keeping average end-to-end data latency $\overline{\Delta}$ in a WSN within acceptable limits. Since data routing is the most energy consuming operation in a WSN, optimization of this process can result into significant extension in the lifetime Φ^{WSN} of a WSN (for details refer to Section 2.1.5.8).

We tackled the problem of energy efficiency at two levels in our research. At the first level, we consider one of the initialization requirements of a WSN, that is, boundary identification of the sensor field. Information about the boundary of a sensor field can be used to determine total area covered by the sensor field, to track the number of events entering or leaving the system, etc. In addition, this information can also be utilized to design an energy efficient routing protocol. However, much of the current state-of-the-art for boundary identification of a WSN has a high-energy demand that is not desirable. In this thesis, we present a boundary identification scheme that exploits the sink mobility for the boundary identification of a WSN. In the process, we specifically investigated the following questions,

- What are the advantages of using a mobile sink for boundary identification? (especially in terms of energy demands)
- What could be bad about using a mobile sink for boundary identification? (especially in terms of completion time of the algorithm)
- What are the energy-latency tradeoffs in a mobile sink based boundary identification scheme? How can we exploit them in the best possible way?

In order to answer these questions, we present a comprehensive mathematical model plus simulation results and draw inferences from them. Results show that our newly proposed Mobile Sink based Boundary Detection scheme *(MoSBoD)* for WSN leads to more than 60% decrease in the average energy dissipation Ψ per sensor node [9, 10] compared to the state-of-the-art boundary identification schemes. In addition, for small communication range of the nodes and high mobility speed of the sink, *MoSBoD* also shows competitive results for completion time compared to state-of-the-art boundary identification schemes.

In the second phase of our research we utilize sink mobility to design a congestion avoidance low latency and energy efficient *(CaLEe)* routing scheme for WSN [11, 12]. *CaLEe* is based on virtual partitioning of a sensor field into sectors; each sector has a data collector node that is responsible for collecting data from the nodes located in its sector. The sink follows a discrete mobility pattern along a path, which connects the data collector nodes in a series. *CaLEe* is a simple and scalable approach that achieves considerable reduction in energy dissipation per node for a wide range of application scenarios. It addresses the challenges of constructing and maintaining data routes in resource constrained sensor networks, as well as practical problems such as balancing the routing load amongst the sensor nodes.

The *CaLEe* routing scheme is based on indirect data routing to the sink; sensor nodes route data to the collector nodes and the sink periodically visits collector nodes to retrieve the data. As a result, we expect relatively high end-to-end data latency $\overline{\Delta}$ in the case of *CaLEe* compared to other state-of-the-art routing schemes such as, direct routing to a static or a mobile sink. In order to make an in-depth analysis of *CaLEe* compared to state-of-the-art routing protocols in terms of data latency $\overline{\Delta}$ we investigated different parameters such as, node density Γ, communication range r of the nodes, throughput G of the sensor field, number of sectors and radius of the mobility trajectory of the sink. It has been shown through extensive simulations that for many realistic scenarios the *CaLEe* routing protocol not only achieves a four fold increase in the lifetime Φ^{WSN} of a WSN compared to state-of-the-art but also results in competitive data latency

$\bar{\Delta}$. One potential problem with *CaLEe* can be a high probability of data loss at the data collector nodes (that have finite buffers) in the case of delayed arrival of the sink. However, this problem is successfully handled by developing an in-network storage model for data persistence [11, 12].

During the analysis of *CaLEe* and current state-of-the-art routing schemes it has been observed that no single routing protocol is capable of providing best-case solution in all scenarios. Furthermore, due to the error prone nature of the sensor nodes network configuration may change during the operation of the network, hence resulting in variable performance (energy efficiency, data latency) from same routing protocol. Therefore, in the last part of the thesis we address the this issue by presenting a comparison of *CaLEe* and two routing schemes from state-of the-art, a *static sink* optimally placed in a WSN *(called SS in the following)* and routing to a *mobile sink* that follows discrete mobility along a fixed trajectory in a WSN *(called MS in the following)*. During the simulations we analyze the impact of six dimensional input space (communication range r of the nodes, node density Γ, throughput G of the WSN, size of a WSN, number of sectors in a WSN and radius of the mobility trajectory of the sink) which defines a network configuration, on the performance (energy efficiency, data latency) of considered routing protocol. Obtained results are then used to identify the operational regions where each of the considered protocol provides optimal performance.

1.2 OVERVIEW OF THE THESIS ORGANIZATION

The rest of the thesis is organized as follows: Chapter 2 provides an overview of wireless sensor networks and motivation for the thesis. Chapter 3 presents in detail current state-of-the-art boundary identification algorithm for WSN. This chapter also presents our newly proposed boundary identification algorithm using a mobile sink. Chapter 4 presents the current state-of-the-art data routing protocols for WSN along with a new scalable congestion avoidance and energy efficient routing protocol (*CaLEe*). It also presents an in-network storage model for data persistence under congestion in WSN. Chapter 5 discusses performance comparison of the *CaLEe* routing protocol with current state-of-the-art data routing schemes. Chapter 6 presents the impact of variation in the network configurations on the performance of *CaLEe, SS and MS*. This chapter also identifies operational regions where each of the considered protocols performs better than the other. Finally, we conclude the thesis in Chapter 7.

CHAPTER 2. BACKGROUND AND MOTIVATION

2.1 OVERVIEW OF WIRELESS SENSOR NETWORKS

With the emergence of small size low cost wireless sensing devices, which found their use in diverse types of applications a new class of networks emerged, called *Wireless sensor networks (WSN)*. These networks consist of spatially distributed autonomous devices called sensor *nodes* or simply *nodes*, which interact with their surroundings by sensing or controlling physical parameters [13]. The sensed values are often transmitted to a base station that is commonly known as *sink*. Figure 2.1 depicts a typical wireless sensor field.

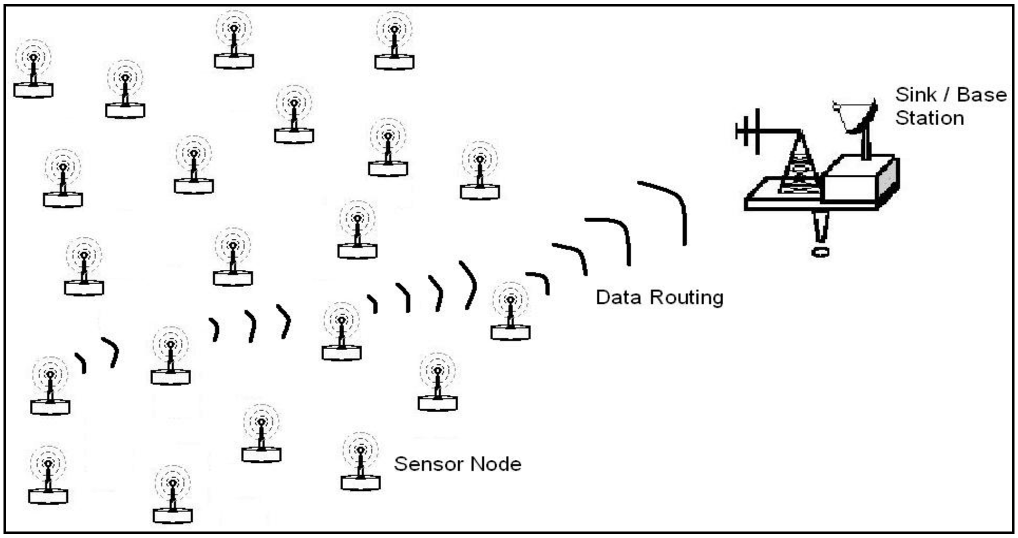


Figure 2.1 A typical wireless sensor network (WSN)

2.1.1 SENSOR NODE

A sensor node is typically composed of a microcontroller, a transceiver, limited storage capacity and an energy resource (a battery or a wired power supply) plus few sensors which are used to sense desired parameters such as temperature, air pressure, light intensity, etc. Depending on the application scenario a node may also be required to estimate its position coordinates in the field, move in the area of interest or perform specialized transmissions (such as data transmission in a particular direction). In order to accomplish these tasks a node can also be equipped with specialized equipment.

Position estimation: It is the task for a node to identify its own position coordinates according to a global or local coordinate system. One way to accomplish this goal is to equip each node with a global positioning system (GPS). However, in large-scale deployments GPS based solutions become very costly. Alternatively, *beacon nodes* (special nodes equipped with a GPS) based schemes have been developed to perform low cost position estimation of the nodes [14-17].

Sensor mobility: It refers to the capability of a node to change its position. A sensor node can be mobile or static. WSNs composed of only mobile nodes or static plus mobile nodes (called mobile WSN) have several advantages over WSN comprised of only static nodes (called static WSN). In case of mobile WSN sensor nodes can reposition themselves in the field to achieve better coverage as well as reduced transmission interference with the neighbouring nodes which results in improved performance of the network. On the other hand, if the mobility pattern of the nodes is uncontrolled (nodes attached to the mobile objects that have random mobility pattern in the field), then designing an efficient mobile WSN with guaranteed end-to-end connectivity and loss free data routing becomes a challenging task.

Hi-tech transceivers: Communication range[1] of a node is defined as the maximum distance up to which a node can communicate with other nodes. It can be fixed or variable (but always upper bounded). Transceivers with variable transmission range are helpful in avoiding a disconnected sensor field, which can result from failing sensor nodes. For example, when a node detects reduced connectivity with its neighboring nodes it increases the transmission range to avoid voids in the sensor field. On the other hand, if a node experiences high channel interference then the transmission range can be reduced to avoid collisions.

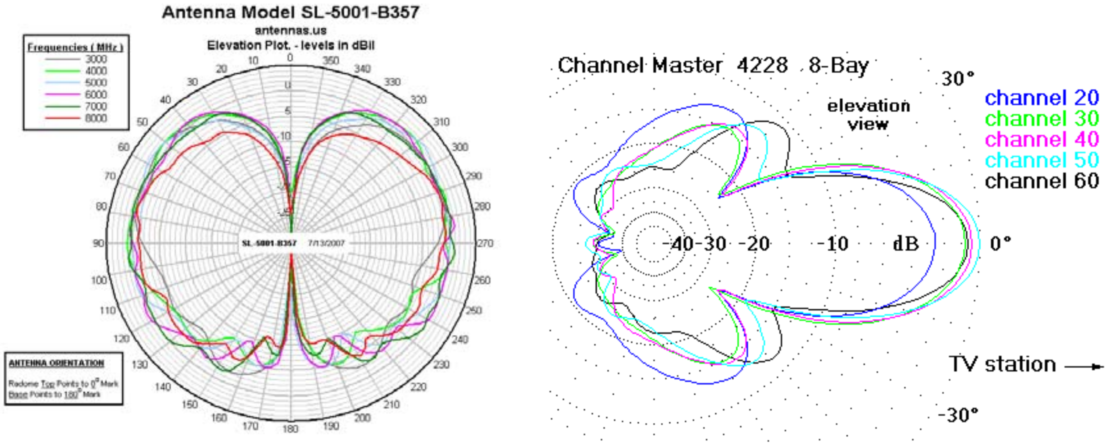
Another categorization of transceiver system can be based upon directional properties of the antenna. The two categories are *omni-directional* and *directional* antennas. It can be inferred from Figure 2.2(a) that under ideal conditions, radiation patterns from an omni directional antenna are almost same in all directions; on the other hand, radiation patterns of a directional antenna can be directed in the desired direction as shown in Figure 2.2(b).

It can be concluded from the above discussion that hi-tech nodes can really make life easy for the developers. However, selection of the node with desired capabilities is greatly influenced by the application scenario and budgetary constraints. For example, environmental or habitat

---

[1] The term communication range in this thesis is used to refer maximum transmission range of a sensor node. Note that communication range is not equal to the signal sensing range that is much greater than the transmission range of a node.

monitoring applications usually require dense deployment of the nodes, therefore it is not preferred to equip each node with expensive equipment such as GPS and hi-tech transceiver. Instead beacon based sensor node location identification [14-17] should be used. On the other hand, issues related to guaranteed end-to-end connectivity and channel contention can be addressed using dense sensor node deployment with an efficient MAC protocol. Thus, by deploying cheap nodes in large numbers instead of expensive nodes comparable performance can be achieved at reduced price.



(a):Omni-directional antenna having same radiation patterns in all direction [18] (b):Directional antenna transmitting radiations in desired direction [19]

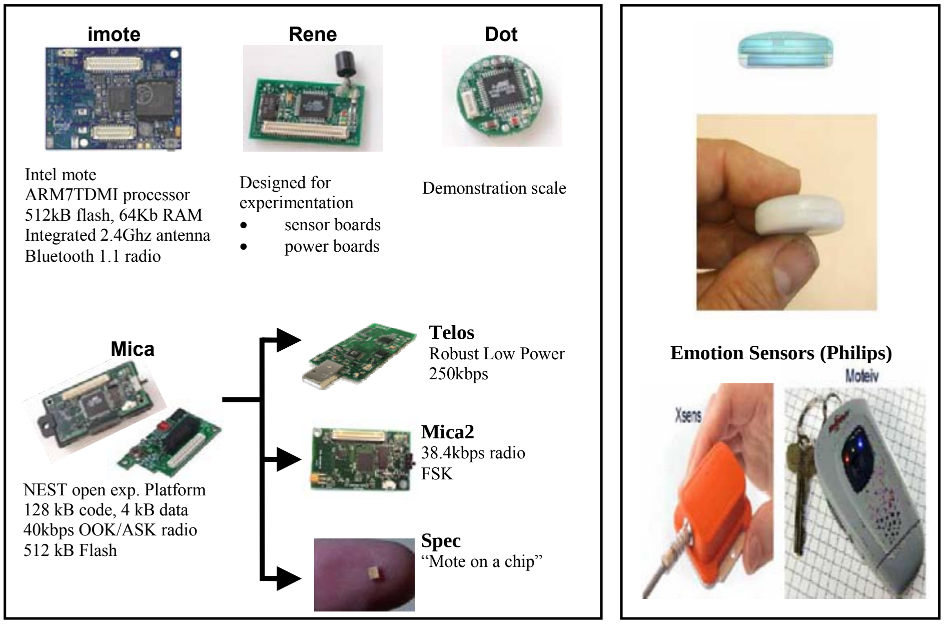
Figure 2.2 Antenna types

Telos, micaz, sky motes, mica2 are some of the most commonly used sensor nodes in research environments and are shown in Figure 2.3 along with some commercially available WSN based solutions.

2.1.2 BASE STATION / SINK

The term sink originates from the fact that all the data routing paths in WSN eventually terminates at a base station, which thus acts as a sink for data from the entire sensor field. In comparison to a sensor node, the sink is assumed to be a constrained free device having high processing power and huge storage capacity. In addition, the number of sinks in a sensor field can vary from one to few therefore, in contrast to sensor nodes, they can be equipped with special devices (if required) such as, GPS for position estimation, directional antennas or antennas with variable transmission, etc., without much influence on the overall budget. In the following, we discuss few characteristics of a sink that can vary from one application to another.

Sink mobility: Sinks can be either static or mobile. During early days of WSN the sink was considered a static device placed in a sensor field. However, with adaptation of WSN in diverse areas of life usability of a mobile sink is explored in detail to achieve better performance at reduced cost. One such example is the work presented in [20] where authors argue that a mobile sink based routing protocol leads to increased lifetime Φ^{WSN} of a WSN compared to that of a static sink. Similarly, its benefits in terms of energy and budget efficiency have also been discussed for sensor field localization [21, 22] and boundary identification of a WSN [9, 10].



Unpackaged sensor nodes

Packaged sensor nodes

Figure 2.3 Sensor nodes

Number of sinks: It has been recognized that if the number of sinks in a WSN is increased an overall improvement in the performance ($\bar{\Delta}$ and Φ^{WSN}) is possible [23]. However, these performance gains are only possible if a strong coordination mechanism amongst the sinks exist, which should be responsible of properly spreading the sinks in the entire field. Designing such a coordination mechanism comes as an added overhead for developers as well as for the WSN because of additional communication between sensor nodes and the sinks.

2.1.3 FOUR MAIN TYPES OF APPLICATIONS OF A WSN

Holger Karl and Andreas Willig [24] categorized applications of a WSN on the basis of data reporting patterns of the sensor nodes. These categories are,

- Periodic sensing

- Event detection
- Object tracking
- Query driven

2.1.3.1 Periodic sensing

Major objective in this class of applications is to analyze changes in the parameter of interest over a period. Sensor nodes are programmed to periodically sense the environment for the said parameter and report its value to the sink. The interval between two consecutive sensing operations of a node depends on the application requirements and can vary from one application to another. Sensor nodes deployed to monitor environmental changes the in Antarctic Ocean for the period of one year; or nodes deployed to study a habitat in the wild, are some examples of this class of applications [25, 26].

2.1.3.2 Event detection

This class encompasses applications used to detect and report occurrence of an event in a sensor field [27, 28]. One way to implement these types of applications is to program nodes with the upper and lower bound values of the parameter of interest such as, temperature. When a node senses an unusual value (outside the given bounds) of the given parameter, it reports an occurrence of this event to the sink. Such applications are often deployed with real time data reporting requirements, thus swift event detection and data reporting is necessary. Fire alarm systems, intruder detection and military surveillance are some examples.

2.1.3.3 Object tracking

Recently, WSN applications for object tracking have attracted increasing attention. Major goal in these applications is to utilize sensor data for tracking mobility pattern of an object [29-31]. For example, consider a farmer who wants to know the real time position of his cattle in the field. Additionally, he wants to know their mobility trajectories taken during the day. This information can be gathered easily by attaching a sensor to each animal that periodically reports the current position of the animal to the base station. The second option is to deploy static nodes with known position coordinates in the field. These nodes broadcast a message to the base station when an animal enters or leaves their coverage area. Thus, by utilizing this information a farmer can understand the behavior of each animal in the herd and can manage his farm better.

2.1.3.4 Query driven

Increasing use of WSN in real life applications has paved the way for the deployment of generic sensor fields, where nodes are deployed to sense multiple parameters. Each node is programmed to publish the collected data but actual data transmission only occurs when the sink queries the sensor field for particular data set [32, 33]. This helps to avoid excessive data transmissions in the field. Furthermore, a single WSN can be used by many users to serve their purposes. For example, a WSN deployed in a metropolitan area can be configured to gather data about traffic patterns as well as air pollution. A user interested in finding a congestion free route to his/her destination can query the WSN for the desired information. On the other hand, if someone is interested in measuring how the air pollution in the city varies during the day can also retrieve this information from the same WSN.

2.1.4 REAL WORLD EXAMPLES

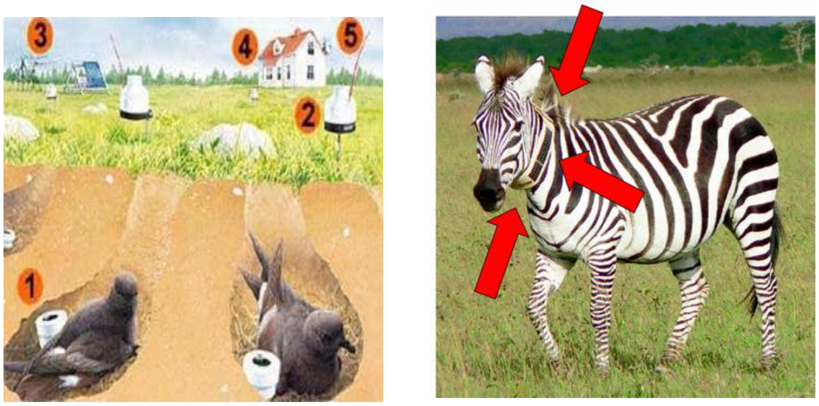
This Section discusses some real world examples of WSN applications mentioned in Section 2.1.3.

2.1.4.1 Habitat monitoring

Study of animal behavior in the wild requires long-term habitat monitoring that can span over weeks or months. It is often not possible for the scientists to spend such a long time in the wild; moreover, they prefer to observe the habitat without disturbing the inhabitants. In order to fulfill these requirements, computer scientists came up with a WSN based solution where sensor nodes are programmed for periodic sensing of the parameter of interest and its reporting to the base station. In the following we discuss two interesting examples of habitat monitoring that are *The Great duck island project* [34] and *Zebra Nets* [25, 26].

Great duck island project [34]*:* Main motive behind this project was to develop a habitat monitoring kit that should enable researchers worldwide to engage in the non-intrusive and non-disruptive monitoring of sensitive wildlife and habitats. The project was initiated to monitor the microclimate in and around the nesting burrows used by the Leach's Storm Petrel. When large number of birds in the area and long term observation requirements caused traditional techniques to fail, then a WSN comprising of one hundred mica motes was deployed in the field as shown in Figure 2.4(a). Numbers 1 to 4 in Figure 2.4(a) show the location of the sensor nodes that are responsible for sensing desired parameters as well as relaying data from other nodes towards the base station positioned at location 5. So, a researcher sitting in laboratory can collect all the required data with great ease without disturbing the inhabitants.

ZebraNet project [25, 26]*:* This project was initiated to determine and analyze the unknown migration patterns of zebras that spans over a very large field. In this project, sensors were attached to zebras in the form of a collar that is equipped with a rechargeable battery (solar cells), two radios for long and short-range communications with the base station and neighbours respectively, some sensors for measuring the heart rate, body temperature, frequency of feeding and a GPS as shown in Figure 2.4(b). Sensors were also equipped with small storage space used to store data in case the forwarding node or the sink is out of reach.



(a): Great duck island project [35] (b): ZebraNet project [36]

Figure 2.4 Habitat Monitoring

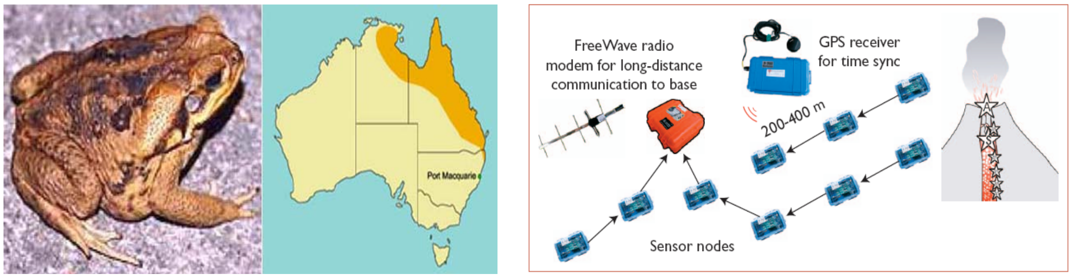
Both the Great duck island project and ZebraNet belong to periodic sensing application type discussed in Section 2.1.3, ZebraNet also provides a good example of a mobile WSN.

2.1.4.2 Environmental monitoring

Long-term, unattended and cordless operation of sensor nodes close to the target area makes them most suitable choice for the environmental monitoring applications. Cane-toad monitoring in monsoonal woodlands of northern Australia [37] and an active volcano monitoring in Volcán Reventador in northern Ecuador [38] are two examples of this class.

Cane-toad monitoring [37]: Researchers investigated a wireless acoustic sensor network application — that was used to monitor amphibian population in the monsoonal woodlands of northern Australia as shown in Figure 2.5(a). The goal of the project was to use automatic recognition of animal vocalizations to estimate the populations of native frogs and the invasive introduced species. This was a challenging application because it requires high frequency acoustic sampling, complex signal processing and wide area sensing coverage. Therefore, a hybrid of resource rich Stargate device used for Fast Fourier Transforms (FFTs) and machine learning, on the other hand cost effective Mica2 devices were used to implement rest of the network.

Active Volcano monitoring [38]: Werner-Allen et.al, deployed a large sensor network on Volcán Reventador in northern Ecuador. The sensor array consisted of 16 nodes equipped with seismo acoustic sensors deployed over 3 km as shown in Figure 2.5(b). Sensor nodes were responsible of collecting data from their surrounding and routing it to an observatory where a laptop was used to log the data. Deployed network was used to captured 230 volcanic events over the period of three weeks.



(a): Cane-toad monitoring [37]

(b): Active Volcano monitoring [38]

Figure 2.5 Environmental monitoring

Cane-toad monitoring is also an example of periodic sensing application type while Active Volcano monitoring is also an example of event monitoring application of a WSN.

2.1.4.3 Military surveillance

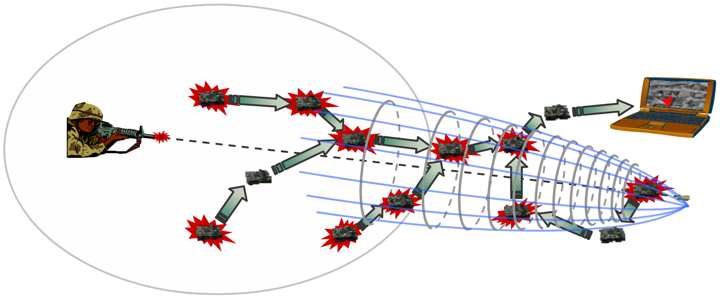
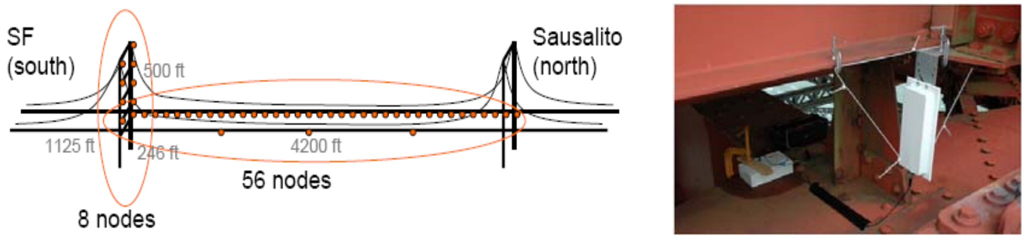


Figure 2.6 Shooter localization in urban terrain [39]

Roots of WSN based applications can easily be found in military specific projects. Therefore, one can find many WSN applications related to military, such as surveillance, target identification, etc. Counter sniper system in urban terrain is another example that is based on event detection model and is shown in Figure 2.6 [39, 40]. In this particular application, deployed WSN is used to pin point snipers that are a big danger for military operations. There is another project currently running for US army where researchers are trying to enable the use of WSN for combat troops which helps them identifying and tracking enemy targets in order to survive with less ammunition [41].

2.1.4.4 Structure health monitoring

Buildings can face different incidents during their long-term use, such as earthquakes, hurricanes, fires, etc. In order to perform structural health monitoring of a building sensor nodes can be placed at various locations in its structure to monitor two discriminating factors. One is the time scale of change which represents long term monitoring (periodic data collection) and second is severity of change (event detection) which comes into play in case of earthquake or explosion [42].



(a): Bridge structure (b): Node deployment

Figure 2.7 Golden Gate Bridge monitoring using WSN [42]

Figure 2.7 presents a real world WSN deployment for structural health monitoring where sensor nodes are deployed on the Golden Gate Bridge in San Francisco Bay to monitor its health over a period of time. Sensors used in this class of application are of high quality and expensive in order to produce precise data as faulty readings in such applications can lead to great disaster. Other examples from this category of applications can be found in [43, 44].

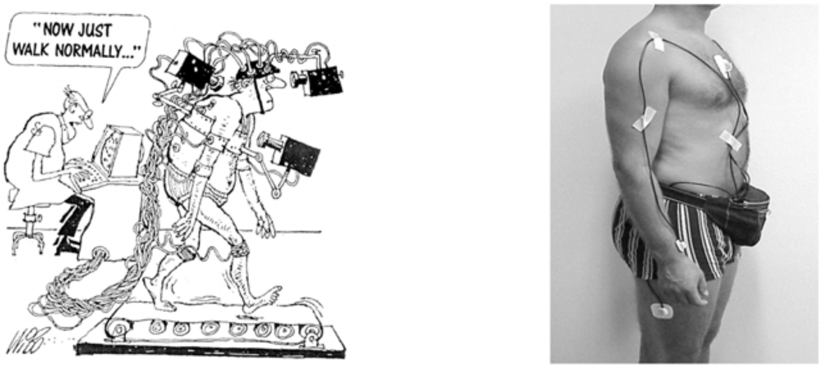
2.1.4.5 Smart compounds

A smart compound is comprised of objects that have sensor nodes embedded in them. Based on the sensed values from the environment these nodes perform certain predefined tasks. Consider a smart room where WSN carry out all the desired operations depending on your position in the room. For example, when you sit on a sofa in front of a TV the WSN automatically switches on the television. If one is sitting on a study table, the light intensity in the room reduces, then the table lamp switches on automatically to keep light suitable for reading. Basic idea of a smart compound is to make life easy for the users, which is the reason for its growing demand. Currently commercial solutions of WSN have a market share of $460 million, which is expected to go up to $2.5 billion by 2011.

“Smart Environment: Residential Laboratory” [45] “Smart Kindergarten” [45] and “the intelligent home” [46] are some examples of the work done in this area. All these applications are based on event detection application type.

2.1.4.6 Medical science

Few years ago, health monitoring involved wiring of sensors on arms and legs of the patient which severely affected their mobility as shown in Figure 2.8(a). Now the use of WSN has revolutionaries the way doctors monitor their patient as shown in Figure 2.8(b).



(a): Old age medical science (b): WSN based health monitoring

Figure 2.8 Revolutionizing medical science [47]

Some examples in this area are: Jovanov et al [48] who presented intelligent WSN and Personal Area Network based solution for health monitoring where nodes are responsible for both signal processing in hierarchical network and data transmission to a personal server that is responsible for logging the delivered information. Gao et al [49] have presented their Advanced Health and Disaster Aid Network (AID-N) that is comprised of wearable sensor nodes. Nodes attached to the body sense the desired values and save them on a server. If the sensed value differs from the standard reading then an alert is sent to a concerned authority. Moreover, a secure web portal is included for physicians to share real time information. Otto et al [50] have taken this one step ahead by developing a prototype system for health monitoring at home. The system consists of a wireless body area network (WBAN) and a home server. WBAN is responsible for periodic data gathering and transmitting it to a home server. Home server then time stamps the reported reading and either saves it on a local database or sends it to a remote medical server placed in some health facility.

2.1.4.7 Transportation and Logistics

Transportation of perishable goods and food stuff has become a very important branch of logistics. Since it is very time consuming and costly process to handle very large quantities of goods produced every hour in large factories, a WSN based solution for automated logistic control and management is a viable solution [51-53]. In these applications a node is attached with the item that is to be monitored, such as luggage moving on a conveyor belt. This attached node can be active where it can communicate with the other devices and keep a history of its mobility. Or the node can be passive such as RFID that are often used to handle luggage at the airports. The devices positioned along the conveyor belt read these RFIDs thus position and place of each item in the systems can be tracked easily at any given time.

2.1.4.8 Precise agriculture

Recently some researchers have tried to address question like what can development in sensor, ICT knowledge and technology mean for farmers, animals and their environments. In pursuit of answer to this question some WSN based applications were developed for management of the fields. Nodes capable of measuring moisture in the air, soil composition and pasture are deployed in the field, which collects desired information and send it to a base station that can be a desktop computer. So, a farmer sitting in his home can have precise information about the state of his field and make vital decisions. Some examples in this regard are [54-56].

2.1.5 FACTORS INFLUENCING SENSOR NETWORK DESIGN

This section discusses various factors that can influence WSN design such as fault tolerance, scalability, production costs, sensor network topology, Quality of Service (QoS), exploiting the tradeoffs such as energy-latency and programmability. These factors are important because they serve as a guideline to design a protocol or an algorithm for sensor networks. In addition, these influencing factors can also be used to compare different schemes [57].

2.1.5.1 Fault tolerance

A sensor node is a tiny sensitive device exposed to harsh environmental conditions during WSN deployment. For the duration of its operation, a node can experience different types of failure such as sensing device being damaged or battery reaching critically low levels. This results in a non-reporting or corrupt data reporting from the sensor nodes. However, these failures should not affect the overall performance of the network. Thus, fault tolerance is the ability of a WSN to sustain its function without any interruption even when nodes are failing or developing some faults. In real time applications such as temperature monitoring of a heating turbine or CO_2 monitoring in a room etc., deployed WSNs must be robust enough to identify the faulty node and exclude wrongly reported data from the received readings, as failing to do so can lead to large-scale catastrophe. F. Koushanfar et al. present in detail various types of faults that can occur in a WSN and also summarize various solutions developed to address these faults [58].

2.1.5.2 Scalability

The hardware requirement of a WSN application varies from few nodes to hundreds and even thousands of sensor nodes. Such variation in the size of sensor field makes application development a complex task. For example, sensor node density in a WSN is highly dependent on the application type and deployment strategy of the nodes (can be calculated as follows, $\mu(R) = (N\pi r^2)/A$ where N is the number of scattered sensor nodes in the area A; and r is the communication range of the nodes [59]).

In the case of machine diagnosis applications the node density is around 300 sensor nodes per 5x5 m^2 area, and the density for the vehicle tracking application is around 10 sensor nodes in the same area [60]. In general, the node density can be as high as 20 sensors nodes/m^3 e.g. in a home where many home appliances can contain sensor nodes [61]. For remote area monitoring application, ad-hoc deployment of the sensor nodes leads to variable node density that also varies over space and time. The number of sensor nodes in these applications ranges from 25 to 100 per region [62]. This requires that the algorithm must be robust enough to handle all these situations.

2.1.5.3 Production cost

Sensor networks usually consist of a large number of sensor nodes as a result, cost per node is very important to justify the overall cost of the network. Currently, researchers have targeted the cost of less than US $1 for a PicoNode. On the other hand, state-of-the-art technology allows a Bluetooth radio system to be less than $10 [63]. Thus, the cost of a low cost communication device such as Bluetooth radio is 10 times higher than the targeted price for a sensor node. In addition, the node may also be equipped with a location finding system, mobilizer or a power generator leading to further increase in the cost. As a result, controlling the cost of a sensor node becomes very challenging task.

2.1.5.4 Routing mechanism

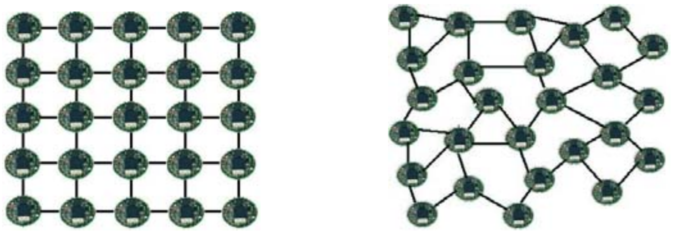
Routing in WSN can be performed in two possible ways that are as follows,

Single hop routing: One way is to establish a direct connection between the sensor node and the sink, where nodes transmits data directly to the sink over single hop. However, such a solution is only possible for small fields or fields having powerful sensor nodes with huge battery and long communication range.

Multi-hop routing: In multi-hop routing data generated at the node takes multiple hops to reach the sink. Such a strategy is highly applicable in WSNs where a sensor node can also act as a relay node, foregoing the need of deploying any additional nodes. While multi-hop routing is an evident and working solution to overcome the problems of large distances or obstacles, it also claims to improve the lifetime Φ^{WSN} of a WSN. The intuition comes from the fact that attenuation of a radio signal is quadratic in most environments therefore it consumes less energy to use relays instead of direct communication.

2.1.5.5 Sensor network topology

Deployment strategy in WSN is highly influenced by the characteristics of the deployment area. For example, in an indoor application, nodes can be placed precisely at strategically critical positions, or in the case of precise agriculture applications, nodes can be placed in the field at predetermined positions in the form a grid as shown in Figure 2.9(a). On the other hand, in applications such as environment monitoring, military surveillance, etc. it is not possible to carry out precise deployment of the nodes due to either terrain particularities or life threatening situations. Therefore, sensor node deployment is carried out by dropping nodes from an airplane or cannon fire that leads to random distribution of the sensor nodes in the field, as shown in the Figure 2.9(b). Large number of nodes, which are also prone to failures, make topology management a difficult task. On the other hand, network itself can also experience varying task dynamics, as they may be the target of deliberate jamming. Therefore, sensor network topologies are prone to frequent changes after deployment. Thus, the developed application should be capable of handling these changes for smooth operation of the network. [57]



(a): Grid deployment (b): Random deployment

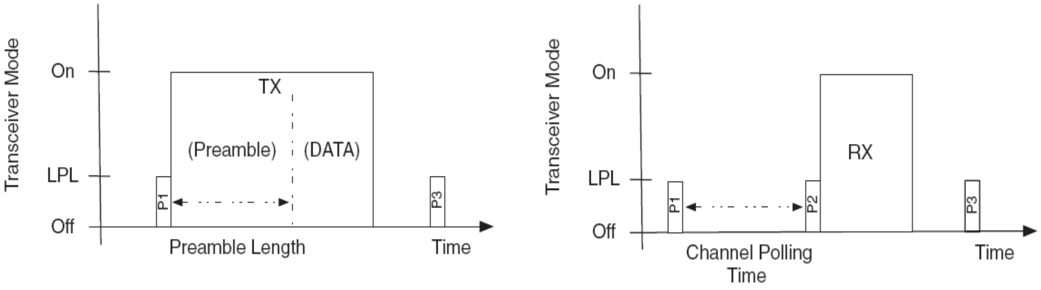
Figure 2.9 Sensor node deployment strategies

2.1.5.6 Shared wireless medium for communication

In WSN, all nodes share a common wireless medium for communication that leads to issues such as packet collision, overhearing, idle listening etc. This causes increased energy dissipation from the nodes. Researchers address these issues by designing a medium access control (MAC) protocol that provide a mechanisms for collision free data exchange amongst sensor nodes. Two famous MAC protocols are carrier sense multiple access / collision avoidance CSMA/CA and BMAC protocol.

In CSMA/CA, when a node wants to broadcast a data packet it first listens to the channel for some fixed amount of time. If the channel is sensed "idle" means no activity is observed then the the node is allowed to transmit. However, if the channel is "busy" the node defers its transmission for random amount of time thus, resulting in reduced probability of data collision.

By default Micaz sensor nodes are installed with BMAC protocol and they adopt following procedure to the communication. The source node uses the period labeled *P1* to perform carrier sense using low power listening (LPL). If the medium is free, the source node must send a preamble equal to the length of the channel polling time to assure the destination node polls the medium during the preamble as shown in Figure 2.10(a). Destination node enters its second polling period (which starts at *P2*, while *P3* marks the third polling period) and senses communication on the medium. The node then turns its transceiver on and receives the remainder of the preamble and the data transmission as shown in Figure 2.10(b) [64, 65].



(a): Source node

(b): Destination node

Figure 2.10 Data exchange in BMAC protocol [65]

2.1.5.7 Quality of service (QoS)

Resource constraint nature of sensor nodes makes WSN very different from traditional computer networks. Unbalanced traffic mix, data redundancy, network dynamics, energy issues, limited processing power, limited storage, small communication range are some characteristics of WSN that makes QoS assurance very challenging.

In WSNs, quality of service has mostly been studied in the context of certain functional layers or application scenarios. For example, in vision-based applications it is required to provide "better" services rather than "best effort" services, similarly in environmental monitoring applications the pivotal QoS criterion is the energy efficient utilization of the sensor network. On the other hand, in case of real time applications it is the end-to-end data latency, which determines the QoS of a WSN. As a result, quality of service in sensor networks remains largely open for research. [66]

2.1.5.8 Exploiting the trade offs

Due to the resource constrained nature of the nodes WSNs have to rely on exploiting various trade-offs between mutually contradictory goals to obtain desired performance from the network. One example of such trade off is the lifetime Φ^{WSN} of WSN and end-to-end data latency in a network.

What is the lifetime Φ^{WSN} *of a WSN?*

There are many definitions for the lifetime Φ^{WSN} of a WSN depending on the application scenario such as,

- The time until the first node fails (or runs out of energy)
- The time until the network is disconnected in two or more partitions
- The time until 50% of the nodes have failed
- The time when for the first time a point in the observed region is no longer covered by at least a single sensor node

In view of given definitions, efficient utilization of the energy resource at each sensor node is critically important for prolonged lifetime Φ^{WSN} of the WSN. To achieve this in practice various energy efficiency schemes have been developed such as data aggregation [67-69] and duty cycling of the nodes [70], mobile sinks, etc. Although these schemes help to decrease energy dissipation and increase lifetime Φ^{WSN} of the network, they also lead to increased end-to-end data latency $\overline{\Delta}$. In case of data aggregation, increased latency is due to time spent in collecting data at an aggregating node and executing the process of aggregation. On the other hand, in duty cycling of the sensor nodes delay is due to long sleep period of the sensor nodes. *What does it mean?* Sensor nodes are generally equipped with a small battery that often cannot be replaced or recharged. Therefore, in order to use this scarce resource efficiently in [70] authors proposed four different operating modes for the sensor nodes as shown in Figure 2.11. In the following a brief description of each of the operating modes extracted from [70].

- **On Duty:** All the components of a node are on. This is the least energy efficient state of a node. In this state a node is called *active*.

- **Sensing unit On-duty:** At least one sensing unit and the processor are turned on but the transceiver is switched off. Therefore, a node can sense the environment and process the results but cannot transmit the data.

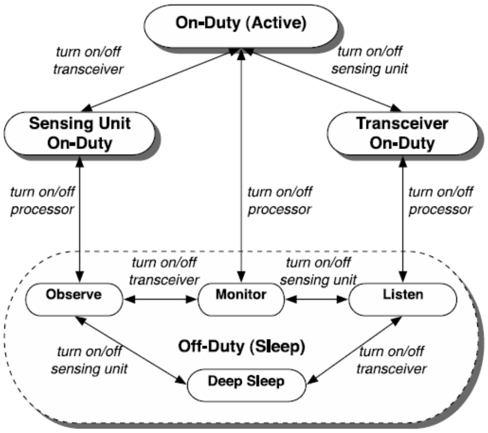


Figure 2.11 Transition between different sensor modes [70]

- **Transceiver unit On-duty:** Transceiver unit and processor are on but all the sensing units are switched off. So a node can transmit and receive message but cannot perform sensing.
- **Off-duty:** The sensor processor is switched off but a timer or some other triggering mechanism is running to wake up the sensor node. This mode is also called *Sleep* mode.

Therefore, 1% duty cycle in this case means that a node awakes for *t* intervals and sleeps for the next *(t*99)* intervals in a deterministic manner. It is known from [65] that in order to achieve the duty cycle of 35%, 11.5%. 10%, 5.61% the polling time in BMAC must be configured to 25ms, 85ms, 100ms, and 185ms, respectively. In order to achieve the duty cycle close to 1% we assume the polling time equals 240ms.

Here the question arise, if sensor nodes are programmed to operate at such a low duty cycles (1%) then what is the probability that a data packet from a node will ever be received at next routing node. In order to address this (and other similar) issue various clock synchronization algorithms [71-73] were developed by the reachers, which can be used to synchronize the active and sleep phases of the nodes. As a result, end-to-end data delivery from source node to the sink can be guaranteed. However, implementation of these synchronization algorithms causes additional communication overhead at the nodes that result in increased energy dissipation. Apart from clock synchronization schemes Gu et al, presented a dynamic switching based forwarding (DSF) scheme that optimizes the expected data delivery and communication delay in extremely low duty cycled sensor field [74] . DSF assumes that each node knows the wakeup times of its neighboring nodes, which can be achieved using MAC layer time-stamping technique. This stamping technique achieves 2.24μs accuracy with an overhead of a few bytes of packets exchange among neighboring nodes for every 5 minutes. When a node wants to transmit a data packet, it can schedule its wakeup time according to the expected receiver node hence ensuring end-to-end data delivery.

Some other examples for such trade-offs are: higher energy expenditure for higher accuracy; longer lifetime Φ^{WSN} of the entire network trades-off against lifetime of individual nodes.

2.1.5.9 Programmability

In highly dynamic environments, data requirement from the sensor field changes over the period of time thus, the network must be flexible enough for remote reprogramming of the sensor nodes to assign new tasks according to the changing application requirements.

2.2 FOCUS OF THE THESIS

In can be inferred from our discussion in Section 2.1 that there is huge diversity not only in available types of sensor nodes and sinks but also in the application areas of the WSN (such as environmental monitoring, smart kindergarten, medical science, etc.). This makes it impossible to state a single set of requirements that clearly classifies all WSN. Therefore, in this thesis we focus our discussion on large-scale applications of WSNs such as, environmental/remote area monitoring. In the following, we discuss some particular issues in these considered application scenarios that leads to build our case for the thesis.

2.2.1 WSN INITIALIZATION SCHEMES

Large-scale applications typically require ad-hoc and dense sensor node deployment. However, large number of nodes and budgetary constraints often result in the selection of low cost sensor nodes that are capable of providing only limited information (e.g. temperature, humidity etc.) about its deployment area. On the other hand, in order to draw some useful conclusions about the field in addition to the parameter of interest some supplementary piece of information is also required from the nodes. For example, consider a case where it is not possible to equip each node with a GPS. As a result, nodes do not know their position coordinates in the field. Now a node senses an event (such as enemy tank has entered in the restricted area) and reports it to the sink. Since the reporting node does not know its position coordinates therefore it can only report partial information that is some intrusion has taken place in the observed region. As the position of the reporting node is not known, the location of the intrusion cannot be identified.

"The process by which nodes in the field determine their position coordinates according to some local or global position coordinate system is called localization. Or in simple terms, localization is a mechanism for discovering spatial relationships between nodes" [75]. However, how we can enable each node to acquire its position coordinates using some WSN initialization schemes in a cost effective and energy efficient manner (without using GPS) is still an open question.

Similarly, ad-hoc deployment of the sensor nodes leads to unknown global topology of the network that is of great importance to both sensor network applications and the implementation of networking functionalities. Especially information about the boundary of a WSN can be very helpful in tracking the number of events entering or leaving the system, to determine the total coverage area of the sensor field, etc. However, due to ad-hoc deployment of the nodes it is not possible to predetermine the boundary of a WSN. In order to acquire this knowledge nodes have to execute some *boundary identification algorithm* to identify the edge nodes and the boundary. Currently, much of the state-of-the-art boundary identification algorithms are either based on message flooding (leading to reduced lifetime Φ^{WSN} of a WSN) or have some special hardware requirements (increased deployment cost). Authors in [76] divided the current state-of-the-art for boundary identification of a WSN into three main classes,

- **Geometric methods:** These methods are based on geographical location of the nodes. The algorithm first identifies a node located at the boundary of a WSN (e.g. the node having maximum value of the y coordinates in the field). It then identifies the neighboring boundary node amongst the neighboring nodes of the current node based on their position coordinates. This process continues until the complete boundary of the WSN is discovered.
- **Statistical methods:** This class of algorithm makes assumptions about the probability distribution of the sensor deployment. The main idea in these algorithms is that nodes located in the center of the sensor field have higher neighbor node density as compared to the nodes positioned close to the boundary of the sensor field.
- **Topological methods:** These algorithms are based on connectivity information amongst the sensor nodes. Basic idea in these algorithms is to select a root node to perform flooding in a sensor field. Then based on hop count measurements from the root node, edge nodes can be identified that can be linked to obtain the boundary of a WSN.

It can be inferred from the above discussion that after its deployment a WSN has to undergo a series of initialization tasks to gather all the required information (localization of the nodes, boundary identification etc.) necessary for desired operation of a WSN. Furthermore, It has been observed that currently available initialization algorithms are either very expensive to implement or lead to high-energy dissipation resulting in reduced lifetime Φ^{WSN} of the WSN.

2.2.2 ROUTE ESTABLISHMENT AND DATA ROUTING IN WSN

Remote area monitoring applications of WSN are usually comprised of large number of tiny sensing devices, which are deployed in ad-hoc manner (spreading them randomly out of an aircraft or a moving land vehicle) over geographically wide areas. Such a random and uncontrolled deployment results in unknown network topology. Furthermore, during its lifetime the WSN usually possesses a dynamically changing topology caused by battery-drained nodes, node/link failures, nodes join/leave the network and in many cases due to node mobility. Such a dynamic environment along with limited battery power, constrained storage capacity and low bandwidth necessitate that each node must always know an energy efficient, low latency and congestion free routing path to the sink. However, ad-hoc deployment of the nodes restricts programmers from pre-configuring routing tables at the sensor nodes.

In the case of slow-changing topologies, typical of most sensor networks, a proactive routing approach seems efficient where network topology discovery is based on the periodic broadcast of a beacon signal from the sink to the whole network. In the simplest case, a beacon is used to flooded the sensor network, the flooding message contains a hop-count that is incremented at each hop, allowing all the nodes to learn how far away from the sink they are and which one of their neighbors will be their parent (upstream neighbor) [77].

2.3 PROBLEM IDENTIFICATION

Information about the boundary of a WSN is one of the most important requirements in many of the remote area / habitat monitoring applications. For example, in the battlefield surveillance scenarios, report from the base station that “none of the enemy’s tanks have entered the region of interest (ROI)” can be misleading without information about the coverage area of a WSN. Furthermore, according to [78] information about the boundary of a WSN can be used to aid following operations of WSNs.

Routing: If boundary of a WSN can be identified beforehand then routing in a WSN can be very efficient, especially geographic routing [79]. Reason is that with known boundary of a WSN one can identify strategically optimal position for a static sink in the field, which leads to minimum data latency as well as minimum energy dissipation from the WSN.

WSN deploying or repairing WSNs: Information about the boundary of a WSN can be used to identify coverage area of a WSN. WSN applications that use ad-hoc deployment of the sensor nodes can utilize the information about the boundary of a WSN to achieve desired coverage during both the WSN deployment and repairing (required due to sensor node failures).

It can be inferred from the above discussion that information acquired during WSN initialization can be used to optimize the performance of a WSN during its operation (such as, data routing). In this thesis, we work to exploit the interdependence amongst WSN initialization and operational requirements. We start with an initialization requirement that is boundary identification of a WSN, and then used this information to develop an energy efficient and robust routing protocol for WSN. Our research revolves around one simple point, how we can exploit a mobile sink to minimize the communication overhead (energy efficiency) from the nodes during WSN initialization and operation.

In the following, we enlist the performance metrics, which we consider during the evaluation of our boundary identification algorithm and data routing protocol compared to state-of-the-art.

2.3.1 EVALUATION METRICS

Total energy dissipation 'E' of a WSN and *completion time 'T' of the algorithm* are two performance metrics, which we will analyze during the analysis of our newly developed boundary identification scheme compared to the current state-of-the-art.

Whereas,

E = Total energy dissipation of a WSN $=\sum_{i=0}^{N} e^i$, e^i is the total energy dissipation by node *i*, N is the total number of nodes in the field

T = Time taken by given algorithm to completely identify the boundary of a WSN

In order to analyze the impact of routing scheme on the behaviour of individual nodes we analyzed two metrics that are lifetime Φ^{WSN} of a WSN and average end-to-end data latency Δ^i experienced by the data packet from each node in a WSN. Whereas,

Φ^{WSN} = time until the first node fails (or runs out of energy)

Δ^i = sum of the time taken by all the data packet from node *i* to the sink / # of data packets

Then in order to get a broader view on the performance of the routing schemes we analyzed average energy dissipation Ψ per node and average end-to-end data latency $\overline{\Delta}$ caused by different routing schemes (refer to Chapter 5 for details). Whereas,

$\Psi = \overline{E} / N$, $\overline{E} = \sum_{i=0}^{N} \overline{e}^{i}$, $\overline{e}^{i}$ is the average energy dissipation by node i

$$\overline{\Delta} = \sum_{i=1}^{N} \Delta^{i} / N$$

2.4 ROLE OF THE MOBILE SINK

During the past few years, the use of a mobile sink in WSN has gained much popularity because of its advantages in terms of prolonged lifetime Φ^{WSN} of the network and increased sensor field coverage compared to the case of a static sink. In the following, we discuss some of these gains.

- Due to scarce energy supply at a sensor node, WSN architecture should be such that it maximizes the lifetime Φ^{WSN} of a network. However, in case of multi-hop routing a sensor node spends most of its energy in relaying data packets. Thus, if we can shorten the distance that a packet has to travel from source to sink then the routing load at each sensor nodes can be reduced resulting in high-energy gains. One way to achieve this is to deploy a mobile sink instead of a static sink. Mobile sink will be responsible for traversing the entire field to collect data from the nodes; each node will transmit data only when the sink arrives within one hop distance to it. Hence, by avoiding multi-hop routing, large gains in the lifetime Φ^{WSN} of a WSN can be achieved.
- Secondly, in a WSN data streams from the entire sensor field converge at a sink that puts heavy relaying load at the nodes positioned in the vicinity of the sink. As a result, these nodes deplete their energy much earlier compared to the rest of the sensor field causing reduction in the lifetime Φ^{WSN} of the network. In contrast, with sink mobility the neighborhood of the sink gets changed frequently and data relaying load gets distributed evenly amongst the nodes located along the mobility trajectory of the sink. Thus, increased lifetime Φ^{WSN} of the WSN is achieved compared to the case of static sink.
- The problem of data collection from a sparsely deployed sensor field is encountered in many scenarios such as, tracking animal migrations in remote-areas [80], monitoring weather conditions in national parks [81], habitat monitoring on remote islands [82], city traffic monitoring etc. Data transmission over one hop is often not preferred because of the high-energy dissipation (required transmission power increases as the fourth power of distance). In

such setups, mobile sink is responsible of periodically visiting different regions in the WSN for data collection and depositing it to some base station in fixed infrastructure.

It can be inferred from the above discussion that the use of mobile sink/entities permits users to program sensor nodes with shorter communication range r that requires less transmission power. Moreover, use of mobile sinks also leads to sufficient reduction in the average path from source to sink resulting in an increased lifetime Φ^{WSN} of the network and decreased average energy dissipation Ψ per node.

2.4.1 EFFECT OF SINK MOBILITY ON END-TO-END DATA LATENCY

Data routing to a sink (static or mobile) can be performed in two ways, single hop or multi-hop routing (refer to Section 2.1.5).

Single hop routing: In a WSN, each sensor node is equipped with a limited buffer that can be used to store the collected data. Considering this fact, researchers came up with the idea of utilizing a mobile sink for data collection by traversing the entire sensor field while sensor nodes are programmed for reactive data transmission. Thus, each node will retain the sensed data until the sink arrives within single/limited hop distance from it and request the data transmission. This removes the route management overhead observed in the case of multi-hop routing. However, time taken by the sink to traverse the entire field will be huge due to slow mobility speed that results in huge end-to-end data latency $\overline{\Delta}$. In addition, in the case of random sink mobility it cannot be guaranteed that the sink will be able to reach every node in WSN as a result data loss can occur. In order to overcome this problem the concept of controlled sink mobility was introduced (refer to Section 4.3.2.3).

Multi-hop routing: In multi-hop routing, each node maintains one or more routes to the sink and data transmission is active from the nodes (forwards data towards the sink and do not wait for data request). If the sink is mobile then the routes require frequent update, especially when the sink moves to a new location. As a result, end-to-end data latency $\overline{\Delta}$ can be higher in the case of mobile sink compared to the case of a static sink. Moreover, an increased overhead in the form of route update also cause increase in average energy dissipation Ψ per node. However, in order to avoid this increase $\overline{\Delta}$ and Ψ due to sink mobility researchers come up with the idea of hierarchical WSNs where some special nodes are responsible of tracking the position of the sink as well as collecting data from the nodes independent of the position of the sink (refer to Section 4.3.2.1).

2.4.2 CONCLUSION – EFFECTS OF USING A MOBILE SINK IN A WSN

Intelligent sink mobility is of critical importance because delayed visits to some regions in a WSN can result in huge end-to-end data latency $\overline{\Delta}$ and probably data loss. Despite these issues mobile sink based data collection schemes have many attractive advantages such as, increased lifetime Φ^{WSN} of the network and if used with single/limited hop routing sink mobility also results in reduced average energy dissipation Ψ per node compared to the case of a static sink.

CHAPTER 3. BOUNDARY IDENTIFICATION OF A WSN USING A MOBILE SINK

3.1 RESEARCH CONTRIBUTIONS

Several parameters can affect the performance of boundary identification algorithm in terms of accuracy, lifetime Φ^{WSN} and completion time *T*. Nowak et al. [83] talk about two of them. One is the relationship between accurate boundary estimation and spatial density of the nodes. Other is the energy constraints of the sensor nodes. Thus, if the boundary estimation algorithm is based on some sort of message flooding technique, which is the case in most of the current state-of-the-art [76, 78, 84] , then there is a trade-off between boundary accuracy and energy consumption of the sensor nodes. Moreover, flooding based approaches also require certain node density to work efficiently, which is difficult to assure during ad-hoc deployment of the WSN.

In this chapter we present an energy efficient *boundary identification scheme for WSN* using a mobile sink *(MoSBoD)*. To accomplish the desired task we transform the problem of boundary identification into one of edge node identification. We utilize a mobile sink for this purpose, which identifies the edge nodes and then connects them meaningfully to obtain the boundary of the sensor field. Compared to current state-of-the-art for boundary identification (discussed in Section 3.2) proposed scheme has various advantages. Firstly, it is independent of the sensor node deployment, and therefore can be used for fields with very low node density. Secondly, it does not require sensor field flooding which helps in saving the nodes' energy. Thirdly, it does not impose any special requirement on the hardware of the sensor nodes, which makes it cost effective.

In order to evaluate the developed scheme we will answer following fundamental questions:

(i) *How much reduction in the communication overhead required for boundary identification is possible compared to state-of-the-art flooding based boundary identification schemes?*

The answer to this question will establish practical bounds on the energy efficiency improvement that can be achieved, and in turn provide a motivation or lack there-of for performing mobile sink based boundary identification

(ii) *What will be the completion time of the algorithm?*

The answer to this question will establish a practical bound on how delay tolerant a WSN application needs to be in order to get the maximum energy benefit.

Rest of the chapter is organized as follows: Section 3.2 discusses the state-of-the-art for boundary identification, Section 3.3 elaborates on the underlying network model, Section 3.4 presents the boundary identification scheme *(MoSBoD)*, Section 3.5 identifies the shortcomings of the MoSBoD algorithm, Section 3.6 presents *M-MoSBoD* that is an improvement over *MoSBoD* algorithm in terms of reduced completion time of the algorithm, Section 03.7 summarizes the performance analysis, and Section 3.8 concludes the chapter by presenting a summary.

3.2 RELATED WORK

In this section, we divide the state-of-the-art methods for boundary identification into three major categories, communication-based methods, hardware-based methods and perimeter-based methods.

3.2.1 COMMUNICATION-BASED METHODS

Algorithms based on hop count measurements calculated using node-to-node communication or sensor field flooding are called communication-based methods. The following are some examples for these methods,

Kröller et al. [84] have presented an edge node identification scheme for the large scale sensor networks. Their scheme works with the limited capabilities of the sensor nodes to recognize the overall structure of the sensor network topology. The algorithm is based on the identification of flower structure (as shown in Figure 3.1) that not only leads to edge node detection but also results in the identification of natural geometric clusters using only local information without the availability of the node coordinates. In order to setup a flower structure each node performs following steps, collect and store neighborhood graph, then every node decide whether it is the seed of a flower like node f_0 in Figure 3.1, if the answer is yes then it is not an edge node. Each of the identified flowers is then analyzed with the neighboring flowers to identify the edge node in WSN as shown in Figure 3.1. The proposed methodology has two main problems. One, it assumes that the communication graph of all the nodes follow a unit disk graph model (a graph formed from a collection of equal-radius circles, in which two circles are connected by an edge if one circle contains the center of the other circle). This is often not the case in realistic deployments of the sensor nodes because of the channel attenuation. Second, the algorithm requires sufficiently high node density (more than 8) to produce acceptable boundary of the WSN.

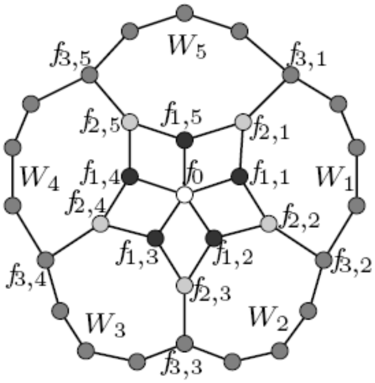


Figure 3.1 A 5-flower structure [84]

Wang et al. [76] proposed an algorithm that determines the edge nodes of a sensor field by utilizing only the neighbor node information at each node. Their scheme does not require any location, angular or distance information from the nodes, also it does not impose any restriction on the communication graph model of the nodes. The algorithm is based on sensor field flooding from an arbitrary root node; then the observation that holes in the sensor field create irregularities in the hop count distances also enables them to identify *cuts* in the sensor field. The cuts are then utilized to determine the holes and their boundaries (called inner boundary). Once the inner boundary is identified then the nodes located at the inner boundary are synchronized to broadcast a message for the identification of the outer boundary. That is calculated by measuring the maximum hop count distances of the nodes from the inner boundary.

Fekete et al. [85] have shown that use of geometric, stochastic and tools from social networks can be used to achieve location awareness amongst the sensor nodes for boundary recognition in geometric sensor networks. They have analyzed the centrality measurement techniques from social networks to show its effectiveness for topology extraction in geometric networks. One particular way for centrality measurement is based on so-called centrality indices, i.e., real-valued functions that assign high values to more “central” nodes, while “boundary” nodes get low values. The idea is based on the fact that the nodes located close to the center of the sensor field have higher centrality than the nodes located near the boundary, provided that the distribution of the nodes follow a suitable random distribution. Once the boundary nodes are identified then they can be connected to form the boundary of the sensor field.

3.2.2 HARDWARE-BASED METHODS

Algorithms based on equipping each sensor node with some special hardware features such as GPS, directional antennas etc., for boundary identification of a sensor field, are called hardware-based methods.

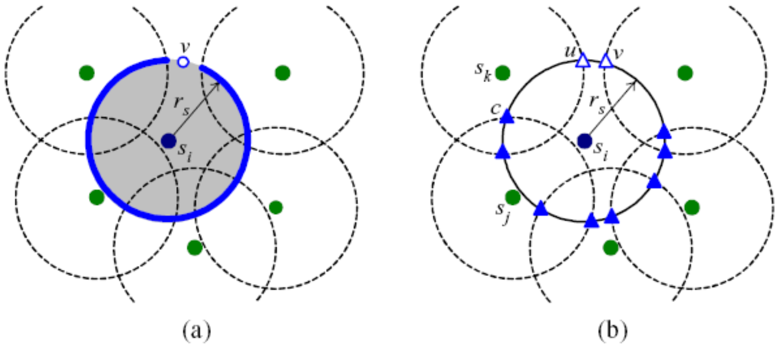
Zhang et al. [78] discuss two algorithms for edge node identification. One is based on a localized Voronoi polygon (LVP) and other on a neighbor embracing polygon (NEP). The LVP based algorithm requires each node to have distance and direction information about all its neighbors, while the NEP based algorithm requires only the direction information of the neighboring nodes. In the NEP based algorithm, each sensor node is required to create a convex hull of its neighboring nodes that is a smallest convex set containing all the neighboring nodes of any given node. Then the given node can be located either inside or outside the boundary of the generated convex hull. If the node is located outside the convex hull boundary then it is an edge node and vice versa. Since both the distance and direction information of the nodes are known in the case of LVP therefore the probability of correctly identifying the boundary node is also high as compared to the NEP algorithm. The major drawback of these algorithms is the high cost, as each sensor node is required to be equipped with distance plus direction (LVP) or only direction (NEP) measuring equipment.

Savvides et al. presented their initial work on the issues involved in identifying the boundary of an region of interest using mobile sensor nodes [86]. They considered a city environment where the mobile sensor nodes are positioned at various locations. On detection of a toxic cloud these mobile nodes are activated and try to form a circular boundary around the toxic cloud. Their solution is based on point-sensing range assumption, which states that a sensor cannot detect the boundary from a distance. It needs to have contact with the boundary in order to detect it. The primary focus of the approach is the selection of a subset of sensors to participate in the boundary estimation process. The selection is based on the proximity of the mobile nodes to other nodes inside the boundary. After the selection is complete, nodes will reposition themselves along the boundary using a distributed coordination algorithm (that is not discussed in the paper). The paper also point out to some open issues involved in building such an application, like connectivity maintenances, efficient motion coordination primitives, node selection, energy/latency awareness etc.

Zeinalipour-Yazti et al. [87] have presented a perimeter algorithm for the identification of the boundary of a sensor field. Prerequisites for the execution of their algorithm include that each node knows its position coordinates along with the position of its neighboring nodes. Then the node having minimum *y* coordinates in the field is determined by flooding or constructing an aggregation tree. The identified node is marked as starting perimeter node, which then selects the neighboring perimeter node by measuring the polar angles of all the neighboring nodes. Two drawbacks of this approach are each node is required to know its position and message flooding.

3.2.3 PERIMETER-BASED METHODS

The first localized boundary node detection algorithm was proposed in [79] based on the information about the coverage of the perimeter of each node's sensing disk. It has been shown that a node is a boundary node if and only if there exists at least one point in its coverage range, which is not covered by any other node Figure 3.2(a)). Based on this criterion, an algorithm with the complexity *O(k log k)* is designed to check locally whether a node is a boundary node, where *k* is the number of neighbors [79]. A crossing coverage checking approach proposed in [88] reduces complexity by checking only special points called crossings on the perimeter. A crossing is an intersection point of two perimeters of sensing disks. A node is a boundary node if and only if there exist at least one crossing, which is not covered by any other node. Figure 3.2(b) shows an example where crossing *c* is covered by the third sensing disk of node s_k. "The problem of perimeter-based approaches is that each node needs to check positions and status of all of its neighbors, which is inefficient when the sensor nodes are densely deployed so that every time when a node dies, all its neighbors need to check the coverage of their perimeters or crossings again" [78].



(a) Perimeter-coverage checking approach proposed in [89]. The solid curve represents the portion of perimeter of sensing disk covered by neighbor nodes. (b) Crossing-coverage checking approach proposed in [88]. Solid and open triangles represent covered and uncovered crossings, respectively.

Figure 3.2 Perimeter-based boundary node detection approaches [78]

3.2.4 SUMMARY

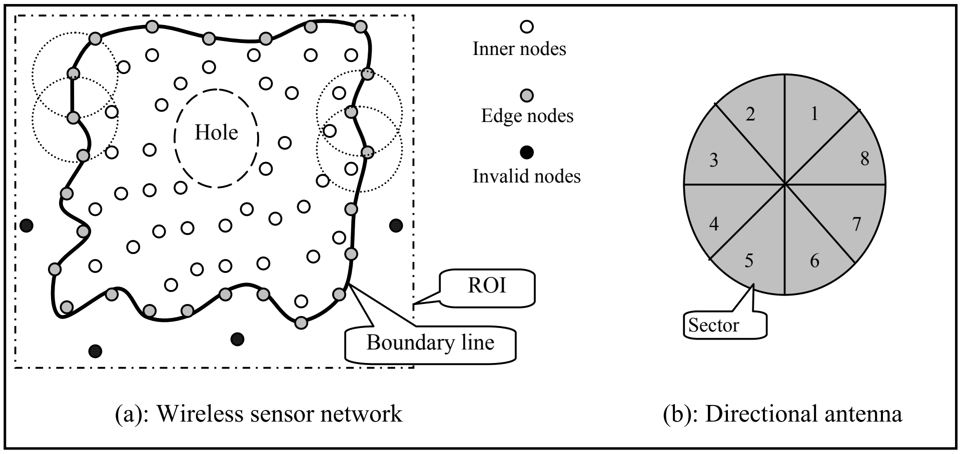
It can be summarized from above discussions that current state-of-the-art for boundary identification either depend on heavy communication amongst the sensor nodes such as, sensor field flooding, or require each node to be equipped with expensive hardware such as, a GPS, directional antenna etc. The first type of schemes results in reduced lifetime of the WSN and second incurs a high budget constraint especially in large-scale deployment of the sensor nodes. These limitations of the available schemes motivated us to investigate other options for energy efficient boundary identification of the WSN with minimum budget.

3.3 NETWORK MODEL AND TERMINOLOGY

This section presents the basic network model, which we assumed for the development of our boundary identification scheme. Moreover, some terminology is also defined which are used in the rest of the chapter.

3.3.1 NETWORK MODEL

It is assumed that we have to monitor an area inaccessible for humans, such as a highly polluted site containing toxic or radioactive materials. However, the terrain of the area is assumed suitable for sink mobility. Nodes are randomly but uniformly distributed in the field. Deployed nodes are assumed to be static and very inexpensive having Omni-directional antennas with same and fixed transmission range which is very small compared to the size of the sensor field. Moreover, sensor nodes have no knowledge of their position coordinates in the sensor field.



(a): Wireless sensor network (b): Directional antenna

Figure 3.3 Network model

After the deployment of the sensor field, each node acquires a “valid” or “invalid” status. The number of its neighboring nodes determines the validity of a sensor node. A node n_i is called *neighbor* of node n_{i+1} if n_i lies within the transmission range of n_{i+1}. It is also assumed that the sensor field contains only one cluster of valid sensor nodes and all the other nodes located outside this cluster are invalid nodes (see Figure 3.3).

Since each node is equipped with limited power supply that cannot be recharged or replaced, therefore sensor nodes are programmed to operate at reduced duty cycle of 1% (refer to Section 2.1.5.8).

On the other hand, we have assumed that the sink is a special node, which is mobile and equipped with an unlimited energy resource, a GPS and a compass used to determine its position and direction of mobility. One example of such a mobile sink is *BigDog* shown in Figure 3.4. It is a DARPA funded research project. This machine is capable of traversing harsh terrain; can move at 4 miles per hour and climb slopes up to 35 degrees [90].

Figure 3.4 BiGDog

It has also been assumed that the sink is equipped with a sectored directional antenna having fixed transmission range equal to that of the sensor nodes. This antenna can be used to determine the angle of arrival (AoA) of a message from a sensor node [91, 92] and to roughly estimate the distance between a node and the sink using RSSI [93, 94]. Furthermore, the sink also knows the *region of interest* (ROI) that is a rectangular region and contains all the deployed sensor nodes. The assumption of an ROI does not affect the generality of our algorithm as it is only used to locate the sensor field by the sink as discussed in Section 3.4.1. In the following, some terminology that is frequently used in this chapter is defined.

3.3.2 TERMINOLOGY

This section states some terminologies that are frequently used.

The *boundary of a sensor field* is a subset of valid sensor nodes with the property that the line obtained by connecting each node in this subset with its neighboring edge node “encloses” all the other valid sensor nodes as shown in Figure 3.3(a).

Edge Nodes (N_{edge}) are valid sensor nodes located at the boundary of a WSN. N_{edge} are connected to obtain a boundary line as shown in Figure 3.3(a).

A *directional antenna model* presented in [95] is considered in this paper. The antenna system is composed of Λ beams such that their intersection is zero and their union covers the entire 360-degree plane as shown in Figure 3.3(b). The width of each beam is equal to $360/\Lambda$ degrees and the area covered by one beam is called a sector. We consider large values of Λ as a result size of an individual sector becomes small.

Edge node position estimation refers to the estimation of the position coordinates of an edge node by the sink. For this purpose the sink utilizes its directional antenna to measure the angle of arrival (AoA) [92] and the received signal strength (RSSI) [94] from the sensor node. Then, based on its current position calculated using GPS, the sink estimates position of the sensor node. However, it has been recognized that RSSI based techniques for distance estimation are cost effective but can also be inaccurate [96].

3.3.3 SENSOR NODE POSITION ESTIMATION PROCEDURE

In this section we will show that even with its inherent inaccuracy in distance estimation, a RSSI based scheme serves our purpose. G. Zhou et al. [96] mentioned the following properties of the radio transmitter of a sensor node which lead to inaccurate distance measurements: *Anisotropy* – the radio signal incurs different path losses in different directions, *continuous variation* – path losses vary with changing propagation direction, *heterogeneity* – difference in hardware and battery power can lead to different signal strength. In order to prove our point we consider two nodes, a mobile node ‘n_m’ (the mobile sink) equipped with directional antenna and RSSI based distance estimation scheme and a static node ‘n_s’. Node n_m is assigned the task to estimate the position of the node n_s. Node n_m transmits a *hello* message and on reception of a response from n_s node n_m estimates the location of n_s (using AoA and RSSI) and moves to it. On reaching the calculated position (which may not correspond to the actual position of n_s), node n_m again transmits a *hello* message to confirm whether the calculated position was correct. Due to inaccuracies in RSSI and AoA calculations node n_m has to compute a new estimate and determines that it still has to move some more distance to reach the node n_s. The process continuous until node n_m reaches node n_s. Thus, even with the inaccuracies in the RSSI and AoA measurements, node n_m will eventually reach the position of node n_s.

Note: The effects of anisotropy and continuous variation get reduced when node m tries to approach s from different directions (because both factors are based on direction of signal propagation).

3.4 MOBILE SINK BASED BOUNDARY DETECTION ALGORITHM (MOSBOD)

This section presents the new algorithm *MoSBoD* for boundary detection using a mobile sink. Bootstrapping and edge node identification are its two main phases.

MoSBoD Algorithm

```
1:  Bootstrapping of the sensor nodes and identification of starting edge node
2:  Edge node identification and boundary traversal using the mobile sink
```

3.4.1 BOOTSTRAPPING PHASE

The bootstrapping phase is an initialization phase of the MoSBoD algorithm. During this phase the sensor nodes prepare themselves for the arrival of the sink by determining their neighbor node density (Γ) and calculating their validity status. Simultaneously, the sink locates a valid starting edge node in the sensor field.

3.4.1.1 Bootstrapping of the sensor nodes

The bootstrapping phase divides the sensor nodes into two groups of valid and invalid nodes. As an input for the bootstrapping phase each node is initialized with a time value t_i (product of the neighbor node density and the time required by a node to send plus receive a message) and the required neighbor density (Γ_{valid}) to calculate its validity status.

The bootstrapping phase for each sensor node comprises the following actions: activate message reception mode, on expiration of time t_i broadcast a message containing own ID. On receipt of messages from neighboring nodes create a list of neighbors containing their ID's and set their validity status equals *false*.

(i) If Γ becomes equal to Γ_{valid} then set own validity equals *true* and broadcast a message containing the ID and newly acquired validity status.

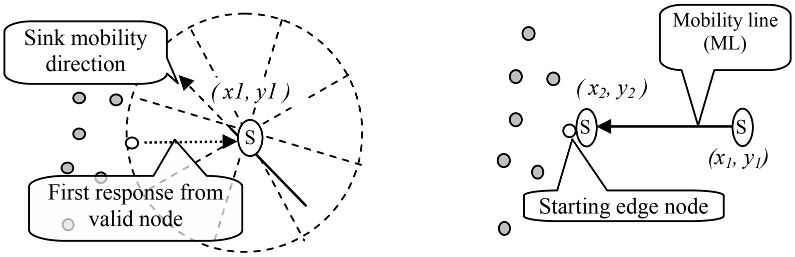
(ii) On receipt of the validity message update the validity status of the sender node.

It can be deduced from the above procedure that under ideal conditions (no message collision) each valid node broadcasts two messages [$M_1(ID)$ and $M_2(ID, validity\_status)$], invalid node broadcasting only one message $M_1(ID)$ and the number of messages received by each node are equal to [number of valid neighboring nodes * 2] plus [number of invalid sensor nodes].

However, during sensor node deployment if one can ensure that for each node $\Gamma \geq \Gamma_{valid}$ then the communication overhead of the nodes can be further reduced. As $M_2(ID, validity\ status)$ which is used to broadcast the validity status of a node will no more be required.

3.4.1.2 Identification of the starting edge node

During this phase, the sink calculates the mobility direction to reach the boundary of the ROI and then locates the starting edge node $(n_{s\text{-}en})$. In order to accomplish this task the sink performs following operations: Determine its current location and alignment with respect to the boundary of the ROI using GPS and compass; calculate mobility direction and move to reach the closest boundary point of the ROI.



(a) (b)

Figure 3.5 Identification of the starting edge node

Upon reaching the boundary of the ROI, sink calculates the center of the ROI (utilizing ROI coordinates), switches on its antenna, starts broadcasting a *hello* message and begins to move towards the center of the ROI. The sink continues until a response [$M_{resp}(ID, validity)$] from a valid sensor node is received. On receipt of a response message, sink marks the responding node as a starting edge node and saves own current coordinates as (x_1, y_1) (see Figure 3.5(a)). Also, by utilizing the AoA of the received response and the *node position estimation* procedure explained in Section 3.3.3, sink calculates and moves to the position of the $n_{s\text{-}en}$ that is stored as (x_2, y_2). Exceptional situations, such as when multiple valid nodes respond to the sink, are also handled in the pseudo code of Module-3.1. Moreover, it should be noted that unlike [87] where the starting node is the node with minimum *y* coordinates (calculated using GPS) and identified by sensor field flooding, Module-3.1 does not impose any such requirements. Its execution will produce following outputs: coordinates of the location when the sink receives first response from a valid sensor node (x_1, y_1); coordinates of the starting edge node (x_2, y_2), and starting edge node ID.

Module-3.1: Locating the ROI and the starting edge node

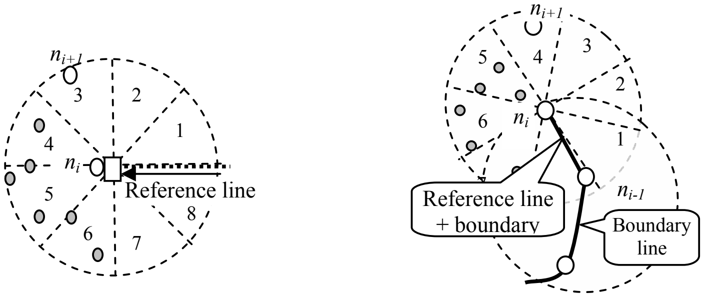
```
INPUT: ROI coordinates, $n_{s-en}$ = null
Calculate and move to the nearest boundary of ROI using ROI coordinates and own position (calculated using GPS)
// Sink moves inside the ROI in search of valid sensor nodes
while $n_{s-en}$ == null
  Move towards center of ROI, broadcasting hello message
  if $M_{resp}$ is received from only one node AND validity == true then
    $n_{s-en}$ = respondingNodeID & $(x_1, y_1)=(x_{sink}, y_{sink})$
  else if $M_{resp}$ is received from multiple nodes then
    if responding nodes are at same shortest distance from the sink then
      $n_{s-en}$ = node with minimum ID
    else $n_{s-en}$ = node at shortest distance from the sink        // (Calculated using RSSI based
                                                                       // distance estimation)
    end if
  end if
  if ($n_{s-en}$ !== null)
    Utilize received response from $n_{s-en}$ to estimate $(x_{s-en}, y_{s-en})$
    Set $(x_2, y_2) = (x_{s-en}, y_{s-en})$ and move to the calculated location of the $n_{s-en}$
  end if
end while
```

3.4.2 EDGE NODE IDENTIFICATION AND BOUNDARY TRAVERSAL

This section presents Module-3.2 that enables the sink to identify the neighboring edge nodes of a current node (node where the sink is currently positioned). The line then obtained by connecting all the identified edge nodes with their corresponding neighboring edge nodes is the desired boundary of the sensor field.

Module-3.2 is based on the use of mobility and a directional antenna by the sink. Prerequisites for the execution of this algorithm are, the sink is positioned at an edge node n_{en^i}, it knows the position coordinates of the current edge node (x_{en^i}, y_{en^i}) and position coordinates $(x_{en^{i-1}}, y_{en^{i-1}})$ of the identified neighbor edge node $n_{en^{i-1}}$ of the current node.

The sink initiates execution of Module-3.2 by calculating the reference line that is defined as the line obtained by joining the position coordinates of the current node n_{en^i} and the identified neighboring edge node $n_{en^{i-1}}$. Then the sink numbers the sectors starting from the one located beside the reference line towards the mobility direction of the sink. We specify that the sink traverse the boundary of the sensor field in counter clockwise direction. In this case, the sink will mark the sector located in counter clockwise direction of the reference line as sector 1, shown in Figure 3.6.



(a): Case of starting edge node

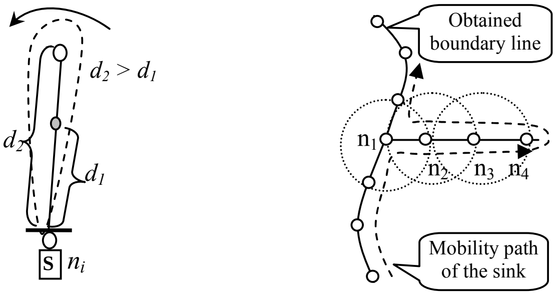
(b): Case of an arbitrary edge node

Figure 3.6 Neighboring edge node identification

Once sectors are numbered, the sink broadcasts a *hello* message to the neighboring nodes of the current node. The node whose response is received in the lowest sector number is assigned the status of next edge node $n_{en^{i+1}}$ by sending an *edge node confirmation message*. For example, in Figure 3.6(b) response from node n_{i+1} is received in sector 4 while the responses from all the other nodes are received in sectors having ID greater than 4. Therefore, node n_{i+1} is assigned the status of next edge node. Moreover, position coordinates of node n_{i+1} are estimated using *node position estimation* (discussed in Section 3.3.3) and the sink moves to the position of newly identified edge node.

On reaching node $n_{en^{i+1}}$, the sink again executes Module-3.2 to identify the neighboring edge node of node $n_{en^{i+1}}$. Thus, by moving from one edge node to the next, the sink eventually returns to the starting edge node completing the boundary trace.

The discussion so far leaves one question open: How does the sink determines the reference line when it is positioned at the starting edge node? It is known that the starting edge node is the first node to be identified as an edge node and at this point of time, the sink has no information about the neighboring edge nodes of the starting node. Thus, the starting edge node is a special case for the reference line identification. In this case we utilize the coordinates of the current node (x_2, y_2) and the coordinates (x_1, y_1) (obtained from Module-3.1, where (x_1, y_1) marks the location of the sink where it received first response from the starting edge node) to define the reference line. Since, it is assumed that the sink traverses the boundary of the field in counter clockwise direction, the sink assigns numbers to the sectors in ascending order starting from the one located towards counter clockwise direction of the reference line, as shown in Figure 3.6(a). The rest of the procedure for the identification of the next edge node is the same as discussed above.



(a) (b)

Figure 3.7 Special cases for neighbor edge node identification in Module-2

During the execution of neighboring edge node identification module some exceptions can also arise which are handled in the pseudo code of Module-3.2. For example, if the sink receives responses from two or more nodes positioned in the lowest sector number then it is assumed that the two nodes are located on a line, as the size of a sector is very small. In this case, the node located farthest from the current node is selected as next edge node as shown in Figure 3.7(a). There may also be the case where a group of nodes are connected to the main sensor field via a single link, like node n_1, which connects n_2, n_3 and n_4 with the rest of the field as shown in Figure 3.7(b). Since there is no restriction on the number of times the sink can visit an edge node during the boundary identification therefore our algorithm can also successfully handle such cases.

Module-3.2: Edge node identification and boundary traversal

Input: The sink is positioned at an edge node n_{en^i}. Coordinates of edge nodes n_{en^i} and $n_{en^{i-1}}$ are (x_{en^i}, y_{en^i}) and $(x_{en^{i-1}}, y_{en^{i-1}})$ respectively.

```
do
    Number sectors according to the reference line
    // Identification of the next edge node
    Transmit a hello message to the neighbors of the current node $n_{en^i}$;
    if only one node $n_{resp}$ responds then
        $n_{en^{i+1}} = n_{resp}$;
    else if multiple responding nodes are located at same farthest distance from sink then
        $n_{en^{i+1}}$ = node having minimum ID
    else $n_{en^{i+1}}$ = the farthest node; // determined using RSSI
    end if
    Apply edge node position estimation for node $n_{en^{i+1}}$;
    Store position coordinates of node $n_{en^{i+1}}$ and move to it
  Send a message to node $n_{en^{i+1}}$ demanding its list of neighbors;        // edge node confirmation message
    Set $n_{en^{i-1}} = n_{en^i}$;      $n_{en^i} = n_{en^{i+1}}$;
until $n_{en^i}$ != $n_{s-en}$
```

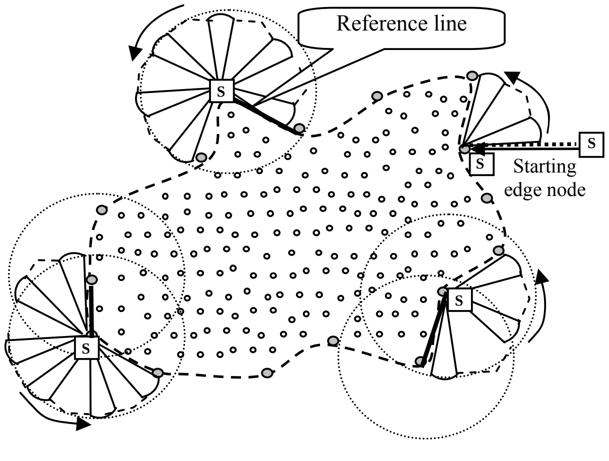


Figure 3.8 Edge node identification

Example. Figure 3.8 shows the implementation of the MoSBoD algorithm on a sample sensor field. Once the boundary is fully identified, the sink can continue its mobility along the boundary line monitoring for possible edge node failures and boundary reconstruction. 2nd and later trips along the boundary line are based on the stored position coordinates of the edge nodes, so they will require less time than the first trip.

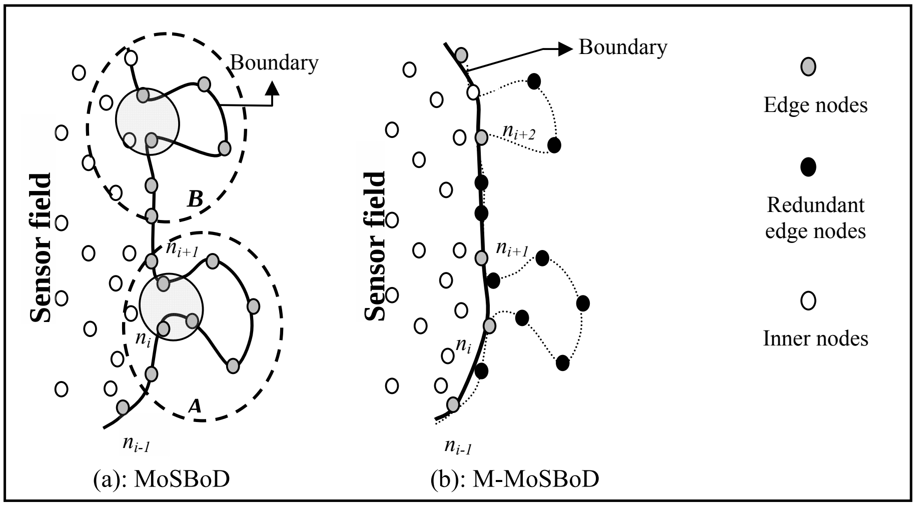
3.5 SHORTCOMINGS IN THE MOSBOD ALGORITHM

Although the MoSBoD algorithm successfully identifies the boundary of the sensor field, the following shortcomings can be identified.

3.5.1 QUALITY OF THE IDENTIFIED BOUNDARY

Currently we defined the boundary as, *"the subset of valid sensor nodes with the property that the line obtained by connecting each edge node with its neighboring edge nodes "encloses" (inner nodes are enclosed by the boundary in* Figure 3.9*(a)) all the other valid sensor nodes"* affects the performance of the algorithm in two aspects.

First, it increases the completion time of the boundary identification process. Figure 3.9(a) presents one such scenario where the boundary is identified using the definition given above. It can be seen that the boundary line contains a number of redundant edge nodes in regions *A* and *B*, which could have been ignored to complete boundary identification in less time, as shown in Figure 3.9(b). Second, execution of each edge node identification procedure causes some energy dissipation, so the larger the number of edge nodes the higher will be the energy dissipation.



(a): MoSBoD (b): M-MoSBoD

Figure 3.9 Obtained boundary shapes using MoSBoD and M- MoSBoD

3.5.2 MOBILITY SPEED OF THE SINK

A mobile sink is usually very restricted in terms of its speed. Therefore, a major shortcoming of the MoSBoD algorithm is that the time required to complete the boundary identification process is rather high, for a realistic scenario 110% compared to both communication-based and hardware-based state-of-the-art methods (detailed analysis can be found in Section 3.7.2). This shortcoming of the MoSBoD algorithm can limit its competitiveness in real world scenarios.

The next section presents an improved version of the MoSBoD algorithm, which successfully addresses mentioned issues.

3.6 MODIFIED MOSBOD (M-MOSBOD)

This section presents the modified version of the MoSBoD algorithm, called M-MoSBoD, which has significantly less completion time *'T'* compared to MoSBoD algorithm at the cost of a negligible increase in the energy dissipation of the sensor nodes.

3.6.1 IMPROVED DEFINITION OF THE BOUNDARY

One possibility to improve the performance of the boundary identification algorithm is to reformulate the definition of the boundary. Therefore, in the following we give a more restrictive definition of the boundary for a sensor field:

"The boundary of the sensor field is the ***smallest*** *subset of valid sensor nodes with the property that the union of their communication ranges either "encloses" (inner nodes are enclosed by the boundary in* Figure 3.10*)* ***or*** *"contains" (redundant nodes are contained in the communication range as shown in* Figure 3.10*) all the other valid sensor nodes."*

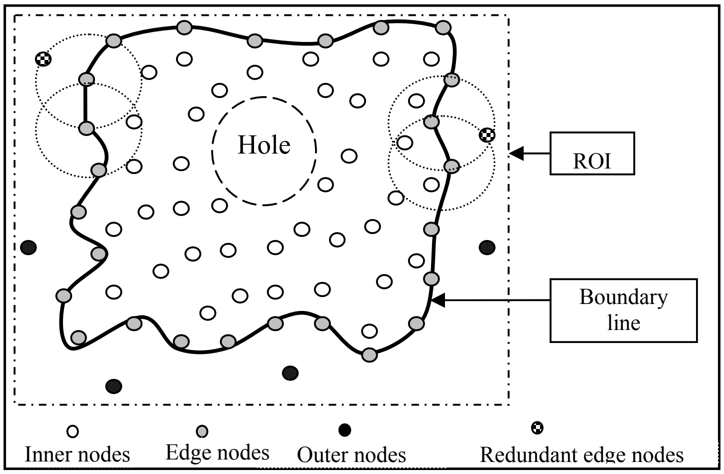


Figure 3.10 Modified boundary definition

This new definition of the boundary is more suitable for wireless sensor networks, where only a rough estimation of the boundary is required in order to retrieve information, such as coverage area of the network, number of events occurring in the network, etc. Thus, the precise marking of the boundary as done in the case of MoSBoD is often not required in sensor networks. In Section 03.7 we will show that this new definition of the boundary leads to much better performance in terms of completion time and lifetime of the network as compared to MoSBoD.

How can we determine the boundary according to the new definition in practice? According to the new definition of the boundary it contains *minimum* possible number of valid sensor nodes. In order to satisfy this condition, we introduce a neighbor *connectivity check,* which avoids the selection of redundant edge nodes by the sink.

Let us assume that the sink is currently positioned at an edge node $n_{en^{i-1}}$, and it has to identify next edge node n_{en^i}. The sink transmits a *hello* message to all neighbors of node $n_{en^{i-1}}$. Each neighbor of node $n_{en^{i-1}}$ is programmed to respond with a message containing the list of its neighboring nodes. The sink identifies that node n_k has responded in the lowest sector number so it should be the next edge node. But the neighbor *connectivity check* implies that node n_k can only be selected as next edge node if it satisfies following condition:

Let NL_{n_k} be the list of neighbors of node n_k,

If the intersection of the two sets $NL_{n_{en^{i-1}}}$ and NL_{n_k} equals NL_{n_k} **then**

> node n_k is a redundant node. Select a node from the set $\{NL_{n_{en^{i-1}}} - n_k\}$ whose response is received in the lowest sector number.

else node n_k is the next edge node.

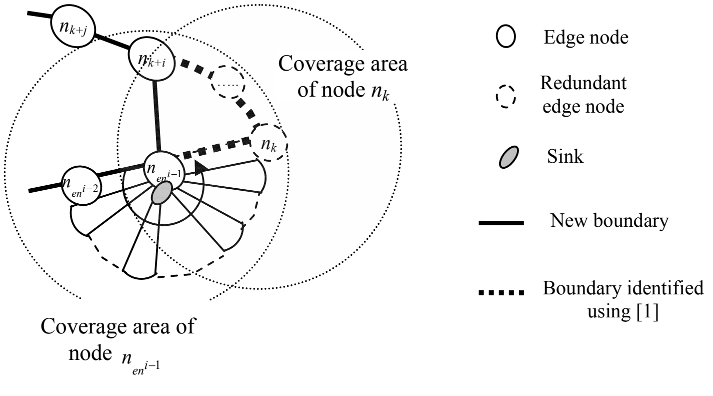


Figure 3.11 Removing redundant edge nodes

The above condition is based on the fact that if the selected node n_k knows at least one node that is not in the neighbor list of the current node $n_{en^{i-1}}$, only then the inclusion of the node n_k in the list of edge nodes will be useful. For example, in Figure 3.11 the sink positioned at node $n_{en^{i-1}}$ receives a response from node n_k in the lowest sector number, but the *connectivity check* shows that the intersection of $NL_{n_{en^{i-1}}}$ and NL_{n_k} is equal to NL_{n_k}. Thus, node n_{k+1} cannot be an edge node. Similarly, *connectivity check* is executed for n_{k+2}, n_{k+3}, ..., until n_{k+i} is found that has a neighbor node n_{k+j} which is not in the neighbor list of node $n_{en^{i-1}}$. Then, the, connectivity check is passed and n_{k+i} is selected as the next edge node.

3.6.2 REDUCING COMPLETION TIME OF MOSBOD

In MoSBoD algorithm, the time required by the sink to identify an edge node can be divided into two parts; the time required for identifying the next edge node and time required to move from the current edge node to the newly identified edge node. Since the speed of the sink cannot exceed a certain value, the only way to reduce the completion time of the MoSBoD algorithm is to reduce the time required by the sink to identify the next edge node. In order to achieve the desired results we exploit the low duty cycling of the sensor nodes. Low duty cycling helps to reduce the energy consumption of nodes, but it also increases the latency in node-to-node communication. Therefore, a possible strategy for reducing the completion time of the MoSBoD algorithm is a selective increase of the duty cycles of sensor nodes. Such an increase in the duty cycle increases the energy consumption of sensor nodes, but we will show quantitatively in Section 3.7.2 that the gains in terms of reduced completion time are very high compared to the decrease in the lifetime of the network. In the following, we discuss this selective increase in the duty cycles of the nodes in detail.

Once the sink selects next edge node $n_{en^{i+1}}$ it transmits an edge node confirmation message to the selected node. The transmitted message also contains a request to the selected edge node $n_{en^{i+1}}$ to increase its duty cycle to 100% along with its neighboring nodes after $t_{i \rightarrow i+1}$ time intervals ($t_{i \rightarrow i+1}$ is the time taken by the sink to move from current node n_{en^i} to the selected edge node $n_{en^{i+1}}$). As a result, node $n_{en^{i+1}}$ broadcasts a message to its neighbors containing a request to operate at 100% duty cycle and a TDMA (time division multiple access) based response schedule. The response schedule is used to avoid message collision when the neighboring nodes of $n_{en^{i+1}}$ respond to the *hello* message from the sink (operating at 100% duty cycle) for the selection of $n_{en^{i+2}}$. Moreover, when node n_{en^i} overhears the *edge node confirmation message* sent to node

$n_{en^{i+1}}$, n_{en^i} reduces its duty cycle along with the neighboring nodes back to 1% (as it was asked to operate at 100% duty cycle when the sink identified node n_{en^i} from $n_{en^{i-1}}$). So, the *edge node confirmation message* acts as an indication for node n_{en^i} that the next edge node has been selected, therefore it can reduce its duty cycle back to 1% along with its neighbors (see Module-3.3). This increase in the duty cycle reduces the communication time between sensor nodes and the sink (no latency due to sleeping nodes). As a result, the time required by the sink to identify the next edge node decreases.

Module-3.3: Modified MoSBoD algorithm (M-MoSBoD)

```
Input: The sink is positioned at the starting edge node $n_{s-en}$.

current node = $n_{s-en}$
do
    Calculate reference line and number sectors according to it
    // Identification of the next edge node
    Transmit a hello message to nodes in $[NL_{n_{s-en}}]$;
    // nodes respond with their list of neigbors
    do
        If a single node $n_i$ responds in the lowest sector number then temp = $n_i$;
        else if multiple responding nodes are located at the same farthest distance from sink then
            temp = node having minimum ID
        else temp = the farthest node; // determined using RSSI
        end if
        // connectivity check
        if Intersection $\{[NL_{n_{en^{i-1}}}],[NL_{n_{en^i}}]\} == [NL_{n_{en^i}}]$ then
            $[NL_{n_{en^i}}] = [NL_{n_{en^i}}] - temp$
        else $n_{en^{i+1}}$ = temp;
    until $n_{en^{i+1}}$ == null
    Apply edge node position estimation for node $n_{en^{i+1}}$;
    Store position coordinates of node $n_{en^{i+1}}$;
    // edge node confirmation message
    Send a message to $n_{en^{i+1}}$, demanding the list of neighboring nodes, and request to operate at 100%
    duty cycle along with its neighbors after $t_{i \rightarrow i+1}$ intervals;
    // When node $n_{en^i}$ hears this message, it degrades the duty cycle to 1% along with the neighbors
    Set $n_{en^{i-1}} = n_{en^i}$;    $n_{en^i} = n_{en^{i+1}}$;
until $n_{en^i}$ != $n_{s-en}$
```

Module-3.3 gives the pseudo code for the M-MoSBoD algorithm with improved boundary and reduced completion time. The procedure assures a tight boundary by avoiding the inclusion of redundant nodes. This helps to reduce the completion time of boundary identification. Furthermore, reducing the number of edge nodes also reduces message exchanges between sensor nodes and the sink, which increases the lifetime of the network.

3.7 PERFORMANCE EVALUATION

This section presents an evaluation of the MoSBoD and M-MoSBoD algorithm based on the analytical findings and OMNeT++ simulation tool. We analyze the communication cost of the MoSBoD and M-MoSBoD algorithm along with the effects of neighbor node density and the size of the sensor field on the accuracy of the boundary identification and the completion time *'T'* of the algorithm.

The basic simulation setup is comprised of an area 800x500m^2, where the sensor nodes are uniformly, but randomly, deployed. We applied the MoSBoD algorithm to sensor fields with random, U shaped, circular and rectangular boundary shapes with varying neighbor node densities Γ (= average number of neighboring nodes for each node) of 4, 6, 14 and 18 nodes. The boundary obtained with M-MoSBoD remains the same irrespective of the change in neighbor node density. In contrast, other flooding based schemes presented in [84], [76] or [96] require a neighbor density of at least 7 to produce acceptable boundaries [76].

In the following sections, we present a comparison of MoSBoD, M-MoSBoD and methods presented in [76] in terms of energy consumption and completion time of the algorithm.

3.7.1 ANALYZING EFFECTS OF REFINED BOUNDARY

In order to analyze the effect of boundary refinement on energy consumption and completion time, we again consider the setup shown in Figure 3.11. It is assumed that the sink is positioned at node $n_{en^{i-1}}$ which has neighbors $n_{k+1}, n_{k+2}, \ldots, n_{k+i}$. The distances between nodes are denoted as follows:

$dist(n_{en^{i-1}}, n_{k+1}) = d_{k+1}$,

$dist(n_k, n_{k+2}) = d_{k+2}$,

$dist(n_{k+(i-1)}, n_{k+i}) = d_{k+i}$,

and, $dist(n_{en^{i-1}}, n_{k+i}) = D$.

Let $t_{edgeNode}$ be the total time required by the sink to send a *hello* message to the neighboring nodes of the current node, receive their responses and select the next edge node based on the received responses. $e_{edgeNode}$ is the total energy dissipation of the current node $n_{en^{i-1}}$ and its neighbors during edge node identification.

It is assumed that we have to identify a section of the boundary from node $n_{en^{i-1}}$ to node n_{k+i} (i is the number of redundant nodes between $n_{en^{i-1}}$ and n_{k+i}). In case of the MoSBoD algorithm, the sink will identify edge nodes $n_{k+1}, n_{k+2}, \ldots, n_{k+i}$ to accomplish the task. Thus, the total energy dissipation from the nodes will be $i * e_{edgeNode}$. On the other hand, in case of M-MoSBoD, the more precise boundary definition and the *connectivity check* avoids the selection of redundant edge nodes ($n_{k+1}, n_{k+2}, \ldots, n_{k+(i-1)}$), and node n_{k+i} is directly selected as the next edge node to accomplish the task. Since the connectivity check is executed by the sink based on already available information (neighbor list of the nodes) it will not cause any additional communication overhead. As a result, the total energy dissipation of the nodes in this case will only be $e_{edgeNode}$ that is $i-1$ times less than the energy dissipation in the case of MoSBoD alogorithm.

Similarly, the time required by the MoSBoD algorithm to identify the boundary section from $n_{en^{i-1}}$ to n_{k+i} is $(i * t_{edgeNode}) + [(d_{k+1} + d_{k+2} + \ldots + d_{k+i})/V_{sink}]$, where V_{sink} is the speed of the sink. On the other hand, for the M-MoSBoD algorithm the completion time equals $t_{edgeNode} + [D/V_{sink}]$. Since D is the shortest path from node $n_{en^{i-1}}$ to n_{k+i} therefore any other path taken to reach node n_{k+i} from node $n_{en^{i-1}}$ (case of MoSBoD) will take more time $(D \leq d_k + d_{k+1} + \ldots + d_{k+i})$. Thus, it can be concluded that refinement in the definition of the boundary (M-MoSBoD) significantly reduces the average energy dissipation of the edge nodes and their neighboring nodes hence leading to reduced average energy dissipation per node Ψ. Also, reduction in the completion time *'T'* compared to the MoSBoD algorithm is observed.

3.7.2 ANALYZING EFFECTS OF INCREASED DUTY CYCLE

This section analyzes the advantages and disadvantages of increasing the duty cycles of the sensor nodes in terms of completion time of the algorithm *'T'*, average energy dissipation per node Ψ and lifetime of the WSN Φ^{WSN}. Since M-MoSBoD belongs to the class of communication-based methods for boundary dentification therefore, we compared the performance of M-MoSBoD with the state-of-the-art communication-based scheme presented in [76]. Network setup considered in [76] is very similar to our setting, which also ensures comparability. The methods presented in [76] employ *sensor field flooding* (a root node initiates a message and each node relays it to other nodes until all the nodes in the field have received a copy of the message), which is the most commonly used strategy by communication-based schemes. The results presented in [76] suggest that it is currently one of the best schemes for boundary identification. We base our comparison on a deterministic model, which consists of a circular sensor field of radius R, containing N nodes with fixed communication range r and operating at 1% duty cycle as shown in Figure 3.12. For initial experimentations we set the average neighbor node density Γ equal to 10.

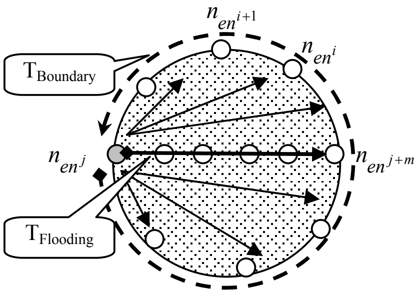


Figure 3.12 Model sensor field for comparing completion time

3.7.2.1 Completion time

This section compares the completion time of the M-MoSBoD ($T_{M\text{-}MoSBoD}$), MoSBoD (T_{MoSBoD}) and the topological boundary identification method ($T_{Floodings}$) defined in [76]. Comparison is performed using a simple deterministic model illustrated in Figure 3.12. Since the MoSBoD algorithm is based on edge node identification, therefore T_{MoSBoD} can be obtained by taking a product of the total number of edge nodes N_{edge} with the sum of the time required by the sink to identify an edge node ($t_{edgeNode}$) and the time taken by the sink to move from the current edge node *i* to a newly identified edge node *i+1* ($t_{i \rightarrow i+1}$).

$$T_{MoSBoD} = N_{edge} * (t_{edgeNode} + t_{i \rightarrow i+1}) \tag{3.1}$$

Figure 3.12 also depicts a model for sensor field flooding that is used for boundary identification in [76]. Since [76] prefers to have the flooding root node located at the periphery of the sensor field therefore we select node n_{en^j} that initiates flooding, as shown in Figure 3.12. Moreover, [76] requires complete flooding of the sensor field at least three times, plus some localized flooding (which is not considered in this calculation). Therefore, the completion time of a flooding based scheme $T_{Flooding}$ is roughly equal to the product of the number of required sensor field floodings Num_{floods} (= 3) and the completion time of one complete flooding process t_{flood}. It is known that [76] prefers the case where the flooder node is located close to the periphery of the WSN therefore t_{flood} (in Figure 3.12) will be equal to the time taken by the flood message from node n_{en^j} to reach a node $n_{en^{j+m}}$ located at the farthest distance along the diameter of the field. So,

$$T_{Floodings} = Num_{floods} * t_{flood} \tag{3.2}$$

It can be inferred from Figure 3.12 that

- the number of edge nodes required to form a connected boundary are upper bounded by $N_{edge} = (2 * \pi * R) / r$
- $t_{i \rightarrow i+1}$ can be calculated by dividing the average distance between the nodes $\overline{d}$ by the speed of the sink V_{sink}.

Substituting these values into Equations (3.1) and (3.2), we get

$$T_{MoSBoD} = [(2 * \pi * R) / r] * [t_{edgeNode} + (\overline{d} / V_{\text{sink}})] \tag{3.3}$$

and,

$$T_{Floodings} = 3 * [\{(2 * R) / r\} * t_{\text{1hop}}] \tag{3.4}$$

In order to compute $t_{edgeNode}$ for T_{MoSBoD} we consider the energy latency model defined in [97] which states that if nodes operate at 1% duty cycle, then the time required by a message to travel one hop distance is 1.126s. But in our setup nodes are operating at 1% duty cycle and the sink operates at 100% duty cycle, so the time required by the message to travel between a node and the sink will be between 0 and 1.126/2 = 0.563s. This delay is mainly due to the sleep time of the nodes. Substituting these values in Equation (3.3) and (3.4) we get,

$$T_{MoSBoD} = [(2 * \pi * R) / r] * [0.563 + (\overline{d} / V_{\text{sink}})] \tag{3.5}$$

$$T_{Floodings} = 2 * [\{(2 * R) / r\} * 1.126] \tag{3.6}$$

On the other hand, in order to compute $t_{edgeNode}$ for M-MoSBoD we assume that the *response schedule* (used to avoid message collision in Module-3.3) issued by an edge node to its neighboring nodes is as follows $\{n_1(0), n_2(0.005), n_3(0.01), ...\}$. Therefore, after receiving a *hello* message from the sink, node n_1 transmits its response immediately; node n_2 transmits a response after a delay of 0.005s, similarly n_3 after a delay of 0.01s and so on. Thus, total time required by the sink to collect responses from all neighboring nodes of the current node will be 50ms (as the average neighbor density Γ = 10). As soon as the sink receives responses from all the neighboring nodes of the current node, it can immediately identify the next edge node (based on the received responses in the corresponding sector numbers and neighbor connectivity check). Therefore, we can say that for M-MoSBoD, $t_{edgeNode}$ is equal to 50ms. Since M-MoSBoD is similar to MoSBoD except the time taken by the sink to identify the next edge node $t_{edgeNode}$ and the number of edge nodes required to identify connected boundary therefore Equation (3.3) can also be used to calculate $T_{M\text{-}MoSBoD}$ as follows,

$$T_{M-MoSBoD} = [(2*\pi*R)/r]*[0.05+(\overline{d}/V_{\text{sink}})] \tag{3.7}$$

Comparing Equations (3.6) and (3.7) yields

$$\begin{array}{cc}
T_{M-MoSBoD}(s) & T_{Flooding}(s) \\
((2*\pi*R)/r)*(0.05+t_{i\rightarrow i+1}) & 3*[\{(2*R)/r)*1.126] \\
3.14*[0.05+(\overline{d}/V_{\text{sink}})] & [3*1.126] \\
0.157+[3.14*(\overline{d}/V_{\text{sink}})] & 3.378
\end{array} \tag{3.8}$$

Similarly, comparing Equations (3.6) and (3.7), we get

$$\begin{array}{cc}
T_{MoSBoD}(s) & T_{Flooding}(s) \\
1.68+[3.14*(\overline{d}/V_{\text{sink}})] & 3.378
\end{array} \tag{3.9}$$

For a comparison between T_{MoSBoD}, $T_{M\text{-}MoSBoD}$ and $T_{Flooding}$, we substitute different values of $\overline{d}$ and V_{sink} in Equations (3.8) and (3.9). Now considering a realistic scenario where $\overline{d}$ and V_{sink} are set to 4m and 2m/s respectively it can be observed that T_{MoSBoD} and $T_{M\text{-}MoSBoD}$ showed increased delay compared to $T_{Flooding}$ by the factor of 1.35 and 0.9 respectively. Thus, a substantial reduction in the completion time of the boundary identification has been achieved ($T_{M\text{-}MoSBoD} < T_{MoSBoD}$). It can also be seen from Equations (3.8) and (3.9) that decrease in $\overline{d}$ or increase in V_{sink} will lead to the reduction in $T_{M\text{-}MoSBoD}$ and T_{MoSBoD}. This is due to the fact that in both cases $t_{i\rightarrow i+1}$ decreases, which is a major factor in the completion time of MoSBoD and M-MoSBoD.

3.7.2.2 Energy Consumption

With respect to average energy dissipation per node Ψ during the execution of M-MoSBoD algorithm, we investigated two hypotheses in the following.

Hypothesis 1: Number of messages sent out by each sensor node is constant and the number of messages received depends linearly on its neighbor node density.

It is known from Section 3.4.1 that after the deployment of a WSN each sensor node is responsible of transmitting only two messages (for boundary identification) one containing the node ID and second broadcasting a signal that a node has acquired validity. As a result, each node receives only two messages per neighboring node. However, it is known from Module 3.3 that each edge node and its neighboring nodes had to perform additional communication (2 message transmissions and 2 receptions) during the identification of the corresponding edge node by the sink. Since the number of edge node and their neighboring nodes in a WSN constitutes a small portion of the total number of sensor nodes in a WSN. Thus, on the average number of messages transmitted by a node during boundary identification procedure is constant (nearly equal to 2) irrespective of the neighbor density, while the number of messages received by a node is a linear function of its neighbor density. Hence, *Hypothesis 1* holds true. Furthermore, increase in the neighbor node density results in linear increase in average energy dissipation per node Ψ and hence linear decrease in the lifetime of a WSN Φ^{WSN} (because energy dissipation of the nodes is almost symmetrical throughout the sensor field).

Hypothesis 2: Scaling up/down the area of the sensor field with constant node density has no effect on the number of messages exchanged by the sensor nodes.

It has already been shown during the investigation of *Hypothesis 1* that during the execution of M-MoSBoD number of messages transmitted by a node is constant (=2) while the number of message received depends on the neighbor node density. Thus given a fixed neighbor node density, number of messages exchanged by the nodes is not affected by a change in the area of the sensor field. Another justification for this behaviour is that in the M-MoSBoD algorithm node-to-node communication takes place only to determine the validity status of a node (refer to Section 3.4.1), which is a localized phenomenon and does not depend on the size of the field.

Hypothesis 2 shows the highly scalable nature of *M-MoSBoD* where the size of a WSN does not affect the average energy dissipation Ψ per node or the lifetime Φ^{WSN} of a WSN (under fixed neighbor node Γ).

Now we compare MoSBoD, M-MoSBoD and algorithm from [76] for total energy dissipation of the sensor field *'E'* during the boundary identification process.

It is assumed that the sensor field shown in Figure 3.12 contains Mica2 motes operating at 1% duty cycle. It is known that the boundary identification scheme presented in [76] requires each node to participate in three complete flooding of the field and in a number of localized flooding. Therefore, the total number of messages transmitted by each node is at least three, the number of messages received by a node equals $3*\Gamma$ and the total energy dissipation for executing boundary identification algorithm presented in [76] is as follows,

$$E_{Flooding} = (E^{tx} * 3 * N) + (E^{rx} * 3 * \Gamma * N)$$
$$E_{Flooding} = 3N[E^{tx} + (E^{rx} * \Gamma)] \tag{3.10}$$

E^{tx}, E^{rx} are the energies required to send/receive a message.

In case of the MoSBoD algorithms the total energy dissipation of the sensor field E_{MoSBoD} is the sums of the energy dissipation during node validity calculation and energy dissipation during edge node identification.

$$E_{MoSBoD} = [2N * \{E^{tx} + (E^{rx} * \Gamma)\}] + [N_{edge} * \{e_{edge} + (\Gamma * e_{edgeNb})\}] \tag{3.11}$$

e_{edge} and e_{edgeNb} are the energies dissipated by each edge node and each of its neighbor nodes during the boundary identification process. Equation 3.11 also holds true for $E_{M\text{-}MoSBoD}$ however, N_{edge} and e_{edge} will have different values, which we discuss in the following.

In case of the MoSBoD algorithm, during next edge node identification, each edge node and each of its neighboring nodes transmit and receive one additional message *(hello message from the sink and response to the sink respectively)* compared to rest of the sensor nodes. Figure 3.13 shows if a node is in the neighbor list of an edge node $n_{en^{i-1}}$, then there is a high probability that it will also be in the neighbor list of edge nodes $n_{en^{i-2}}$ or $n_{en^{i}}$. Therefore, the number of additional messages sent and received by an edge node and its neighboring nodes is two. Moreover, an edge node receives an additional message from the sink, the *edge node confirmation message*. Therefore, the additional energy dissipation by an edge node is $e_{edge} = 3E^{rx} + 2E^{tx}$, and for the neighboring nodes of an edge node $e_{edgeNb} = 2E^{rx} + 2E^{tx}$. On the other hand, in case of the M-MoSBoD algorithm $e_{edge} = 3E^{rx} + 3E^{tx}$. An additional message sent is the *request to upgrade duty cycle message to neighboring nodes*. Similarly, $e_{edgeNb} = 5E^{rx} + 2E^{tx}$, where 3 additional received messages are *two duty cycle upgrade messages* and *edge node confirmation (duty cycle degrade)*. Since, in case of the M-MoSBoD algorithm nodes receive messages while operating at *100%* duty cycle, the increased energy consumption by an edge node and its neighboring nodes becomes $e_{edge} = e_{100\%} + 3E^{tx}$ and $e_{edgeNb} = e_{100\%} + 2E^{tx}$, where

$e_{100\%}$ is the total energy depletion of the node for the period of time it operated at 100% duty cycle. Substituting into Equation (3.11) we get,

$$E_{MoSBoD} = 2N[E^{tx} + (E^{rx} * \Gamma)] + N_{edge}[(3E^{rx} + 2E^{tx}) + ((2E^{rx} + 2E^{tx}) * \Gamma)] \tag{3.12}$$

and,

$$E_{M-MoSBoD} = 2N[E^{tx} + (E^{rx} * \Gamma)] + N_{edge}[(e_{100\%} + 3E^{tx}) + ((e_{100\%} + 2E^{tx}) * \Gamma)] \tag{3.13}$$

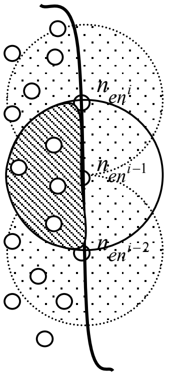


Figure 3.13 Edge nodes and their neighboring node

Analyzing Equations (3.10), (3.12) and (3.13) it can be inferred that $E_{Flooding}$ is roughly $(N - N_{edge})$ times higher than the E_{MoSBoD} and $E_{M\text{-}MoSBoD}$. In order to quantify the gains we assume a sensor field composed of Mica2 nodes where current depletion during reception is *10mA* and during transmission it is *27mA*. It is also known from [98] that the energy consumption of a node can be calculated using the following equation:

$$E(joules) = current(A) * time(s) * voltage(volts) \tag{3.14}$$

Here we assume that the time taken by a node to send or receive a message *(time(s))* is 0.01 seconds (for BMAC when nodes operate at 35% duty cycle then t_{1hop} = 0.025 seconds [64]). Furthermore, in the case of M-MoSBoD algorithm it is assumed that each edge node and its neighboring nodes operate at 100% duty cycle for 1second in listening mode to receive *hello* messages from the sink. Utilizing Equation (3.14) we can calculate $E^{tx} = 0.027 * 0.01 * 3 = 0.81mJ$, $E^{rx} = 0.01 * 0.01 * 3 = 0.3mJ$, $e_{100\%} = 0.01 * 1 * 3 = 30mJ$. Substituting these values into Equations (3.10), (3.12) and (3.13) we can calculate the energy depletion of the nodes for MoSBoD and M-MoSBoD. Since the MoSBoD algorithm consumes least energy from the sensor nodes therefore we plotted $E_{M\text{-}MoSBoD}$ and $E_{Flooding}$ as the percentage of E_{MoSBoD} (= 100%). Curves 1sec, 2sec, 3sec, 4sec and 5sec show $E_{M\text{-}MoSBoD}$ when M-MoSBoD require each edge node and its neighboring nodes to operate at 100% duty cycle for 1, 2, 3, 4 and 5 seconds. Consider the scenario where edge nodes and their neighbors operate at 100% duty cycle for only *1s* during the execution of M-MoSBoD and the radius of the sensor field is 500m (size of a small farm land). It can be seen from Figure 3.14 that $E_{M\text{-}MoSBoD}$ and $E_{Flooding}$ are almost 75% and 50% higher than E_{MoSBoD}, respectively. But with the increase in the size of the sensor field (R = 2500m) difference between $E_{M\text{-}MoSBoD}$ and E_{MoSBoD} get down to only 3% which validates our hypothesis derived from Equation (3.13). This trend occurs because the M-MoSBoD algorithm only increases the duty cycles of edge nodes and their neighboring nodes. Since the percentage of these nodes decreases compared to the total number of nodes in the WSN with the increase in the size of the sensor field therefore, we observe a decrease in the total energy dissipation of M-MoSBoD compared to MoSBoD.

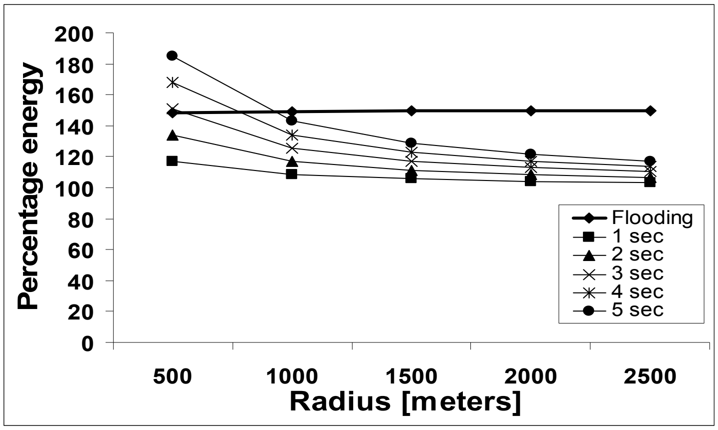


Figure 3.14 $E_{M\text{-}MoSBoD}$ and $E_{Flooding}$ as the percentage of E_{MoSBoD}

3.8 SUMMARY

In this chapter, we introduced a new scheme for boundary identification of a WSN. Our approach utilizes a mobile sink for identifying edge nodes, which are then connected in a meaningful way to obtain the boundary of a WSN. Our analysis has shown that the use of a mobile sink for boundary identification (M-MoSBoD) can lead to the reduction in node-to-node communication and hence reduction in the average energy dissipation per node as well as increased lifetime of the network compared to the state-of-the-art. Some other advantages of M-MoSBoD are:

- Boundary obtained using M-MoSBoD is independent of the neighbor node density, on the other hand schemes presented in [84], [76] or [96] require a neighbor density of at least 8 to produce acceptable boundaries [76].

- In contrast to [84], [76] and [96] the M-MoSBoD algorithm is independent of sensor field flooding, as a result gains in the lifetime of a WSN for M-MoSBoD increases with the size of a sensor field compared to [84], [76] and [96].
- Unlike hardware based boundary identification schemes M-MoSBoD algorithm does not have any special hardware requirements from the nodes. Thus, cost efficiency is achieved.

In the following chapters we will utilize this obtained boundary information and the sink mobility to establish energy efficient routing paths for data collection from the sensor field.

CHAPTER 4. CONGESTION AVOIDANCE FOR LOW LATENCY AND ENERGY EFFICIENCY IN WSN (CALEE)

4.1 RESEARCH CONTRIBUTIONS

According to traditional definition of wireless sensor networks (WSN), a dense and static sensor node deployment is implicitly required. Subsequently, there arises a fundamental problem in WSNs with static topology: the non-uniformity of energy consumption among the sensor nodes. In fact, the nearer a sensor node lies with relative to the sink node; the faster its energy will be depleted. In case of sensor node failure or malfunctioning around sink, the network connectivity and coverage cannot be guaranteed.

Intuitively, there are two solutions to the above problems. On the one hand, if some sensor nodes withdraw from the network due to energy exhausting such that the network loses necessary connectivity and sensing coverage, there must be other supplementary sensor nodes deployed. On the other hand, the sensor nodes should be capable of finding and reaching the sink node in possibly different positions, whether there are multiple sink nodes or the sink node is able to change its location.

Some network designers consider mobility harmful because it leads to dynamic and unpredictable links among devices. Compared with traditional wired networks, it indeed becomes much more difficult to design network control functions, such as routing path maintenance, in an efficient way under mobility. On the other hand, some researchers also realize that, by involving mobility into network design (i.e., mobility in a desirable way) one can improve the performance of wireless networks.

This chapter presents a data routing protocol that exploits sink mobility in a desirable way to achieve energy efficiency and low latency in WSN. Two main building blocks of our algorithm are sink mobility and virtual partitioning of the sensor field into sectors. Both, the use of a mobile sink and virtual partitioning of the sensor field helps to improve the lifetime of a WSN by reducing the data routing load from the sensor nodes. This chapter also discusses an in-network storage model for data persistence that exploits high sensor node density to avoid data losses.

Rest of the chapter is organized as follows; Section 4.2 introduces routing in WSN. Section 4.3 discusses current state-of-the-art data routing protocols used in different WSN configurations and argue the need for a new routing protocol. Section 4.4 presents our routing scheme titled, congestion avoidance for low latency and energy efficiency (CaLEe); Section 4.5 elaborates the in-network storage model, which ensures data persistence in WSN, Section 4.6 concludes the chapter.

4.2 INTRODUCTION

Over the last few years, many communication protocols for energy conservation in WSNs have been proposed. These include, energy conserving routing (e.g.,[99-101]), topology control (e.g., [102-104]) and clustering (e.g., [105, 106]). Although all these protocols achieve their optimization goals under certain conditions, they always focus on the sensor nodes. Recently, it has been realized that further improvements on the lifetime of WSNs Φ^{WSN} can be achieved if one shift focus to the behavior of sinks [13].

As data traffic must be concentrated towards a small number (typically one) of sinks, the nodes around a sink have to forward data for other nodes whose number can be very large; this problem always exists, regardless of what energy conserving protocol is used for data transmission. In other words, applying energy conserving protocols does not directly lead to load balancing within the whole network. We will show how unevenly the load is distributed within a network under different network configurations. As a result, those bottleneck nodes around sinks deplete their batteries much faster than other nodes and, therefore, their lifetime upper bounds the lifetime of the whole network Φ^{WSN}.

Intuitively speaking, the load on sensor nodes can be more balanced if a sink changes its position from time to time, because such a sink can distribute over time the role of bottleneck nodes and even out the load. Although a sink is usually assumed static, we argue that mobile sinks are practical for many realistic applications of WSNs. For example, consider a scenario where nodes are equipped with batteries that cannot be replaced because the sensor nodes are not accessible, or because changing batteries would be hazardous or costly. This may be the case for sensors in smart buildings, where batteries might be designed to last for decades, and in environmental or military sensing under hostile or dangerous conditions (e.g., avalanche monitoring). In such cases, it may be desirable and comparatively simple to move a sink very infrequently (e.g., once a day or a week) by a human or by a robot. For example, in the avalanche-monitoring scenario, a sink may be deployed at the periphery of the monitored area, and moved by helicopter once in a while. In the building scenario, the sink can be "virtually" moved: computers in different offices serve as the sink in shifts. In the military monitoring scenario, moving the sink may require some effort, but it can be acceptable if done infrequently.

Thus, under different realistic scenarios sink mobility is possible and can help to increase the lifetime of a WSN Φ^{WSN} by evenly distributing the relaying load on the sensor nodes.

4.3 RELATED WORK

In the following, we have categorized current state-of-the-art based on the sink behavior in the sensor field that can be static or mobile.

4.3.1 STATIC SINK BASED WSNS

In early days, a typical WSN was composed of static sensor nodes and a static sink placed inside the observed region. In such a setup, major energy consumer was the communication module. In small networks, sensors can send data directly to the sink node; in larger setups, multi-hop communication is required. In both cases, the energy consumption depends on the communication distance. One way to reduce the communication distance is to deploy multiple sinks and each sensor node is programmed to route data to the closest sink. It will reduce the average path length from source to sink (η) and hence reduced energy dissipation of the nodes.

Once decision to the use of multiple static sinks has been made then it is to be decided where to deploy them inside the observed region. This is a typical facility location problem: given a set of facilities (e.g., sink nodes) and a set of costumers (e.g., sensors) to be served from these facilities, where to deploy those facilities, and which facility should serve which costumer, so as to minimize the total serving cost (e.g., the overall energy consumption)? There are several well-known solutions for such a problem, based on integer linear programming or iterative clustering techniques. However, these solutions can usually handle only small networks, they can choose only from a fixed set of possible locations, and most importantly need a global knowledge of the network. After deploying the sink nodes, certain sensors will deplete their energy rapidly: those near the sink in case of multi-hop routing, and those far from the sink in case of direct sensor-sink exchange [107]. Thus, raising questions about the static sink based solutions.

4.3.2 MOBILE SINK BASED WSNS

In order to overcome the shortcomings observed in the case of static sink the use of mobile sink was proposed. The mobile sink can follow different types of mobility patterns in the sensor field such as random mobility, predictable/fixed path mobility, and controlled mobility. In the following we discuss some of these proposed solution from each class.

4.3.2.1 Random mobility

In this class, the sink follows a random path in the sensor field and implements a pull strategy for data collection from the sensor nodes (a node forwards the data only when the sink initiate a request for it). Chatzigiannakis et al. [108] have shown that if high data latency is permissible then random sink mobility can be used to improve the lifetime of a WSN. The sink can requested data from either one or Λ (where, $\Lambda>1$) hop neighbors. Single hop data collection leads to high energy gains (no data relaying load on the sensor nodes) but it also result in high end-to-end data latency $\bar{\Delta}$ because the time taken by the sink to traverse entire sensor field can be very high. It has also been observed that in case of random sink mobility with single hop data routing it is often not possible to guarantee 100% data collection from a WSN, because with random mobility pattern there is no guarantee that the sink will be able to reach all the nodes in the sensor field. However, if reduced end-to-end data latency $\bar{\Delta}$ and maximum coverage of the sensor field is desired then the sink can be programmed to collect data from Λ hop neighbors. This results in increased relaying load on the sensor nodes and reduces lifetime of the WSN. In the following, we discuss few applications that utilize random mobility of the sensor node and/or sink for energy efficient operation of the WSN.

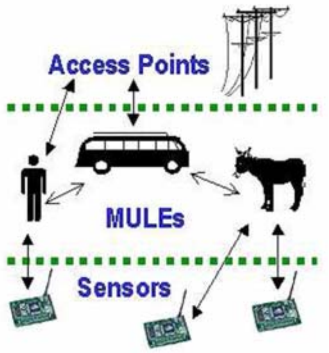


Figure 4.1 A three tier architecture [109]

R.C. Shah et al; proposed a three-tier architecture that exploits two-dimensional random walk of mobile entities (called MULES), such as humans or animals, in order to collect and relay information received from the sensors to a central control station as shown in Figure 4.1 [109]. They consider number of MULEs, sensors and access points as key system variables. On the other hand, performance is measured as the data success rate and the required buffer capacities on the sensors and the MULEs.

Krishnamachari et al; investigated the coverage time of a graph when performing a random walk that uses deterministic choice [110]. It was achieved by introducing the Random Walk with Choice, RWC(d), in which, instead of selecting one neighbor at each step, the walk selects $\wedge$ neighbors uniformly at random. Then the least visited node among them is chosen to move to. A related modification to random walks called Vertex-Reinforced Random Walks, VRRW, was proposed in [111] and studied outside of the context of coverage time. In VRRW, the walk prefers the most visited nodes, without choice.

Volkov et al; investigated the impact of the number, velocity, transmission radius and data gathering mode of mobile sinks on large scale and sparse WSN [112]. The sparse wireless network is characterized by loose connectivity, in which the process of data gathering is mainly performed by mobile sinks. In such a situation, it is much difficult to guarantee low data delivery delay as each sensor node has to wait for a mobile sink to approach before the transfer occurs. Whereas analysis and simulation results demonstrate that by choosing the appropriate number, transmission range and velocity of mobile sinks, an acceptable end-to-end data latency can be achieved, which is sufficient for many practical applications.

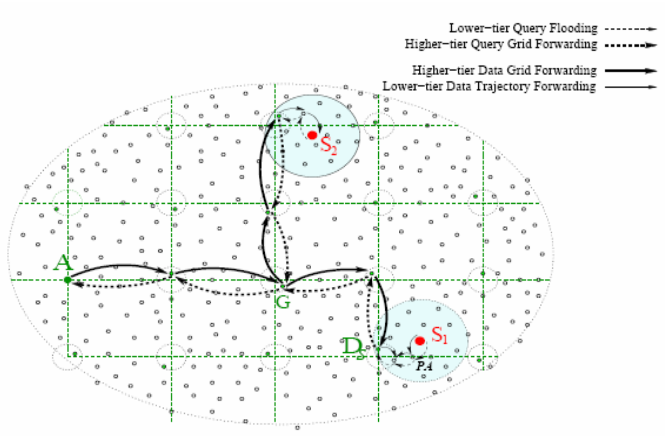

Lower−tier Query Flooding
Higher−tier Query Grid Forwarding
Higher−tier Data Grid Forwarding
Lower−tier Data Trajectory Forwarding
S_2
A
G
D_s
S_1
PA

Figure 4.2 A two-tier data dissemination approach [33]

Ye et al; [33] have presented a TTDD, a Two-Tier Data Dissemination approach that addresses the multiple, mobile sink problem. Their solution is based on a grid structure where only the nodes located at grid points need to acquire and forward the information as shown in Figure 4.2. When a node detects an event it proactively build a grid structure and select the nodes (called dissemination nodes) positioned close to the grid points as data forwarding nodes. With the help of this grid structure, a query from a sink can reach the source in two tiers. “The lower tier is within the local grid square of the sink's current location (called cells), and the higher tier is made of the dissemination nodes at grid points. The query forwarding process lays information of the path to the sink, to enable data from the source to traverse the same two tiers as the query but in the reverse order.” [33]

Kinalis et al; proposed three protocols based on different mobility patterns of the sink and data collection models [113]. Starting with the simple random walk authors propose a combination of random walk and deterministic biased walk that attempts to equally distribute mobile sinks in the whole network area. In particular: a) the first protocol introduces many mobile sinks; its main advantage is significant reduction in latency as well as low energy dissipation. b) The second protocol introduces a more aggressive data collection strategy and uses a combination of static together with mobile sinks; results have shown that even with small number of mobile sinks great benefits can be achieved in terms of reduced latency and increased data delivery rate. c) Third protocol is based upon the second protocol and further enhances its’ performance by loosely coordinating the mobile sinks. Figure 4.3 shows that each sink leaves a trail on its mobility path when other sinks encounter this trail they change their mobility direction to improve the coverage of the sensor field as a result even lower latency and higher delivery rates can be achieved. To retrieve the data from sensors, the sink movement is combined with two data collection strategies: a passive single-hop and a limited multi-hop.

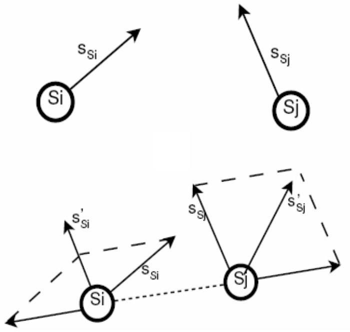


Figure 4.3 Sink mobility patterns in [113]

Major overhead in random sink mobility based schemes is the difficulty in tracking the current position of the sink. In order to address this issue Shim et al., presented locators based solution for mobile sinks that track current location of the sink [114]. When a sensor wants to report data to sinks, it can acquire sinks' location from these locators, which are uniformly distributed in the sensor fields by hash, bashed structured replication as shown in Figure 4.4. In the presented work each sink update its location only to neighboring four locators, which are then responsible of propagating this information to other locators.

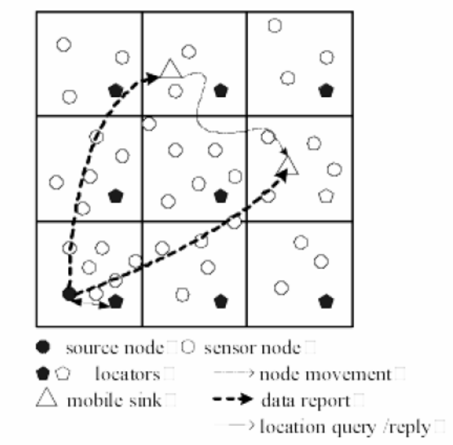


Figure 4.4 Tracking the position of randomly moving sink in WSN [114]

Another work in this regard was presented in [115] where *Interest Dissemination with Directional Antenna (IDDA)* for reactive routing in WSNs was presented. IDDA is based on assumption that the sink knows its mobility speed and direction of mobility, which implies it can calculate its future position in the field. The utilizing a directional antenna the sink broadcast interest packets along its direction of mobility so that the data from the nodes located in the vicinity of the future position of the sink can be gathered as shown in Figure 4.5. Authors claim that their scheme result in increased packet delivery ratio and reduced power consumption.

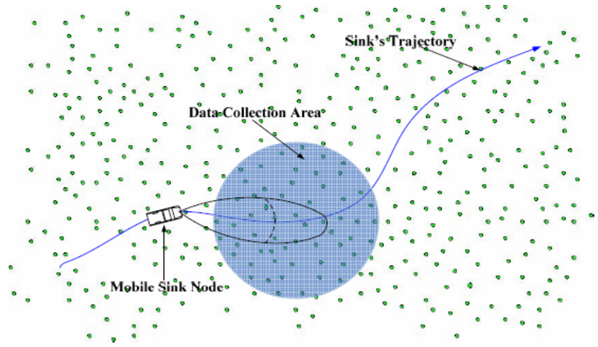


Figure 4.5 Interest dissemination using directional antenna at a mobile sink [115]

4.3.2.2 Predictable/fixed path mobility

In predictable/fixed path mobility schemes sink is programmed to follow fixed path in a round robin fashion.

In [116] authors assumes that a mobile sink is responsible of traverses a fixed path repeatedly and fetching data from neighboring nodes as shown in Figure 4.6. On the other hand, sensor nodes are programmed for reactive routing thus they transmit only when the sink arrives in their vicinity. As a result, energy dissipation per node is very low however; end-to-end data latency $\overline{\Delta}$ is high.

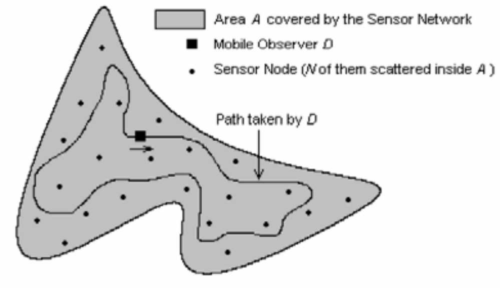

Area *A* covered by the Sensor Network
Mobile Observer *D*
Sensor Node (*N* of them scattered inside *A*)
Path taken by *D*

Figure 4.6 A mobile sink traversing the sensor field along a fixed path [116]

Luo et al. [117] worked to find a mobility trajectory for the sink that should result in most balanced energy dissipation from the nodes. Their results have shown that in periodic sensing class of applications (Section 2.1.3) longest lifetime Φ^{WSN} for the WSN can only be achieved if the mobility trajectory of the sink is close to the *periphery* of the sensor field. Increased end-to-end data latency $\overline{\Delta}$ and packet loss are major problems that arise due to such sink mobility in WSNs. Luo et al. [118] have also addressed these problems by presenting a routing protocol that not only balances the energy dissipation of the nodes but also tries to reduce data latency and data losses. Their scheme is based on discrete mobility of the sink, where the sink sojourn time at predetermined location is greater than its mobility time that helps to avoid frequent route updates that results in increased lifetime of the network as shown in Figure 4.7.

Yaoyao Gu et al; investigated the mobile element scheduling (MES) problem and propose a solution based on virtual partitioning of the sensor field and scheduling [119]. Their idea is to partition the nodes into groups of nodes based on their data generation rate. Then a schedule is prepared for mobile element (sink) to visit each group based on the data generation rate of the nodes in that group. In the last schedules of the groups are merged to obtain a mobility path for the sink such that buffer overflows can be avoided at each node.

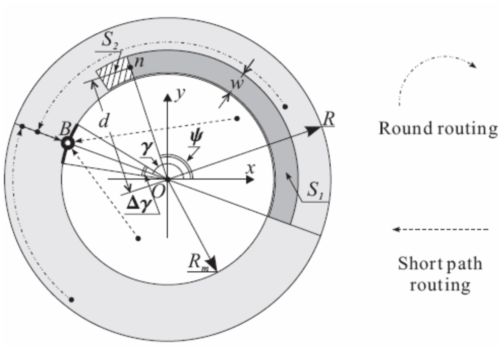


Figure 4.7 Joint mobility and routing strategy

4.3.2.3 Controlled mobility

Use of controlled sink mobility has also been analyzed for increasing the lifetime of a WSN. Jayaraman et al. [120] outlined a framework which utilizes context aware mobile pervasive devices for data collection from the sensor field. These context aware devices are supposed to be intelligent enough to retrieve their possible future location and direction of mobility based on data gathered from the sensor field. This information is utilized for planning efficient data collection from the sensor field.

Bi et al; argued that an energy-unconscious mobility of the sink results in uneven energy dissipation from the nodes [121]. To address this particular problem authors presented a mobile sink based approach, where the sink tries to keep away from the nodes with less residual energy and try to be in the vicinity of those nodes that have high residual energy. This helps to balance the energy dissipation from the nodes hence resulting in increased lifetime Φ^{WSN} of the network.

The idea of using controlled mobility in WSNs for increased lifetime Φ^{WSN} of the network was also discussed in [122]. In order to determine sink movements, authors first define a Mixed Integer Linear Programming (MILP) based analytical model whose solution determines those sink routes that maximize network lifetime Φ^{WSN}. Moreover, a Greedy Maximum Residual Energy (GMRE) heuristic moves the sink only to those sites where maximum residual energy of the nodes is maximum as shown in Figure 4.8. With the help of experiments it has been shown that controlled mobility of the sink can result in six-fold increase in the lifetime of a WSN compared to the case of static sink placed at the center of a WSN and two-fold increase compared to the case of uncontrolled mobility of the sink.

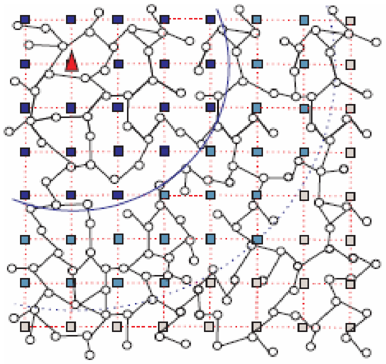


Figure 4.8 Controlled sink mobility in a WSN [122]

4.3.3 SUMMARY

Following conclusions can be drawn from above discussion,

Sink mobility strategy (random, periodic/fixed path or controlled) is greatly influenced by the application requirements. However, irrespective of the used mobility strategy, mobile sink helps to improve the lifetime Φ^{WSN} of a WSN compared to that of a static sink; on the other hand, two major issues arise due to sink mobility, increased end-to-end data latency $\overline{\Delta}$ and increased overhead for route maintenance. Both these issues are caused by frequent routing path updates that result due to sink mobility. In order to address these particular issues in the following we propose a new routing protocol.

4.4 CALEE ROUTING PROTOCOL

Assuming a mobile sink, we investigate the problem of load-balanced data collection from the nodes that result in increased lifetime Φ^{WSN} of a WSN. Both the adapted mobility strategy of a sink and performance of the developed routing protocol are highly influenced by the characteristics of a WSN. Thus, before we go into the details of our routing protocol we state assumptions being made regarding deployed sensor field and adapted sink mobility strategy.

4.4.1 ASSUMPTIONS

It is assumed that wireless sensor nodes are uniformly but randomly deployed to a remote region in dense numbers. Nodes are responsible for sensing and reporting their readings with constant interval *t*. Boundary of the sensor field is known that is identified using M-MoSBoD algorithm presented in Chapter 3.

4.4.1.1 Adopt sink mobility

Basagni et al. investigated this question in [122]. They consider a scenario in which 361 sensor nodes are deployed on a 19×19 grid covering a squared area of side L = 400m. Sensor nodes have a transmission radius equal to 30m, which imposes a maximum of 8 neighbors per node. The sink moves among 8×8 sink sites (grid deployment) at the speed of 1m/s. A shortest path-like routing is used to deliver data in a multi-hop fashion from the sensor to the current site of the sink. The time for sink sojourns at a node is set from 50000s to 1000000s and the maximum distance between current position of the sink and its next sojourn location was set to 190m. Figure 4.8 depicts the considered sensor field.

Authors used ns2 to evaluate following routing schemes on the considered sensor field;

OPT - A mathematical model is defined that optimizes network lifetime by moving the sink among a finite given set of sites (the sink sites), and computes the optimal sojourn time at those sites.

GMRE - Greedy Maximum Residual Energy is more suited to the nature of WSNs. The main idea behind the GMRE distributed protocol is that of controlling the movements of a mobile sink toward the zones of the network where nodal residual energy is higher. This leads to balanced energy consumption throughout the network, and to a longer network lifetime.

PM - Peripheral mobility is the case where sink moves along the periphery of the WSN.

RM - Random mobility is the case where sink follows a random path in the sensor field.

STATIC - A scenario where the sink is (optimally) placed at the center of the deployment region.

Table 4.1 shows the percentage energy gains that OPT, GMRE, PM and RM achieved over the case of a static sink.

Sojourn time	OPT	GMRE	PM	RM
50000s	512	374	326	279
1000000s	484	315	192	42

Table 4.1 Improvements over the case of STATIC sink (%) [122]

Table 4.2 shows the average packet latency for OPT, GMRE, PM and RM compared to the case of a static sink.

Sojourn time	50000	1000000
OPT	0.19	0.19
GMRE	0.214	0.214
RM	0.175	0.18
PM	0.211	0.212
STATIC	0.13	0.13

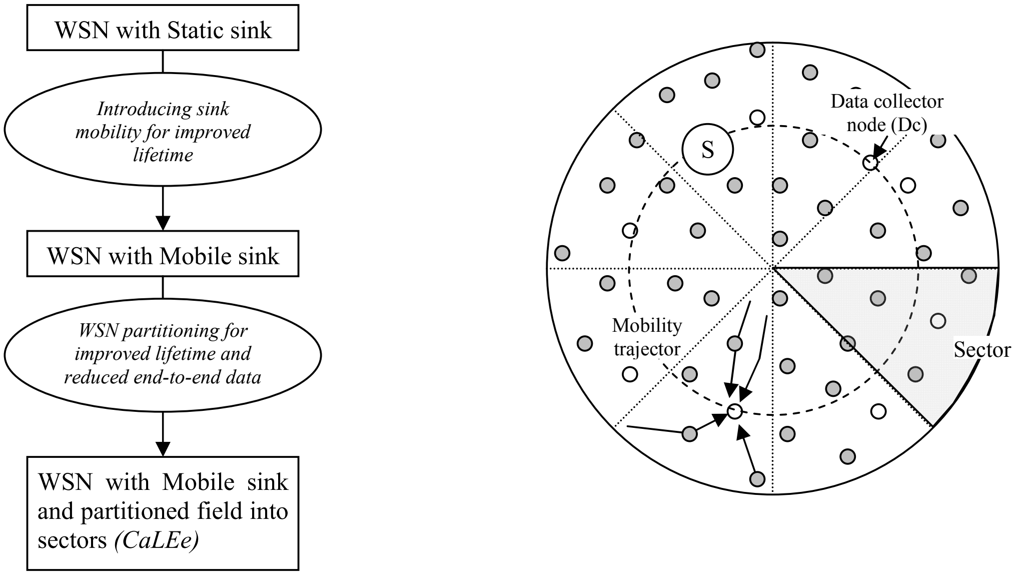
Table 4.2 Average packet latency (seconds) [122]

It can be inferred from Tables 4.1 and 4.2 that sink mobility in any form results in increased lifetime Φ^{WSN} of the network as well as increased end-to-end data latency $\overline{\Delta}$ which may make them non suitable for real time applications. Based on these observations major goal of our research is to utilize sink mobility for increased lifetime Φ^{WSN} while keeping $\overline{\Delta}$ comparable to that of a static sink case.

4.4.2 ALGORITHM

We now present our routing scheme titled *Congestion avoidance for low Latency and Energy efficiency (CaLEe)* in WSN. Schemes presented in Section 4.3 improves the lifetime Φ^{WSN} of a WSN by introducing sink mobility in a static WSN, *CaLEe* takes this one step further by introducing sensor field partitioning in order to achieve lifetime improvements with minimum overheads (end-to-end data latency $\overline{\Delta}$ and effort required for route maintenance) observed in current state-of-the-art. Figure 4.9(a) depicts this evolution of the routing protocols.

CaLEe is based on virtual partitioning of a sensor field into sectors and discrete mobility of the sink along a fixed trajectory that passes through each sector only once. Each sector has a data collector node (*Dc*), which are connected linearly to obtain the mobility trajectory of the sink. Each *Dc* is responsible for collecting data from nodes located within its sector and the mobile sink periodically visits each data collector node to retrieve the gathered data as shown in Figure 4.9(b). On reaching the *Dc* the mobile sink transmits a data request to it. *Dc* respond by reporting the total number of bytes it wants to transfer followed by actual data stream. Sojourn time of the mobile sink at a *Dc* is determined by the amount of data that is to be transferred from the data collector node. This saves us from keeping track of the position of the sink during the data transfer (whether the sink is within communication range of *Dc* or it has moved out of range) that is required by most of the existing mobile sink based data routing protocols [33, 114] to avoid data losses. Thus, by introducing sensor field partitioning we hope to achieve better lifetime for WSN compared to the cases of static sink (hereinafter, just called SS) and non partitioned sensor field with mobile sink (hereinafter, just called MS).


WSN with Static sink
Introducing sink mobility for improved lifetime
WSN with Mobile sink
WSN partitioning for improved lifetime and reduced end-to-end data
WSN with Mobile sink and partitioned field into sectors (CaLEe)
S
Data collector node (Dc)
Mobility trajector
Sector

(a): Evolution of the routing protocols in WSN (b): WSN implementing CaLEe routing scheme

Figure 4.9 Routing schemes for WSN

Now we will answer following open questions:

- How can we partition the sensor field into sectors?
- How can a routing node decide where it should forward the data?

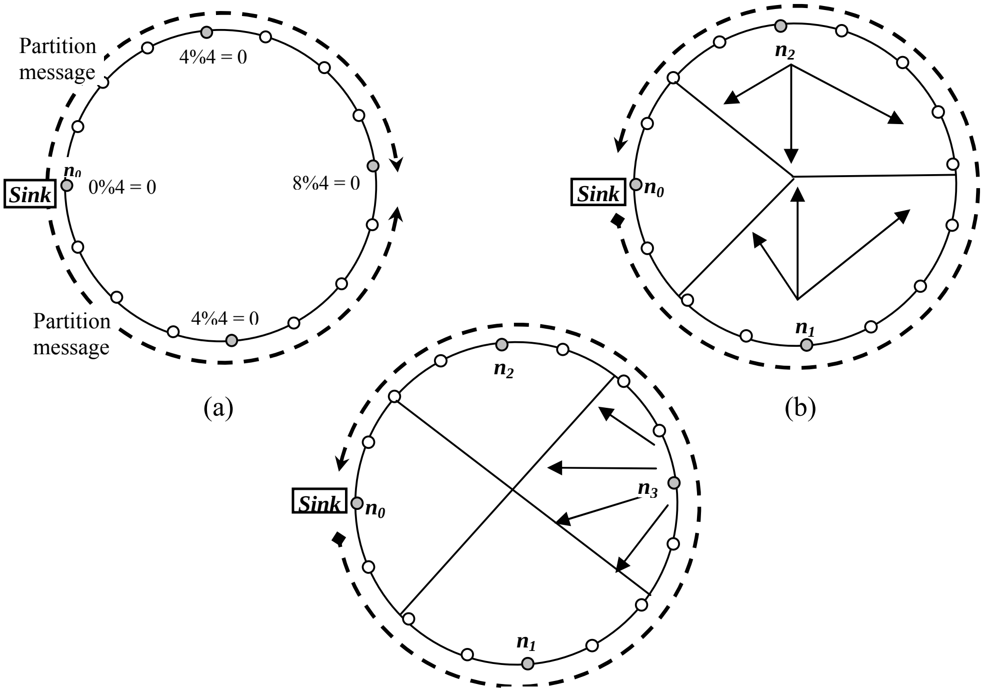
4.4.3 PARTITIONING AND ROUTE ESTABLISHMENT

It has been assumed that the sink knows the boundary of the sensor field identified using M-MoSBoD algorithm presented in Chapter 3. This implies that the sink knows the total number of edge nodes *(N_{edge})* in WSN and each node positioned at the boundary of the sensor field also know that it is an edge node. Then depending on the desired number of sectors *($N_{sectors}$)* the sink initiates sensor field sectoring by transmitting a partition message to the current edge node (node where sink is currently position). The partition message is comprised of two fields *HopCount* and *Sect* *($= N_{edge}/N_{sectors}$)*. Each edge node that receives the partition message performs the following check,

```
if I am an edge node AND I have already received a partition message then
        Drop the partition message
```

```
else if I am an edge node AND $HopCount\%N_{sectors} == 0$ then
        Initiate a flooding message to declare a new sector, acquire the status of Dc
HopCount++
Forward partition message to the neighboring edge node
```

First *if* statement is used to stop the propagation of partition message once all the edge nodes have received it, as shown in Figure 4.10(a). Second check defines the criterion for an edge node to become a data collector node. Here, *HopCount* is simply a hop count that gets incremented as the message progress from one hop to the next.



(a) (b)

(c)

Figure 4.10 Partitioned sensor field

Then partitioning of the sensor field into sectors and route establishment in the WSN is achieved with restricted sensor field fielding from the data collector nodes. Each *Dc* initiates a flood message to inform sensor about its newly acquired status. The flood message contains two fields: ID of the data collector node and the hop count that is initialized with 1. However, in order to make the flooding process energy efficient a restrictive flooding rule was implemented at each node. Restrictive flooding prohibits a node from forwarding those messages that do not improve the routing efficiency in WSN. For example, consider a scenario where a node n_i knows a data collector node *Dc-1* that is Λ hops away. If n_i receives a flooding message from another data collector node *Dc-2* that is located Π hops away then the node n_i will forward this message only if the $\Pi < \Lambda$. This simple rule avoids unnecessary flooding in the sensor field and helps to improve lifetime of a WSN.

Figure 4.10(a) presents one scenario where total number of edge nodes in the sensor field N_{edge} equals 16 and desired number of sectors $N_{sectors}$ equals 4. Based on these numbers we can calculate *Sect (=* $N_{edge}/N_{sectors}$*)* 16 / 4 = 4. Now the sink positioned at node n_0 transmits a partition message to current node where $N_{sectors} = 4$ and $HopCount = 0$. Node n_0 will perform following check $HopCount\%N_{sectors}$, since the result equals zero therefore it will acquire the status of *Dc* and forwards the partition packet to neighboring edge nodes and floods the field to inform the sensor nodes about its newly acquired status. Each node that receives the flooding message will verify whether the newly reported data collector node is located at the shortest hop count distance than the previously known data collector node, if yes then update routing path to new collector node and forward the flood message else drop the flood message. As a result, in given example flood message from node n_0 will be transmitted to entire sensor field because n_0 is the first node to flood the sensor field for route establishment. However, flood messages from nodes n_1 and n_2 will only be transmitted to 1/3rd of the total number of sensor nodes as shown in Figure 4.10(b). Moreover, the flood message from node n_3 will reach only 1/4th of the total number of the sensor nodes that are located close to n_3 then any other *Dc* node as shown in Figure 4.10(c). Therefore, at the end of this procedure we obtain a partitioned sensor field with a handful of data collector nodes. Furthermore, each node in the sensor field also knows a shortest possible route to at least one of the data collector nodes.

4.4.4 OPEN ISSUES IN CALEE ROUTING PROTOCOL

Application of the *CaLEe* routing protocol is expected to result in an increased lifetime Φ^{WSN} of the WSN, because of the reduction in average path length from source to sink (η). However following questions are still open which we will discuss in Chapter 5 during the evaluation of the *CaLEe* routing scheme compared to current state-of-the-art *(SS and MS)*,

- Sink mobility and data collection at *Dcs* leads to high end-to-end data latency compared to the case of a static sink, how *CaLEe* routing protocol can address this issue?
- What should be the mobility trajectory of the sink? How the mobility trajectories of a sink effect the performance of *CaLEe* routing scheme?
- How the numbers of sectors in a WSN effect the performance of *CaLEe* routing scheme?

On the other hand, from implementation point of view, *CaLEe* can face one major limitation that is finite buffer capacity of the data collector nodes. Since the *CaLEe* routing protocol is based on sink mobility for data collection from *Dc's* therefore it is possible that the time elapsed between two consecutive visits of a sink to given *Dc* become so large that the data loss starts due to congestion and buffer overflow at the data collector node.

In order to address the issue of congestion and data loss we developed an in-network storage model for data persistence in WSN.

4.5 DATA CONGESTION AND DATA LOSS IN WSN

Before we go into the details of our in-network storage model in the following we discuss current state-of-the-art used to avoid and control congestion in WSN.

4.5.1 STATE-OF-THE-ART CONGESTION AVOIDANCE AND CONTROL TECHNIQUES

Chen et al. divided the techniques developed to address the problem of data congestion in WSN into two groups: congestion avoidance and congestion control [123]. The former focuses on strategies to avoid congestion from happening and the latter works on removing congestion when it has occurred. In the following divide the state-of-the-art congestion avoidance and control schemes into three main groups that: data aggregation techniques, multi path routing techniques, and flow control techniques.

4.5.1.1 Data aggregation techniques

These techniques focus on utilizing spatial or temporal correlation between sensed data to reduce its quantity and hence prevent congestion [67] thus these schemes belong to congestion avoidance class. They are especially useful in environmental or remote area monitoring applications where consecutive data readings from the sensor nodes do not vary much over the time. Aggregation schemes that exploit such type of correlation amongst the sensed data are called temporal aggregation schemes [124]. Other types of aggregation schemes are known as spatial aggregation schemes. Sensor nodes implementing spatial aggregation schemes try to find correlation amongst the data received from different sensor nodes in an effort to reduce data size and hence avoid congestion. Galluccio et al. showed that the use of spatial aggregation can be very helpful for avoiding congestion in the vicinity of the sink [69].

Barton et al. tried to improve the data aggregation rate by applying a cooperative communication technique, where multiple nodes in a network cooperate to send data to the sink [125]. There scheme is based on clustered sensor node, when a node wants to transmits data to the sink it first send it to all the member cluster nodes. Then the nodes in a cluster cooperate to broadcast this data to the sink by synchronously transmitting a stream of identical data. However, their scheme requires cross layer design for routing, scheduling and communication protocols to achieve preeminent results. Gao et al. presented a tree based data aggregation scheme to connect the sensor nodes that are sparsely located at high activity regions in the sensor field [68]. Each node routes data up the tree, which is aggregated at every hop thus achieving data reduction. Yoon et al. introduced an aggregation technique that utilizes both the spatial and temporal correlation amongst the sensed data to setup clusters of nodes [126]. If a set of nodes is sensing values within a certain threshold then they form a cluster. It is only possible if these nodes are spatially correlated. The cluster lasts as long as the reported values from the nodes are with in defined threshold. Advantage of this scheme is that sensed values of the nodes are highly correlated and hence can be aggregated. As a result, only one value per cluster is transmitted to the sink.

4.5.1.2 Multi path routing techniques

The mechanism of establishing multiple paths between a source and destination is termed as multi-path routing. It has two major advantages. One it helps to avoid data congestion by splitting the traffic from source to sink along multiple routing paths. Secondly, it can be used to increase the probability of successful data transfer from source to sink by transmitting multiple copies of data along different routing paths thus increased accuracy is achieved at the expense of increased energy dissipation.

This idea is exploited in [127], when a routing node senses increased data traffic and data packets start to drop, it requests the neighboring nodes to become part of the routing scheme, thus creating a multi path routing topology to share the data traffic and eliminate congestion from the network. Zhu et al. studied multi path routing schemes in the context of the trade-off between the network lifetime and end-to-end data latency. Although they do not address the issue of congestion directly, but their work can be used to estimate the overhead caused by multi-path routing in terms of increased energy dissipation from the nodes [128].

Vidhyapriya et al. proposed to split the routing load along different paths from source to sink with proportion to the residual energy of the nodes located at these paths [129]. This helps in balancing the energy dissipation from the nodes thus resulting in increased lifetime of the network. Based on the obtained results authors also made following comment in their paper *"multi-path routing is cost effective for heavy load scenario, while a single path routing scheme with a lower complexity may otherwise be more desirable"* [129]. The rationale behind this statement is that in the case of heavy routing load scenarios it is worth spending some additional energy to distribute the routing load along multiple paths as doing so results in increased lifetime of the network. In the case of low routing load scenarios, often the effort required to setup multiple routing paths is higher than the achieved energy gains.

The following two rules provides the base for a very well known routing scheme, called Directed diffusion [32]. First, routing paths are only established when they are required, meaning when a node has some data to transmit only then it initiates the route establishment mechanism. Second, in order to achieve robustness a periodic low-rate flooding is used to maintain alternate routing paths to the sink. In order to achieve energy efficient recovery from route failure authors in [130] suggest the construction of two types of multi paths, disjoint multi path (alternate paths are node disjoint with primary path) and braided multi path, which is composed of several partially disjoint multi path schemes.

4.5.1.3 Flow control techniques

Flow control techniques try to control the amount of data that is flowing on the routing path to control congestion. Wan et al. [131] implemented a back pressure mechanism to restrict data flow, Chen et al. [123] and Akan et al. [132] allow the receiver node to regulate the outflow of data from the sender node.

Wang et al. [133] proposed a node priority based congestion control scheme for wireless sensor networks. Their scheme is based on the assumption that the nodes located in a WSN have different bandwidth and wireless media control requirement for data transmission. Therefore, a node priority index can be generated on the basis of packet inter arrival time and service time at each node. With the help of this index, sensor nodes having heavy data traffic can be assigned more access to the transmission media than the nodes with less traffic. However, additional overhead is involved in maintaining the priority index of the sensor nodes.

Chen et al. [123] presented a congestion avoidance scheme based on the idea that at any given point of time a node (client node) on a routing path has complete information about the buffer status of the node (parent node, data forwarding node of the client node) located down the stream on the routing path. Therefore, in case of congestion the client node either reduces the data that it forwards to the parent node or switches to some other parent node.

In general, all schemes belonging to this class restrict the flow of data, which results in data loss due to buffer overflow at the sensor nodes who cannot forward the collected/received data.

4.5.1.4 Summary

Current state-of-the-art for congestion avoidance and control can be summarized as follows.

Aggregation based techniques entirely depend on finding correlation amongst the sensed data. If the collected data is of diverse nature (possibly due to highly dynamic WSN) then aggregation based techniques will not be of much use. Thus, when it is necessary to report all the events to the sink aggregation based techniques fail provide a reasonable solution.

Multi path routing techniques tend to work well for controlling congestion and avoiding data losses, but they fail to avoid congestion in the vicinity of the sink caused due to the *funneling effect* (Many to one transmission). Furthermore, it also leads to increased energy dissipation in an effort to maintain multiple routing paths to the sink.

Flow control techniques are mostly based on controlling the data flow. When a node senses data congestion it transmits a reduce data rate request to nodes forwarding data to it. Thus, applying a sort of back pressure to its up stream neighbors. The node which receives this back pressure message will repeat the process till it reaches leaf sensor nodes on a routing tree. This implies the leaf node can no more forward the gathered data at desired rate. On the other hand, new data is being periodically sensed by the leaf nodes, thus in order to make some space in the buffer to accommodate newly collected data leaf nodes start to drop previously stored data resulting in data loss.

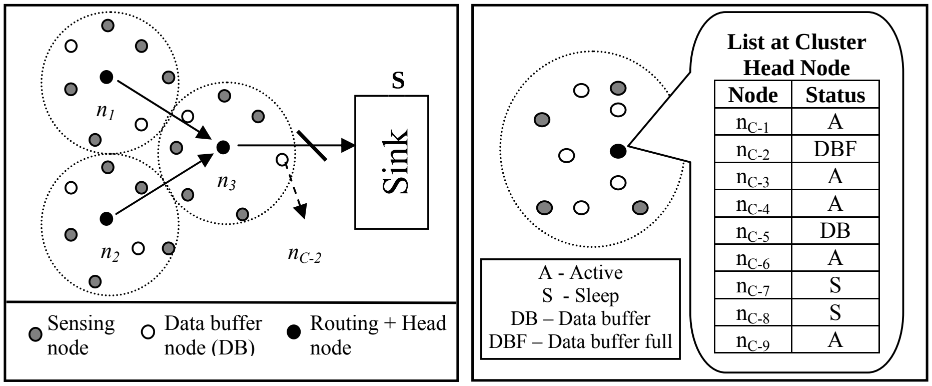
In summary, existing schemes have high potential for data loss. In order to overcome this particular problem, an in-network storage model has been developed.

4.5.2 IN-NETWORK STORAGE MODEL FOR DATA PERSISTENCE IN WSN

This section presents the in-network storage model that is designed to achieve data persistence in WSNs. The in-network storage model neither removes nor avoids congestion from happening, but it ensures data persistence under congestion and localizes its effects.

The in-network storage model is based on a clustered sensor field [134], whereas each cluster is comprised of number of sensor nodes and a cluster head node. Cluster head node is responsible for data collection from the member cluster nodes and forwarding this data towards the sink. It has been observed that under dense deployment of the sensor nodes only a subset of the nodes is needed to achieve complete coverage (every point in the field is covered by at least one node) of the sensor field. Therefore, redundant sensor nodes are set to “sleep mode” by the cluster head nodes. This helps to increase the lifetime of the WSN and to avoid congestion by reducing the amount of data flowing along the routing paths towards the sink. The basic idea of the in-network storage is to utilize these redundant nodes (sleeping nodes) located in the vicinity of routing nodes (head node) as data buffers to avoid data loss from congestion in WSNs.

In order to better understand the idea, consider Figure 4.11(a), which shows a clustered sensor field and Figure 4.11(b) that presents a detailed view of a cluster of nodes. It can be seen from Figure 4.11 that the nodes in a cluster are divided into two groups. One is the group of active sensor nodes, which collect data from the field, and the other is the group of sleeping sensor nodes. Since the head node is managing all the sensor nodes in a cluster, it maintains a list of all the cluster member nodes along with their status as shown in Figure 4.11(b).



Node	Status
n_{C-1}	A
n_{C-2}	DBF
n_{C-3}	A
n_{C-4}	A
n_{C-5}	DB
n_{C-6}	A
n_{C-7}	S
n_{C-8}	S
n_{C-9}	A

(a): Clustered WSN (b): Detailed view of a cluster

Figure 4.11 Clustered sensor field with data congestion at node N

Assume that the routing node n_3 in Figure 4.11(a) fails to forward the collected data to the sink because of a temporary out-of-sight problem or because the sink is busy collecting data from other neighboring nodes. As a result, congestion starts to build at node n_3. In order to avoid data loss and to localize the effect of congestion in the vicinity of the node where congestion has occurred, node n_3 (which is also cluster head node) utilizes the in-network storage model and consults the list of client nodes shown in Figure 4.11(b). This list of neighbors provides information about the sensor nodes that can be used as data buffers. When the buffer at node n_3 reaches a certain threshold limit, then n_3 selects a buffer node (e.g., n_{C-5} shown Figure 4.11(b)) and starts to redirect the arriving data from n_1 and n_2 to the data buffer node. When a data buffer node becomes full to its capacity then it sends a BUFFER FULL message to the cluster head node. Then the head node marks this node as DBF and selects another sleeping node as data buffer. As a result, data loss can be avoided and the affect of congestion remains localized. Later on, when the forward link gets clear, the data can be retrieved from the buffer nodes and forwarded towards the sink by n_3.

Cluster head nodes (routing nodes) can deploy two schemes to redirect the arriving data to the buffer nodes. One is to route the incoming data through the cluster head node towards the data buffer node. In this case no client node is allowed to communicate with any other node outside the cluster boundary. Therefore, all the incoming data is routed to appropriate buffer node via head node. Figure 4.12(a) shows one such scenario where the node n_1 is sending the data to node n_3, which is then routed to buffer node n_{C-4} (by the node n_3) to avoid data loss due to congestion at forward link n_3S.

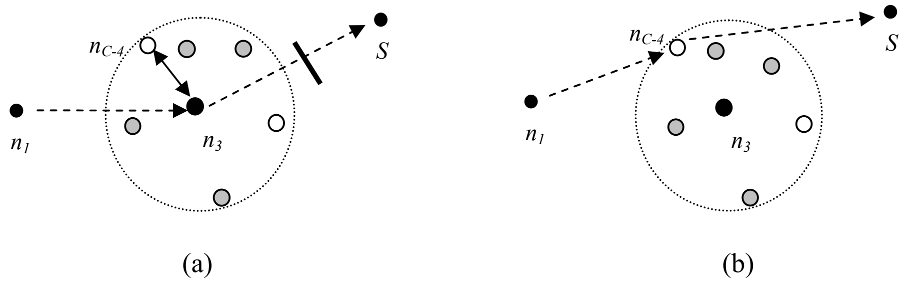


Figure 4.12 Data storage and retrieval from the buffer node

The second option is to establish a direct connection between the data sender and the data buffer node, if they are within the communication ranges of each other. In this case, on detecting congestion the routing node n_3 first queries all the sleeping nodes about their list of neighbors and buffer capacities. If a buffer node (e.g., n_{C-4}) is located within the communication range of the data sender node n_1 then a direct link is established between n_1 and n_{C-4} by the routing node n_3 as shown in Figure 4.12(b). For this purpose, the node n_3 transmits a message to n_1 informing about the new routing path n_1n_{C-4} and the maximum amount of data that can be routed to the node n_{C-4} depending on n_{C-4}'s buffer capacity.

The process of retrieving the stored data from the buffer nodes is similar to the process of storing the data. If the buffer node is located within the communication range of the next routing node S, then n_{C-4} is directed by the head node to send the stored data directly to S. Otherwise, the head node retrieves the data from n_{C-4} and forwards it to the node S, which potentially increases data latency and energy loss.

4.6 SUMMARY

This chapter has introduced a new routing strategy for WSNs that is based on virtual partitioning of the sensor field into sectors and sink mobility along a fixed trajectory. In addition, an algorithm for virtual partitioning of the sensor field in an energy efficient manner is also presented. However, following questions are still open in *CaLEe* routing scheme and will be discussed in Chapter 5:

- Sink mobility and data collection at *Dcs* leads to high end-to-end data latency compared to the case of a static sink. How *CaLEe* routing protocol can address this issue?
- What should be the mobility trajectory of the sink? How do the mobility trajectories of a sink effect the performance of *CaLEe* routing scheme?
- How does the number of sectors in a WSN effect the performance of *CaLEe* routing scheme?

The *CaLEe* routing protocol is based on data collection at the data collector nodes that have limited buffer capacity. In order to avoid data loss and to ensure data persistence the in-network storage model is also presented.

CHAPTER 5. PERFORMANCE EVALUATION OF THE CALEE ROUTING PROTOCOL

5.1 RESEARCH CONTRIBUTION

In Chapter 4, we introduced the *CaLEe* routing protocol that leads to an increase in the lifetime Φ^{WSN} of a WSN by reducing the average path length that data has to take from source to sink and evenly balancing the routing load amongst the nodes in a WSN. In this chapter, we compare the performance of our newly developed routing protocol with two standard routing configurations. Static sink placed at the center of the sensor field (*SS*) and mobile sink moving along a fixed trajectory (inside a WSN) in discrete fashion (*MS*).

Our initial discussion focus on two principal parameters that constitute the *CaLEe* routing protocol: *"sink mobility radius*" (distance of the mobility trajectory of the sink from the center of the sensor field), and *"number of sectors"* in a WSN. Later, we will analyze how parameters such as communication range *r* of the nodes, sensor node density Γ, throughput *G* of the WSN and size of the sensor field affect the performance of *SS*, *MS* and *CaLEe*.

Our results show that compared to *SS* and *MS* the *CaLEe* routing scheme results in minimum average energy dissipation Ψ per node. Furthermore, for small communication range *r* of the nodes, low sensor node density Γ and high throughput *G* of the WSN *CaLEe* provides lower average end-to-end data latency $\overline{\Delta}$ compared to *SS* and *MS*. It has also been shown that with an increase in the size of the field performance gains of *CaLEe* over *SS* and *MS* increases.

5.2 SYSTEM DESCRIPTION AND ASSUMPTIONS

In order to simulate *CaLEe* and make a comparison with other state-of-the-art routing schemes (*SS, MS*) we used a custom-built discrete-time simulator written in C++ from Michele Garetto. Much of the system description and assumptions discussed in the following (Section 5.2 and few sub sections in 5.3) are reproduced from his paper [135].

We consider a network composed of *N* stationary, identical sensor nodes that are uniformly distributed over a disk in the plane. Each node has a common fixed communication range *r* and is equipped with omni-directional antennae; thus, any pair of nodes can communicate if they are up to distance *r* from each other. A mobile sink is assumed to be positioned at the center of the disk, responsible for gathered data from the nodes. However, sink can move to any desired location in the field. In addition, we consider such a network topology that for any sensor node there exists at least one routing path, which connects sensor to the sink.

Every node organizes the collected data into data units of fix size, which it stores in a buffer of infinite capacity; the buffer is modelled as a centralized FIFO queue. It is assumed that a sensor node cannot transmit and receive at the same time therefore the time is divided into slots. In addition, transmission/reception of each data unit takes one time slot and the wireless channel is assumed error-free.

5.2.1 SENSOR BEHAVIOR

Three major operational requirements that cause energy dissipation from a sensor node are sensing, communication, and data processing [2]. In Section 2.1.5.8, we have identified various operational states for a sensor node to reduce its energy dissipation. However, in order to reduce the complexity in this simulation we consider only two major operational states: *active* and *sleep*. The *sleep* state is the least power consuming state where a node cannot perform any task (transmission, reception or data processing). Being in *active* state a sensor node will be performing one of the following operations: *transmission*, *receptions*, or *idle*. Thus, temporal evolution of the sensor state can be depicted in terms of a cycle that is shown in Figure 5.1. Each cycle comprises a sleep phase *(S)* and an active phase *(A)*. After a transition from sleep to active phase, the node calculates a time period after which it should go back to the sleep mode. The scheduled periods of sleep and activity are modelled as random variables geometrically distributed with parameter *q* and *p*, respectively. It has been shown in [135] that above mentioned framework encapsulates the behavior of sensors switching between sleep and active states that is the case in real time systems [136, 137].

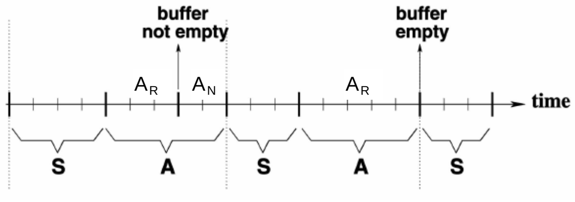


Figure 5.1 Temporal evolution of the sensor state [135]

Each sensor maintains a routing table having a maximum of Λ routes, each route corresponds to a different next-hop neighboring node. In order to handle heavy routing load [138], a sensor can prolong its active state. When a node wants to transit from active to sleep mode it can only be done if the buffer is empty. Therefore, the node will extend its active phase to forward all the data to one of the next-hop neighboring node. However, during the extended active phase sensor neither accept relaying data nor produce data to quickly make a transition to sleep phase. Figure 5.1 highlight the sleep/active cycle of the sensor nodes where the active phase can be sub-divided into phase A_R and A_N (if required). Phase A_R corresponds to scheduled active phase where a node can receive, transmit as well as generate data units with fixed time intervals (follows a Bernoulli process with parameter *g*). On the other hand, during phase A_N sensor node either forwards the data or wait in idle state for an opportunity to forward the data. Finally, in order to ensure end-to-end connectivity for each node (from source to sink) during simulations, it is assumed that nodes neither fails nor run out of energy thus leading to a stationary topology.

5.2.2 DATA ROUTING

After the execution of the boundary identification algorithm *(MoSBoD)* and partitioning/route establishment modules of *CaLEe*, each sensor in the WSN has information about its neighboring nodes, as well as about the possible routes to the sink or data collector nodes. Since a network of stationary nodes performing, for instance, environmental monitoring and surveillance, is considered, the routes and their conditions can be assumed either static or slowly changing, consequently, the overhead incurred due to routing path management (caused due to sensor node failures) can be neglected.

We associate a certain energy cost with each of the Λ routes maintained by a sensor node. For the generic route ρ, the energy cost $e(\rho)$ is computed as follows: Given a node $n_i \in \rho$ and $\sigma_\rho(n_i)$ denotes the node immediately succeeding n_i on ρ. Here the route includes the source and the relays but not the sink because it is assumed that the sink has unlimited energy resource compared that of the sensor nodes. We have:

$$e(\rho) = \Sigma_{n_i \in \rho} E_{n_i,\sigma_\rho(n_i)} = \Sigma_{n_i \in \rho} (E^{tx}_{n_i,\sigma_\rho(n_i)} + E^{rx}_{\sigma_\rho(n_i)})$$

$E_{n_i,\sigma_\rho(n_i)}$ is the energy required to transfer a data unit from node n_i to its next-hop neighboring node. $E_{n_i,\sigma_\rho(n_i)}$ is equal to the sum of the energy spent by n_i in transmission $E^{tx}_{n_i,\sigma_\rho(n_i)}$ and energy $E^{rx}_{\sigma_\rho(n_i)}$ spent by $\sigma_\rho(n_i)$ in reception. $E^{rx}_{\sigma_\rho(n_i)}$ is the sum of energy dissipation caused by transceiver electronics E^{ele} and processing functions E^{proc}; while $E^{tx}_{n_i,\sigma_\rho(n_i)}$ is the sum of E^{ele} and E^{proc}, as well as the energy dissipation in the amplifier, that is proportional to the squared distance between transmitter and receiver [139]. Thus, we rewrite $e(\rho)$ as,

$$e(\rho) = \sum_{n_i \in \rho} [2(E^{ele} + E^{proc}) + d^{\chi}_{n_i,\sigma_\rho(n_i)} E^{amp}]$$

E^{amp} is a constant value, $d_{n_i,\sigma_\rho(n_i)}$ is the distance between n_i and $\sigma_\rho(n_i)$, and χ is the exponential decay factor that vary between 2 and 4. However, during simulation runs we do not consider an online power control mechanism, as low cost sensor nodes do not support it.

Considered simulation model which we borrowed from [135] is general enough to deal with different routing schemes. However, here we consider the following strategy: when a sensor wants to transmit a data unit it will always give priority to those next-hop neighboring nodes, which have lowest energy cost associated to them. The process continues until either an available next-hop neighboring node is found that is ready to receive the data or the transmitter node has polled all of the next-hop neighboring nodes listed in its routing table. Thus, it can be inferred that the selected routing strategy favors energy efficiency over data latency.

5.2.3 CHANNEL ACCESS

It is assumed that sensors employ a CSMA/CA mechanism (refer to Section 2.1.5.6) with handshaking. When a node n_i wants to transmit to n_j $(1 \le n_i \le N, \mathit{and}\ 0 \le n_j \le N, 0 \text{ indicating the sink})$ it will try to access the channel. If the channel is free then node n_i sends a transmission request to n_j and waits till either a response is received that n_j is ready to receive or the timer expires. In the former case, n_i sends the data to n_j; in the latter case, n_i will poll the following next-hop neighboring node. Note that if in the meanwhile a neighboring node of n_i, say n_k, accesses the channel by polling node n_l, n_i will detect the channel as busy only if it hears a reply from n_l; otherwise, after a timeout, n_i will consider the channel as idle.

It is known from [138] that energy dissipation caused by protocol overhead is very low therefore, during our simulations it is assumed that the transmission of each data unit takes one time slot, which also includes the polling phase (refer Section 2.1.5.6). To conclude, the model accounts for channel contention, however, data transmissions are assumed collision-free.

5.3 EVALUATION MODEL

At the beginning of each simulation, a random topology is generated such that when all sensors are active the field is connected at the physical layer. Then each node selects the best Λ routes based on the energy cost defined in Section 5.2.2. To simulate the ideal MAC protocol described in Section 2.1.5.6 and Section 5.2.3, following strategy was adopted: At the beginning of each time slot, all sensors can potentially transmit or receive data units during the slot. To solve the channel contention issue, a random permutation of the indexes 1, 2, . . . , N associated to the sensor nodes is extracted. Then, based on the order resulting from the permutation if a node is able to transmit a data unit it is allowed to do so. Furthermore, because of the CSMA/CA mechanism (refer to Section 2.1.5.6) with handshaking, no other node within the communication range of the transmitting sensor node is allowed to receive until the beginning of the next slot. On the other hand, no node within the communication range of the receiver node is allowed to transmit during the current slot. As a result, transmissions in the network can be selected one can fairly during each time slot.

5.3.1 EVALUATION METRICS

Lifetime Φ^{WSN} of a WSN, average energy dissipation Ψ per node and average end-to-end data latency $\overline{\Delta}$ are three metrics, which we will analyze during the evaluation of *CaLEe* compared to current state-of-the-art data routing protocols.

We defined the lifetime Φ^{WSN} of a WSN as the time until the first node runs out of energy. On the other hand, average energy dissipation Ψ per node is calculated as, sum of the energy dissipations from all the nodes divided by the total number of nodes in WSN. Hence, Φ^{WSN} is bounded by maximum energy dissipation by a node in a WSN while, Ψ captures a broader picture by providing average energy dissipation per node in a WSN.

Note that energy dissipation from a sensor node can be of three types that are energy dissipated at the transceiver E^{ele}, energy spent during transition from sleep to active mode E^{t} and energy disseminated by the node during idle active periods of the nodes. Therefore, in order to keep the simulation model as close as possible to reality total energy dissipation of a sensor node is calculated as the sum of these three types of energy dissipations as discussed in Section 5.2.2.

Average end-to-end data latency $\overline{\Delta}$ in a WSN is equal to the sum of the delays experienced by all messages from source nodes to the sink divided by total number of messages. On the other hand, delay experienced by a message per hop is the sum of the transmission delay, delay occurred in acquiring the channel due to contention and delay due to duty cycling of the sensor nodes.

5.3.2 EVALUATION PARAMETERS

During performance evaluation we focused on following primary parameters,

1. *Sink mobility radius:* distance of the mobility trajectory of the sink from the center of the sensor field
2. *Number of sectors:* total number of virtual partitions of the sensor field

Furthermore, the following network parameters are also studied for comprehensive evaluation and comparison of the three routing schemes *SS*, *MS* and *CaLEe,*

1. *Communication range r of the nodes:* maximum Euclidean distance up to which a node can communicate with other node.
2. *Data rate g of a node:* number of data packets generated by a node per time slot
3. *Duty cycle of a node:* ratio of the time that a node spends in active and sleep mode
4. *Node density* $\Gamma = N\pi r^2 / \pi R^2$
5. *Size of a WSN:* total covered area of the sensor field

Data rate and duty cycling of the nodes are used to derive a new parameter that is the throughput *G* of a WSN. We define the throughput *G* as, the total number of data packets generated in a WSN per time slot.

$$G = Ngp/(p+q)$$

Where, *N* is the total number of nodes in a WSN, *g* is the data generation rate of the nodes and *p/q* are parameters for the sleep/active transition, which follow a geometric distribution (refer to Section 5.2.1). Note that *G* represents the sum of the throughputs of all sensor nodes and only includes parameters that are in input to the system model. In our simulation model, we assumed low cost sensor nodes that have simple transceiver sub-system. Also all the nodes use same frequency for communication. As a result, at any given time the sink can communicate to only one node and exchange only one data packet per time slot. This limitation is depicted in the simulation model as maximum theoretical throughput of the network that cannot exceed 1 (the sink cannot receive more than one data unit per time slot), so it seems reasonable to limit the network throughput *G* to the interval $(0,1]$. Now suppose we have a sensor field with *N=2000*, nodes are operating at 1% duty cycle which implies *p=0.99* and *q=0.01* and we want to achieve G=0.8. We can use equation 5.1 to calculate the value for *g*. Thus, we can configure the nodes to provide us desired throughput using input parameters of the system model.

5.3.3 EVALUATION ENVIRONMENT

We set up following simulation environment for studying lifetime Φ^{WSN} of a WSN, average energy dissipation Ψ per node and end-to-end data latency $\overline{\Delta}$ of the *CaLEe* routing protocol in comparison to the state-of-the-art *SS* and *MS*.

We considered reasonable sized sensor field having *R*=800 meters where *N=2000* micaz nodes are randomly deployed to achieve average sensor node density Γ of 20 node. Such high density is required during random deployment of the nodes to ensure that each node has a route to the sink. Furthermore, we set communication range *r*=80 meters (maximum communication range of the micaz motes is 100 meters), $\chi = 2$, $E^{amp} = 0.057mJ/slot$, $E^{ele} = E^{proc} = 0.24mJ/slot$, $E^{sleep} = 300nJ/slot$, and $E^{t} = 0.48mJ$ (these values are typical for the sensor nodes from crossbow). Throughput *G=0.3*, low value for *G* is configured to minimize the probability of data queuing and hence alteration in the scheduled duty cycle of the nodes (refer to Section 5.2.1). Since our work focuses on large-scale applications of WSN such as, environment or habitat monitoring where energy efficiency of a WSN is always a great concern, therefore nodes are configured to operate at 1% duty cycle.

It is known from [140] that in the case of a circular sensor field maximum lifetime of a WSN can be achieved if radius of the mobility trajectory of the sink is set to $\sqrt{2}R/2 \cong 0.7 * R$. Therefore, for initial experimentation we set the mobility trajectory of the sink at $(\sqrt{2} * 800)/2 \cong 560$ meters from the center of the sensor field and $V_{sink}=2$ m/s (7.2 km/h) (refer to Section 2.1.2 and 3.3.1), it is the speed which robots such as, BigDog (6.4 km/h) will soon achieve.

For *CaLEe* routing protocol it is assumed that the field is partitioned into eight sectors.

5.3.3.1 End-to-end data latency in CaLEe routing protocol

In the *CaLEe* routing protocol high end-to-end data latency is caused by slow mobility speed of the sink and is dominated by the time taken by the sink to complete a trip along its mobility trajectory and sojourn time of the sink at each data collector node.

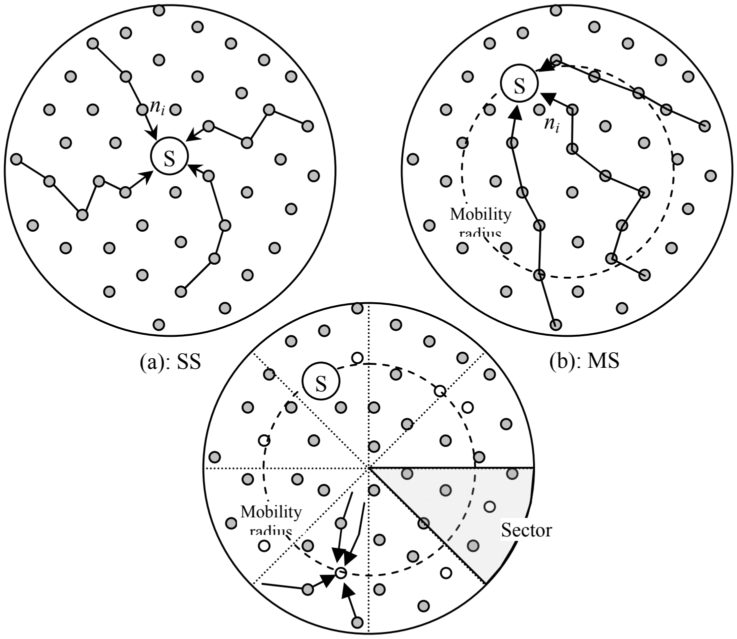
The time taken by the sink to complete a trip along its mobility trajectory can be calculated easily using length *L* of the mobility trajectory $= (2\pi \frac{\sqrt{2}}{2} R) = \sqrt{2}\pi R \approx 3552m$ and the mobility speed of the sink $V_{sink} = 2m/s$. In order to calculate the sojourn time S_{sink} of the sink at each data collector node we consider a WSN comprised of Micaz nodes that have a data buffer capacity of 512 Kbytes and a data transmission rate of 250 kbps. Now assuming the data buffer of the collector node is always full when the sink arrives, then the time required by the data collector node to transfer all the data to the sink equals 512Kbytes/250 kbps = 16s = S_{sink}. So, total time required by the sink to complete a trip along its mobility trajectory equals $(L/V_{\sin k})+(N_{\sec tors}*S_{\sin k})$. We use this formula to calculate the maximum end-to-end data latency $\Delta_{\max}$ experienced by a node. For example in considered WSN having $R=800m$, $N_{sectors}=8$, $V_{sink}=2m/s$ and radius of the mobility trajectory of the sink equals $0.7*R$ maximum end-to-end data latency $\Delta_{\max}$ experienced by a node equals $(L/V_{\sin k})+(N_{\sec tors}*S_{\sin k})=(3552/2)+(8*16)=1904s$. On the other hand, nodes located in the vicinity of data collector nodes will experience minimum end-to-end data latency $\Delta_{\min}\approx 0$ when the sink is positioned at the corresponding data collector nodes. So, average end-to-end data latency $\overline{\Delta}$ experienced by the nodes equals $(\Delta_{\max}+\Delta_{\min})/2=952s$. Similarly, $\overline{\Delta}$ equals *692s* and *440.8s* for mobility trajectories equals *0.5*R* and *0.3*R* respectively.

5.3.3.2 End-to-end data latency in SS and MS

End-to-end data latency in *SS* and *MS* depends on two parameters: maximum hop count distance of a node from the sink and time taken by a message to travel one hop *(t_{1hop})*. Maximum hop count distance depends on the size of the sensor field and communication range of a node (size of a WSN / communication range of a node). On the other hand, t_{1hop} greatly depends on the functionality of the implemented MAC protocol (refer to Section 2.1.5.6). By default Micaz nodes are installed with BMAC protocol. It is known from Section 2.1.5.8 that in order to achieve 1% duty cycle under BMAC polling time must be set to 240ms, which means each node will sense the medium for traffic after every 240ms. As a result, considering no channel contention $t_{1hop}=240ms$. In our simulation, one time slot is the time required by a node to transmit or receive a message therefore we set one time slot equal to 240ms.

5.3.4 EVALUATED SCENARIOS

In the first simulation scenario, we place the sink at the center of the sensor field and all nodes are programmed to route data directly to the sink as shown in Figure 5.2(a) (a typical *SS* case). Figure 5.2(b) presents the second simulation scenario where static sensor nodes are uniformly but randomly placed in the field, which route their data directly to the sink. However, the sink is mobile and follows a fixed trajectory inside the WSN in a discrete fashion (stop-and-go) [117, 118]. Furthermore, we assume that the sojourn time of the sink is higher than the mobility time (the sink only moves when the energy level of the nodes in its vicinity falls below certain pre-defined threshold).



(a): SS

(b): MS

(c): CaLEe

Figure 5.2 Simulation set up of a WSN

As a result, overhead[1] incurred due to routing path updates caused by the sink mobility can be neglected. Figure 5.2(c) depicts the third scenario for the *CaLEe* routing protocol where sensor field is partitioned into $N_{sectors}$=8 sectors each having ($N/N_{sectors}$) 2000/8 = 250 nodes and a data collector node which is positioned close to the mobility trajectory of the sink, as explained in Chapter 4.

5.4 PERFORMANCE RESULTS

In the first phase, we analyze how the implementation of *CaLEe*, *SS* and *MS* routing schemes effects the energy dissipation and end-to-end data latency experienced by each node in a WSN. We consider the simulation scenario presented in Section 5.3. For each protocol, we placed the sink at the most energy efficient location known from the literature. In *SS* the sink is placed at the center of the sensor field, in *MS* it has been recognized that maximum lifetime of the network can be achieved if the mobility trajectory is set close to $\sqrt{2}R/2 \cong 0.7*R$ [140]. Therefore, for both *CaLEe* and *MS* routing schemes we mobilize the sink around the circle with radius $\sqrt{2}R/2 \cong 0.7*R$. In the case of *CaLEe* we assume that the field is partitioned into eight sectors.

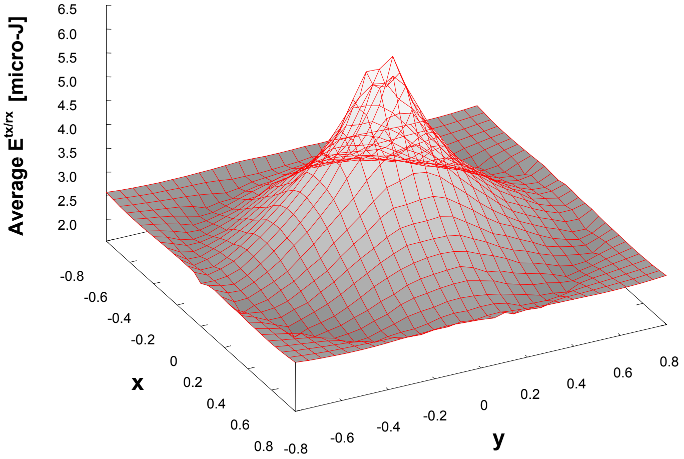
---

[1] Easiest way to update the routing paths is to initiate a flood message from the sink once it reaches desired new location. Overhead is this case equals energy spent by each node to relay the flood message till it reaches all nodes.

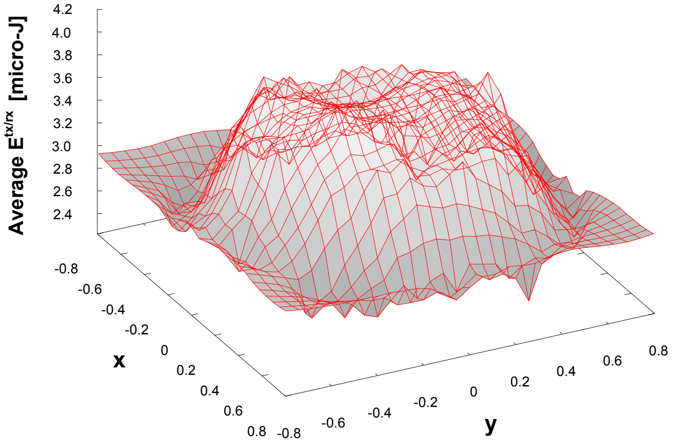
Placement of the sink in WSN has high influence on the data-relaying load of individual sensor node. For example, node located in the vicinity of the sink experience high relaying load and hence average $E^{tx/rx}$ ($=E^{tx} + E^{rx}$, where E^{tx} is the average energy dissipation by a node in transmitting messages and E^{rx} is the average energy dissipation by a node in receiving messages) energy dissipation of these nodes will be high compared to rest of the sensor field. Figure 5.3 plots x, y coordinates of each node in a WSN and its average $E^{tx/rx}$ energy dissipation for *SS, MS* and *CaLEe*. It can be anticipated that average $E^{tx/rx}$ by any single node will be maximum in the case of *SS* followed by *MS* and then *CaLEe*. Our intuition is based on simple fact that due to static position of the sink in *SS* nodes located in the vicinity of the sink experience high relaying load (forwarding data packets from rest of the WSN to the sink) and hence deplete their energy much faster compared to rest of the sensor field as shown in Figure 5.3(a). On the other hand, in the case of *MS* sink periodically changes its position in the WSN. As a result, heavy relaying load experienced by the nodes located in the vicinity of the previous position of the sink shifts to the nodes located in the vicinity of the new position of the sink. Thus, in comparison to *SS* $E_{\max}^{tx/rx}$ ($E_{\max}^{tx/rx} = \max\limits_{i \in [1,N]} E_i^{tx/rx}$, where $E_i^{tx/rx}$ is the average $E^{tx/rx}$ energy dissipation by the node i) energy dissipation by any node in a WSN is reduced. It is also known from the definition of the lifetime Φ^{WSN} of a WSN that higher the value of $E_{\max}^{tx/rx}$ lesser will be the lifetime of a WSN. So, based on our simulation results shown in Figure 5.3(a) and (b) ($E_{\max}^{tx/rx}$ in *SS* and *MS* are $6.4\mu J$ and $4.12\mu J$ respectively) it can be inferred that sink mobility improves the lifetime Φ^{WSN} of a WSN.

Compared to *MS* in the case of *CaLEe* routing scheme two factors contribute to further reduction in $E_{\max}^{tx/rx}$. One is the reduction in average path length between the source and the sink, which comes from the fact that in *CaLEe* routing scheme data from each node has to travel only to its closest data collector node. This reduces the relying load experienced by the nodes and hence reduces $E_{\max}^{tx/rx}$ compared to the cases of *SS* and *MS*. Second, by replacing a single data collection point (the sink) with multiple data collector nodes we manage to distribute the routing load from the nodes located in the vicinity of the sink to the nodes positioned in the vicinity of the eight data collector nodes. Thus, further reduction in $E_{\max}^{tx/rx}$ (as shown in Figure 5.3(c)) and hence improvement in the lifetime Φ^{WSN} of a WSN is achieved compared to *SS* and *MS*.

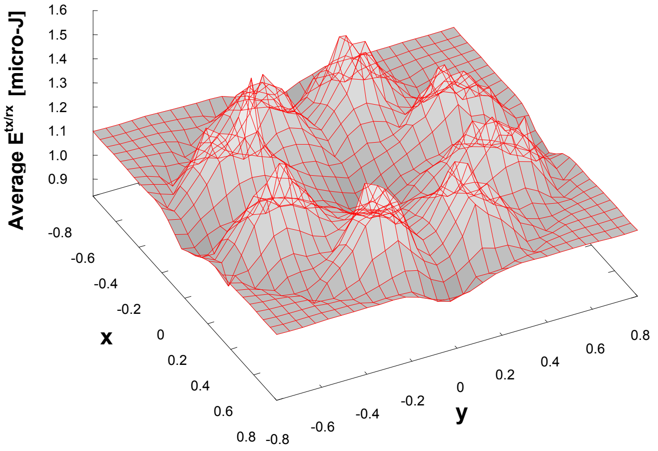
Based on our simulation results shown in Figure 5.3 it can be stated that *CaLEe* improves the lifetime Φ^{WSN} of a WSN compared to *SS* and *MS* by the factor of 4.5 and 3.25 respectively.



(a): SS



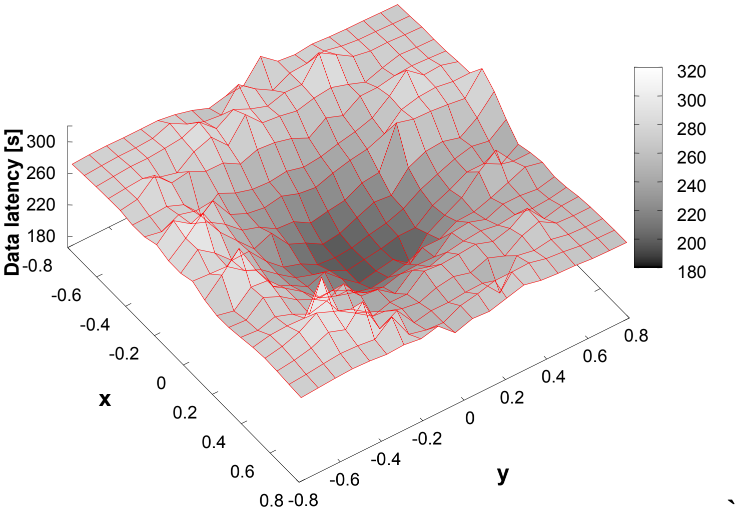
(b): MS



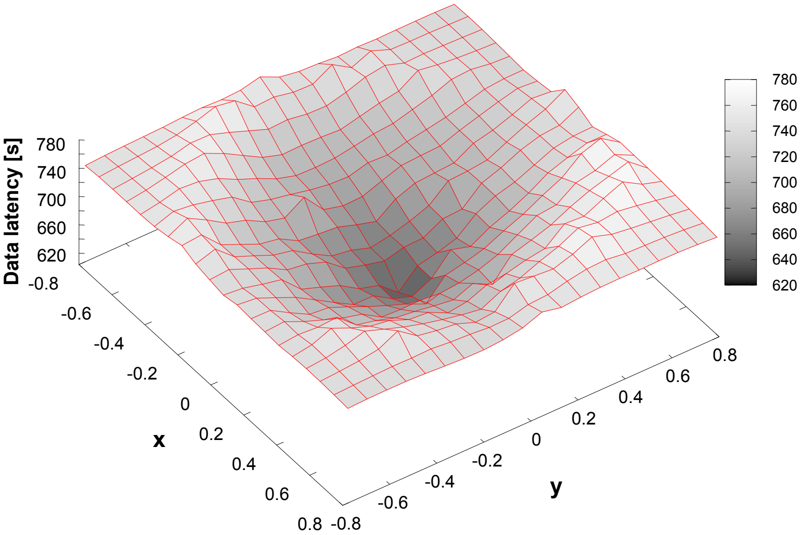
(c): CaLEe

Figure 5.3 Energy dissipation patterns per node in SS, MS and CaLEe

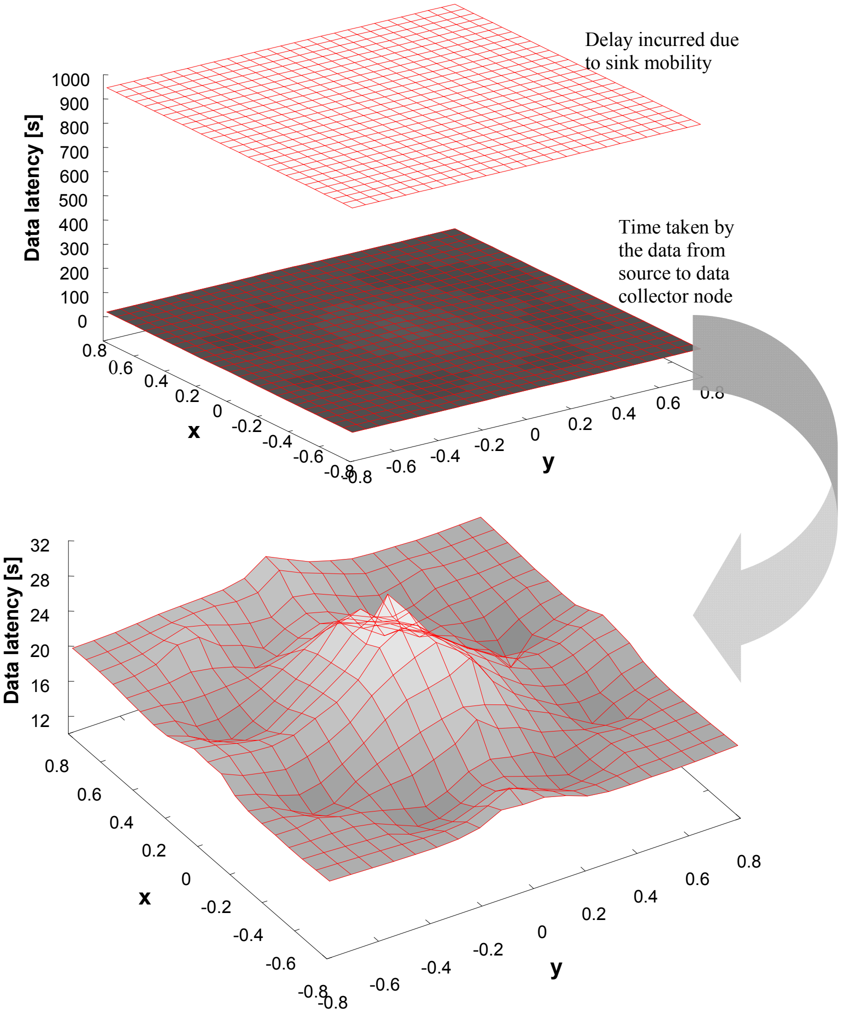
Figure 5.4 presents the position coordinates of each node *i* in a WSN and average end-to-end data latency experienced by a data packet from node *i* to the sink. It can be seen from Figure 5.4 that *SS* leads to minimum data latency for the nodes amongst the three schemes. It is due to the fact that the average path length from source to sink is less in *SS* compared to *MS,* while in the case of the *CaLEe* data reaches the sink only when the sink visits the data collector node where data has been submitted by the node *i*. In Section 5.3.3.1 we have calculated the average end-to-end data latency experienced by a node in the case of the *CaLEe* routing protocol. Figure 5.4(c) plots this data latency along with the time taken by the data from each sensor node in a WSN to its corresponding data collector node.



(a): SS



(b): MS

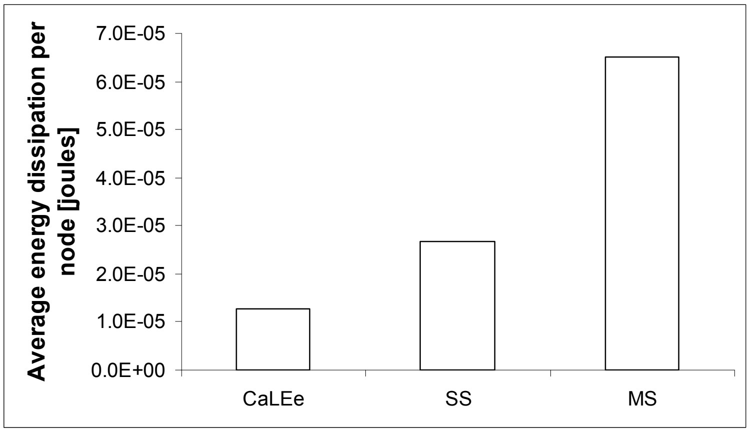


(c): CaLEe

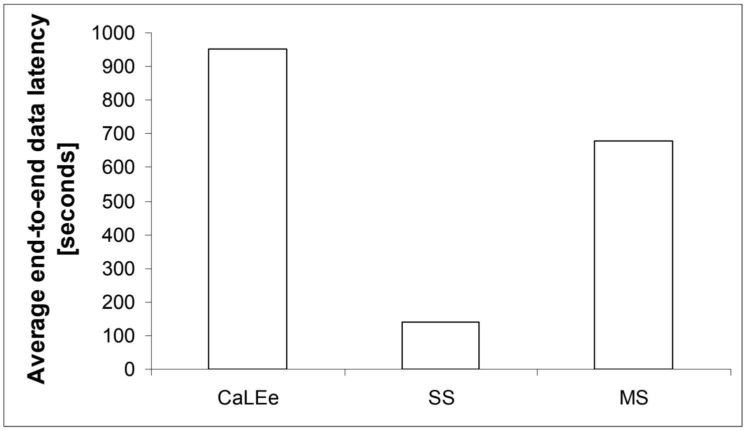
Figure 5.4 End-to-end data latency experienced by each node in SS, MS and CaLEe

Figure 5.4(c) validates our hypothesis that in the case of the *CaLEe* routing scheme delay is not caused by the node particularities but it is due to the slow mobility speed of the sink. It can also be inferred from Figure 5.4 that both *MS* and *CaLEe* have high average end-to-end data latency compared to *SS* by the factor of 2.5 and 3.5 respectively.

An interesting fact is shown in Figure 5.5(a) that plots the average energy dissipation Ψ per node $\left(= \sum_{i\in[1,N]} E_i \big/ N\right.$, where E_i is the energy dissipation by a node i and N is the total number of nodes in WSN) of the sensor field for *CaLEe*, *SS* and *MS*. We observed that the average energy dissipation per node in *CaLEe* is minimum followed by *SS* and *MS*. So, based on the results shown in Figure 5.3 and Figure 5.5(a) it can be inferred that on the average *SS* leads to lesser average energy dissipation per node compared to *MS*, but the lifetime (the time until the first node runs out of energy) of the WSN is better in the cases of *MS* and *CaLEe*. This is due to balanced and uniform energy dissipation of the sensor nodes due to sink mobility.



(a): Average energy dissipation Ψ per node



(b): Average end-to-end data latency $\overline{\Delta}$ per node

Figure 5.5 End-to-end data latency experience by each node

Average end-to-end data latency $\overline{\Delta}$ $\left(= \sum_{i\in[1,N]}\Delta_i \big/ N \right.$, where Δ_i is the average end-to-end latency by a node i and N is the total number of nodes in WSN) in *CaLEe*, *SS* and *MS* is shown in Figure 5.5(b). *CaLEe* has the highest $\overline{\Delta}$ caused due to slow mobility speed of the sink. On the other hand, *MS* has higher $\overline{\Delta}$ compared to *SS* because of the uneven relaying load distribution amongst the nodes located at same hop count distance from the sink. Specifically, consider node n_i in Figure 5.2(a) and (b) although it is one hop neighboring node of the sink in both *SS* and *MS* but in the case of *MS* relaying load on node n_i is higher compared to *SS*. It is due to the fact that relaying load experienced by one hop neighboring nodes of the sink in *MS* is unevenly distributed due to non optimal position of the sink. As a result, node n_i is responsible for relaying data from much larger set of nodes in *MS* compared to *SS*. Furthermore, nodes located in the vicinity of node n_i also experience similar high relaying load in the case of the *MS*. As a result, probability of data congestion in the vicinity of node n_i due to channel contention will be high in the case of *MS* compared to *SS* which leads to increased average end-to-end data latency $\overline{\Delta}$ in the case of *MS* compared to *SS*.

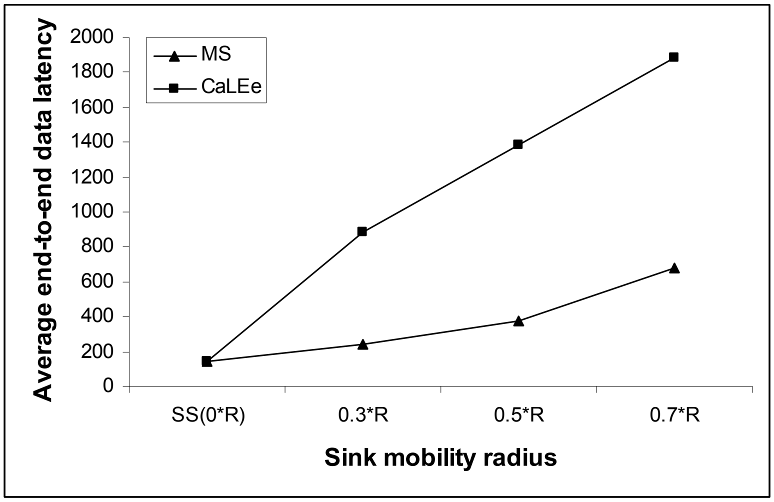
So, it can be concluded that closer the sink stays to the center of the WSN lesser will be average end-to-end data latency $\overline{\Delta}$ as well as average energy dissipation Ψ per node. For example, both $\overline{\Delta}$ and Ψ are minimum in the case of *SS* (that is mobility radius of the sink = 0) and increases with radius of the mobility trajectory of the sink.

In the following, we will study the effect of two primary parameters – radius of the mobility trajectory of the sink and number of sectors in a sensor field - on the performance ($\overline{\Delta}$ and Ψ) of our routing protocol *CaLEe* in comparison to *SS* and *MS*.

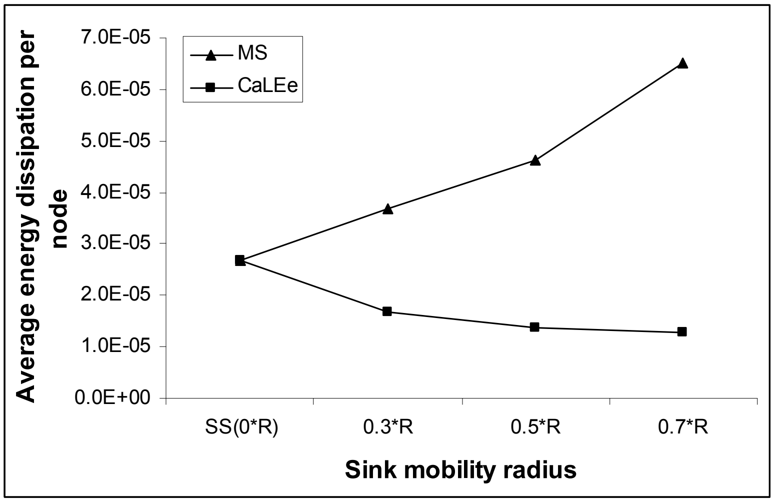
5.4.1 VARYING THE MOBILITY TRAJECTORY OF THE SINK

Variation in the radius of the mobility trajectory of the sink can greatly influence the performance ($\overline{\Delta}$ and Ψ) of the considered routing algorithms which we discuss in the following.

Figure 5.6 shows that when the sink is placed at the center of the WSN (*SS(0*R)*) minimum end-to-end data latency $\overline{\Delta}$ as well as minimum average end-to-end data latency was observed. Increase in the radius of the mobility trajectory of the sink results in increased latency $\overline{\Delta}$ and increased average energy dissipation Ψ in both *MS* and *CaLEe*. In the case of *MS* latency $\overline{\Delta}$ increases compared to *SS* by the factor of 1.38, 1.72 and 2.43 when the radius of the mobility by reducing the sink mobility is set to 0.3*R, 0.5*R and 0.7*R respectively. Similarly average energy dissipation Ψ increases by the factor of 1.73, 2.68 and 4.89 for 0.3*R, 0.5*R and 0.7*R respectively. By mobilizing the sink closer to the center of the sensor field (small values of radius of the mobility trajectory), we manage to evenly distribute the routing load amongst the nodes located at equal hop count distance from the sink. This leads to reduced data congestion and channel contention thus resulting in reduced end-to-end data latency $\overline{\Delta}$ (consider the discussion regarding node n_i of Figure 5.5 in above section). On the other hand, it is known from Section 5.2.1 that under data congestion and channel contention routing nodes can increase their active periods (>1%) to fulfill the data relaying requirements that causes increased average energy dissipation Ψ of the nodes as shown in Figure 5.6(b). However, we have seen in Figure 5.3 that $E_{\max}^{tx/rx}$ increases with decrease in the radius of the mobility trajectory of the sink (*SS(0*R)* has greater $E_{\max}^{tx/rx}$ compared to *MS(0.7*R)*). Because, at small radius of the mobility trajectory of the sink nodes positioned close to the center of the WSN experience high routing load that results in increased energy dissipation from these nodes and hence $E_{\max}^{tx/rx}$ increases. Thus, it can be inferred that although decrease in the radius of the mobility trajectory of the sink causes reduced average energy dissipation Ψ but it also results in decreased lifetime Φ^{WSN} of the WSN.



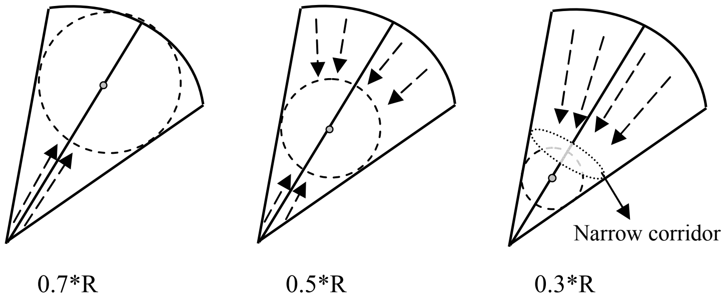
(a): Average end-to-end data latency $\overline{\Delta}$



(b): Average energy dissipation Ψ per node

Figure 5.6 Effect of sink mobility radius on Ψ and $\overline{\Delta}$

Figure 5.6 also shows that by increase in the radius of the mobility trajectory of the sink from 0*R to 0.3*R, 0.5*R and 0.7*R results in increased latency $\bar{\Delta}$ for the *CaLEe* compared to *SS* by a factor of 6.34, 9.96 and 13.57 respectively. In the case of *CaLEe* major cause of delay is the huge time that the sink takes to complete its trip along the mobility trajectory (as, length of the mobility trajectory equals $2\pi R$) for data collection from the collector nodes. With an increase in the radius of the mobility trajectory length of the mobility trajectory of the sink increases thus causing increased latency $\bar{\Delta}$. On the other hand, in contrast to *MS* average energy dissipation Ψ per node reduces in the case of *CaLEe* compared to *SS(0*R)* by a factor of 0.62, 0.51 and 0.48 when the radius of the mobility trajectory is increased to 0.3*R, 0.5*R and 0.7*R. Why?



0.7*R 0.5*R 0.3*R

Figure 5.7 Sector of a WSN with various positions of a data collector node

In the case of *CaLEe* the data collector node has to be located at the mobility trajectory of the sink, and thus change in the mobility trajectory of a sink causes a change in the position of the data collector node. Figure 5.7 shows a sector along with data collector nodes at positions 0.7*R, 0.5*R and 0.3*R from the center of a WSN. When the data collector node is positioned at 0.7*R then there is a small difference in the routing load experienced by the nodes located at equal hop count distance from it (shown by the circle in Figure 5.7) compared to the cases of 0.5*R and 0.3*R. This phenomenon can be easily seen in the case of 0.3*R where most of the data have to pass through a narrow corridor to reach the data collector node as shown in Figure 5.7. Relay nodes located at this narrow corridor have to prolong their active periods to fulfill data relaying requirement that causes increased energy dissipation from these nodes compared to rest of the sensor field. Hence, resulting in increased Ψ and $E_{\max}^{tx/rx}$ (reduced lifetime of the WSN).

5.4.2 VARYING THE NUMBER OF SECTORS

Another primary parameter that can have a strong influence on the performance of *CaLEe* routing protocol is the *"number of sectors"*. In this section we analyze its impact on the performance of the *CaLEe* routing protocol in terms of average energy dissipation Ψ per node and average end-to-end data latency $\overline{\Delta}$.

Figure 5.8 plots the percentage reduction in average energy dissipation Ψ per node of *SS* when the sensor field of size *R* (=1000, 800 and 600 meters) is partitioned into 4, 8, 12 and 16 sectors. It can be seen from Figure 5.8 that irrespective of the size of a sensor field increase in the number of sectors leads to a strong reduction (more than 60% compared to the case of *SS*) in the average energy dissipation Ψ per node. These gains originate from the fact that increase in the number of data collector nodes leads to the reduction in average path length between a source node and a data collector node which in result reduces the overall data relaying overhead from the nodes. However, achieved gains in Ψ becomes almost constant as the number of sectors in a WSN approaches 16 because further increase in the number of sectors results in very little decrease in the average distance between the nodes and data collector node.

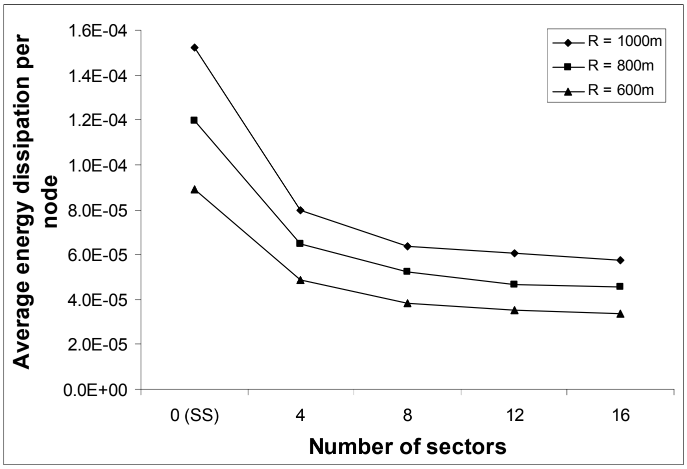


Figure 5.8 Number of sectors vs. Ψ for *CaLEe* routing protocol

On the other hand, it is known from Section 5.3.3.1 that total delay in *CaLEe* is the sum of the time required by the sink to complete a trip along its mobility trajectory and the time required to collect data from the data collector nodes. Since none of these two parameters is affected by, the change in the number of sectors therefore latency $\overline{\Delta}$ remains constant in *CaLEe*.

Thus, it can be concluded from above discussion that partitioning the WSN into small number of sectors (12 to 16) leads to substantial reduction in Ψ. On the other hand, the number of sectors in a WSN does not influence $\overline{\Delta}$. Does it mean, more the number of sectors higher the energy gains? The answer is no. As it can be seen from Figure 5.8 that once the number of sectors in a

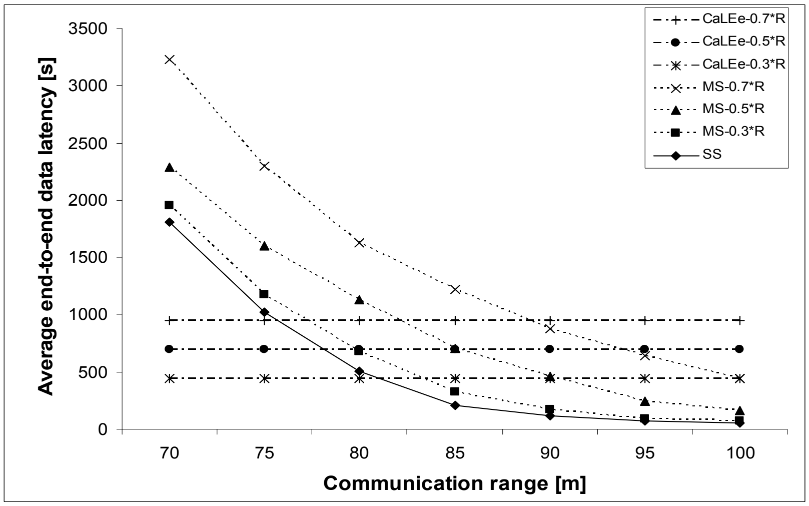
WSN approaches 16 the energy gains become constant and further increase in the number of sectors improves nothing. On the other hand, it is known from Chapter 4 that more the number of sectors higher will be the effort required to establish them causing increased energy dissipation from the nodes. Therefore, it is recommended that for a WSN of small to average size *(R=600m to R=1000m)* 12 to 16 sectors are ideal, however if the size of a WSN grows extremely large then increase in the number of sectors can be considered.

Now we will evaluate the performance of the *CaLEe* approach under different network configurations along with *SS* and *MS*. We vary the communication range of the nodes, throughput of the WSN, sensor node density and size of the sensor field to analyze their impact on average energy dissipation Ψ per node and on average end-to-end data latency $\overline{\Delta}$. Note, hereafter *A-x*R* in the figures denoted the results obtained by applying algorithm *A* when the mobility trajectory of the sink is set at a distance of *x*R* from the center of the WSN. For example, *CaLEe*-0.7*R denotes the use of the *CaLEe* routing scheme such that the radius of the sink mobility path is set to 0.7*R.

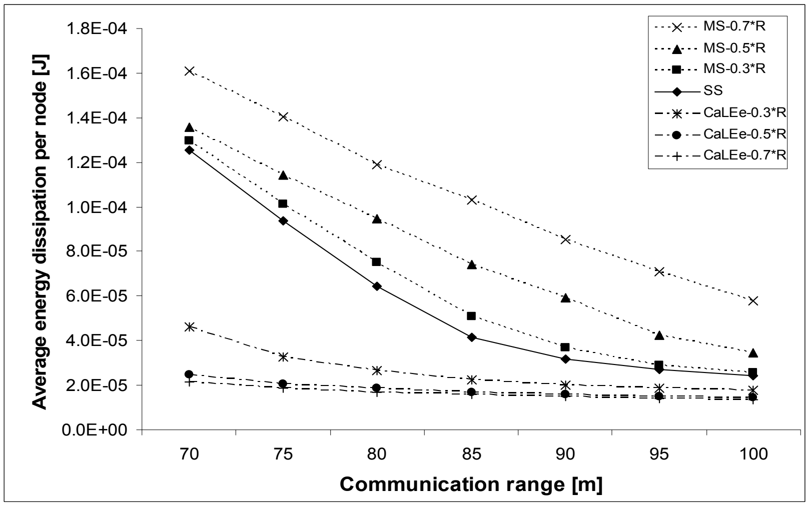
5.4.3 VARYING THE COMMUNICATION RANGE OF THE NODES

In this section, we analyze the effect of the communication range *r* of a node on the average delay incurred by a message from source to sink. It has already been discussed in Section 5.3.2 that due to identical transceiver system of low budget sensor nodes sink can only communicate with one node at a time as a result maximum network throughput cannot exceed 1. Therefore, for this particular simulation we set the throughput *G = 0.8 packets per time slot* and the communication range *r* of the nodes is varied from 70m to 100m (for micaz maximum *r* equals 100m).

End-to-end data latency $\overline{\Delta}$ in WSN is mainly caused by two factors: one is the channel access delay and second is the reduced duty cycling of the sensor nodes. Channel access delay is caused by shared wireless medium in a WSN, while nodes are often programmed to operate at reduced duty cycle to preserve energy. A message experiences these hurdles at each hop on its way from source to sink. Thus, one way to reduce the delay is to reduce the number of hops that the data has to take from source to sink. This can be achieved easily by increasing the communication range *r* of the sensor nodes. However, increase in the communication range of the sensor nodes results in increased channel contention and hence increased average end-to-end data latency $\overline{\Delta}$ of a WSN [141]. The intuition behind this is that increase in *r* results in increased neighbor node density per node and therefore channel access probability of the nodes reduces because of the shared communication medium resulting in increased $\overline{\Delta}$. Since in our model we assumed that the nodes are operating at 1% duty cycles therefore increase in the neighbor node density does not have much influence on the channel access probability of a node (as nodes spent most of their time in sleep state). On the other hand, by increasing *r* we reduce the number of hops from source to destination (sink/data collector node) and increase the probability of finding next hop neighboring node in active state thus substantial reduction in average end-to-end data latency $\overline{\Delta}$ is achieved in the cases of *SS* and *MS* as shown in Figure 5.9(a).



(a): Communication range of the nodes vs. average end-to-end data latency $\overline{\Delta}$



(b): Communication range of the nodes vs. average energy dissipation Ψ per node

Figure 5.9 Varying communication range of the sensor nodes

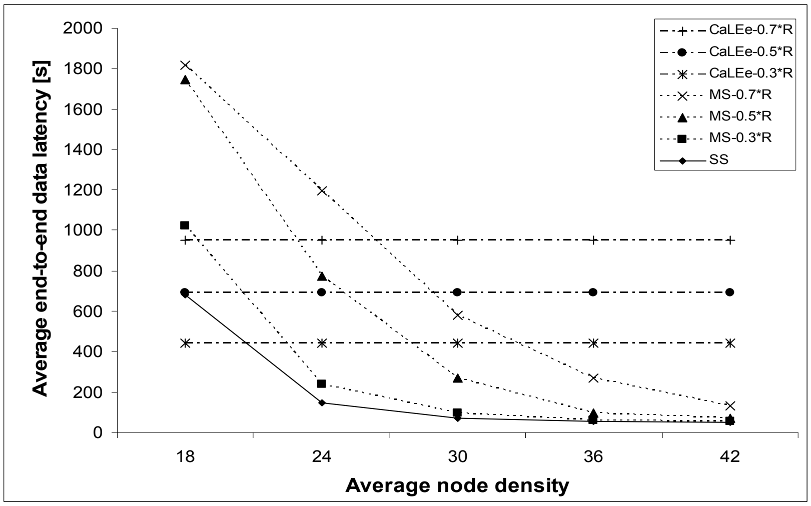
In the case of the *CaLEe* routing protocol end-to-end data latency $\overline{\Delta}$ is caused due to slow mobility speed of the sink and the time required by the sink to collect data from the data collector nodes. Under constant throughput of the WSN time required to collect data from the data collector nodes is constant also, time required by the sink to complete a trip along the mobility trajectory is also constant under fixed size of the WSN (refer to Section 5.3.3.1) therefore straight lines are plotted for *CaLEe* in Figure 5.9.

Note that during the analysis for average energy dissipation Ψ per node we assumed that irrespective of the communication range *r* of a node the energy required to transmit a data packet is the same. With this assumption we can compare Ψ in *SS, MS* and *CaLEe* that is the main goal of these simulations but can not comment on what can be the optimal value of *r* under different protocols and scenarios. Figure 5.9 (b) shows that for all considered communication ranges *CaLEe* has the least average energy dissipation Ψ per node. These gains were possible due to partitioning of the sensor field into sectors and establishment of a data collector node in each sector that results in reduced average path length from source to destination (data collector node) compared to both *SS* and *MS*.

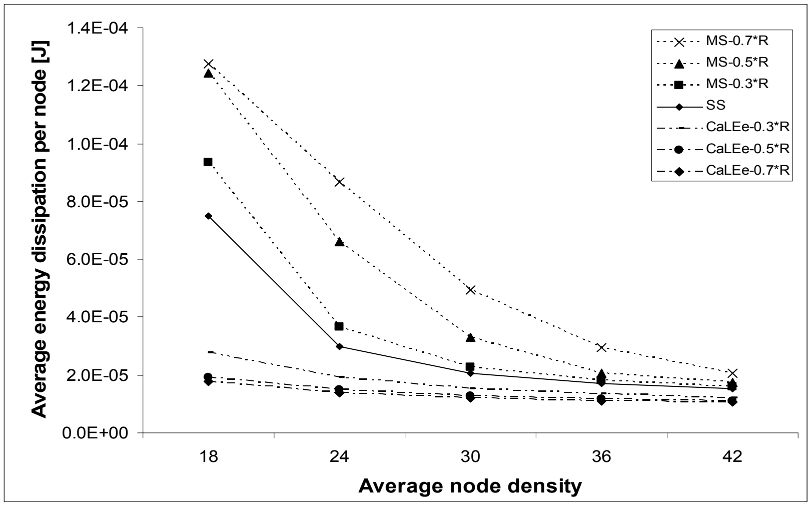
5.4.4 VARYING THE SENSOR NODE DENSITY

In this section we will analyze the effects of average sensor node density Γ on average end-to-end data latency $\overline{\Delta}$ and average energy dissipation Ψ per node. Figure 5.10 plots $\overline{\Delta}$ and Ψ for *CaLEe*, *SS* and *MS* under different sensor node densities. For these simulations we set the communication range of the node to 80 meters (for micaz maximum *r* equals 100m) and the throughput to 0.72 packets per time slot (G is bounded by (0, 1]).

In contrast to general assumptions that increase in the sensor node density leads to increased channel contention and hence increased end-to-end data latency Figure 5.10 shows the opposite. Intuition behind this assumption is that increase in the sensor node density results in an increased number of nodes contending for channel causing channel contention and hence increase in the latency. However, in considered network setting reduced duty cycling (1%) of the nodes leads to long sleep intervals for the nodes that minimize the probability of channel contention. On the other hand, if a node wants to transmit a data packet then due to unsynchronized and low duty cycling of the nodes the probability of finding the next hop in active state is also very low. If a node cannot find any of its next hop neighboring node ready to receive its data packets (because they are in sleep mode), then it goes back to sleep mode and wakes up again after a pre-configured time interval and retries to transmit. Therefore, it can be concluded that in the current setup delay is mainly caused by the long sleep intervals of the nodes and not due to channel contention. Now by increasing the sensor node density we increase the number of next hop neighbors of a node and hence the probability that even with low duty cycling at least one next hop will be available for data collection when a node wants to transmit its data. Thus, increase in the sensor node density helps to reduce $\overline{\Delta}$ by reducing the time that the data spends in the buffer of relaying nodes (waiting time) that can be seen from Figure 5.10.



(a): Sensor node density vs. average end-to-end latency $\overline{\Delta}$



(b): Sensor node density vs. average energy dissipation Ψ per node

Figure 5.10 Effect of sensor node density on performance of a WSN

For the case of the *CaLEe* routing protocol end-to-end data latency $\overline{\Delta}$ is caused due to slow mobility speed of the sink and the time required by the sink to collect data from the data collector nodes. Under constant throughput of the WSN time required to collect data from the data collector nodes is constant also, time required by the sink to complete a trip along the mobility trajectory is also constant under fixed size of the WSN (refer to Section 5.3.3.1) therefore straight lines are plotted for *CaLEe* in Figure 5.10.

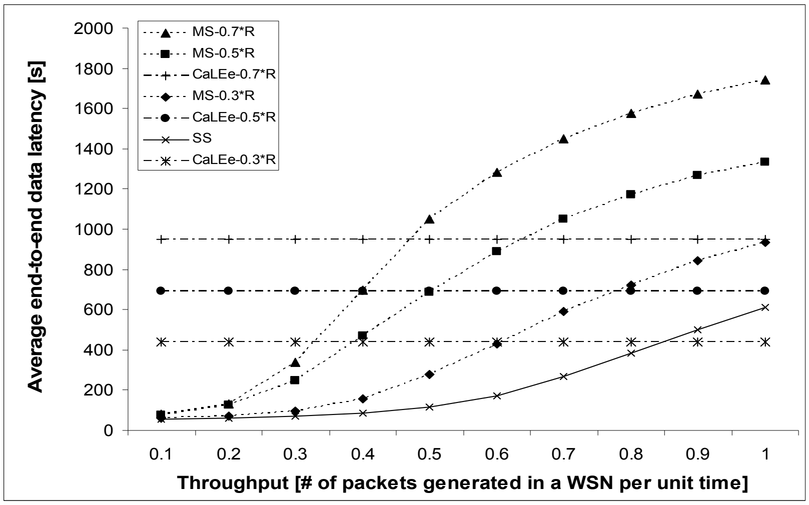
It can be seen from Figure 5.10(b) that *CaLEe* again outperforms *SS* and *MS* in terms of average energy dissipation Ψ per node. Figure 5.10 (b) shows that for small node density the average energy dissipation per node in both *SS* and *MS* is very high. With an increase in the node density we observe a steep fall in the energy curve. This phenomenon can be easily understood by considering Figure 5.1 that presents temporal states of a sensor node. Due to low sensor node density and reduced duty cycling a node has to wait for a long period before it manages to make a connection with one of its next hop neighboring node. During this time, the node itself has generated a handful of data packets and may have received relay data packets from neighboring nodes. After making the connection, it is often not possible for this node to relay all data packets in its scheduled awake period. Therefore, in addition to scheduled awake period (A_R) it extends the awake period (A_N, refer to Figure 5.1) to forward all of its data leading to an increased energy dissipation. With an increase in the sensor node density the probability of finding available next hop neighboring node increases, thus minimizing the requirement for extended awake period of the nodes and hence leading to reduced average energy dissipation Ψ per node.

5.4.5 VARYING THE THROUGHPUT OF THE SENSOR FIELD

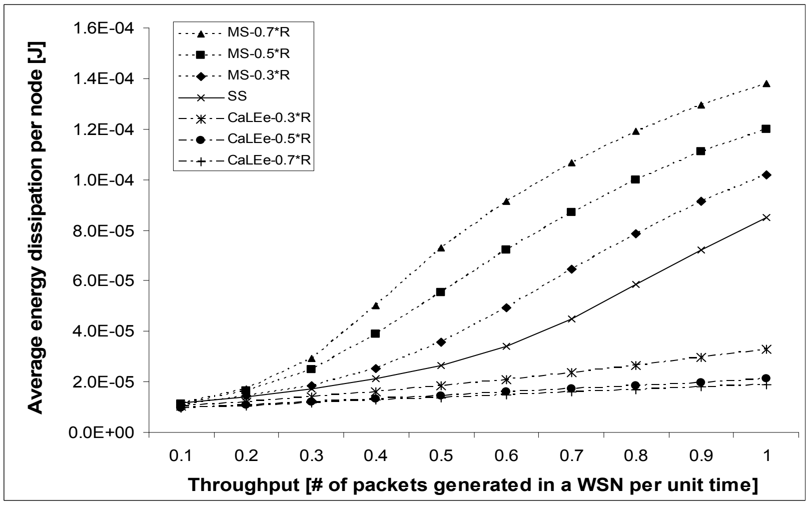
In this section, we analyze how the network throughput influences average end-to-end data latency $\overline{\Delta}$ and average energy dissipation Ψ per node. For these simulations, we set the communication range of the sensor nodes to 80m (for micaz maximum *r* equals 100m) and average sensor node density is fixed at 20 nodes (in order to ensure connected coverage).

Figure 5.11(a) presents average end-to-end data latency $\overline{\Delta}$ experienced by a node when the throughput of the sensor field is varied from 0.1 to 1.0. Under constant sensor node density increase in the network throughput is achieved by increasing the data generation rate of the nodes that results in increased relaying load per node, channel contention and hence increased data latency $\overline{\Delta}$ in the cases of *SS* and *MS*. Behavior of the *SS* and *MS* curves in Figure 5.11(a) can be understood as follows. When the throughput of a WSN is small then there is no data queuing at the nodes and they can relay the data packets as the data arrive. However, with an increase in the throughput queues start to build at the relay nodes, (we have assumed unlimited buffer capacity at the nodes) and a steep rise in the delay curve is observed due to channel contention.

When the throughput gets very high (approaches 1.0) then every relay node maintains a data queue because it cannot forward the sensed/received data due to high channel contention resulting in very high delay (compared to the case of low throughput). It is known from our simulation model that when a node acquires a channel it retains the channel until it has forwarded all the data to one of its next hop neighbor. Therefore, at high throughput (close to 1.0) data will experience constant but high delay, which can be seen from Figure 5.11(a).



(a): Sensor field throughput vs. average end-to-end latency $\overline{\Delta}$



(b): Sensor field throughput vs. average energy dissipation Ψ per node

Figure 5.11 Effect of throughput on performance of a WSN

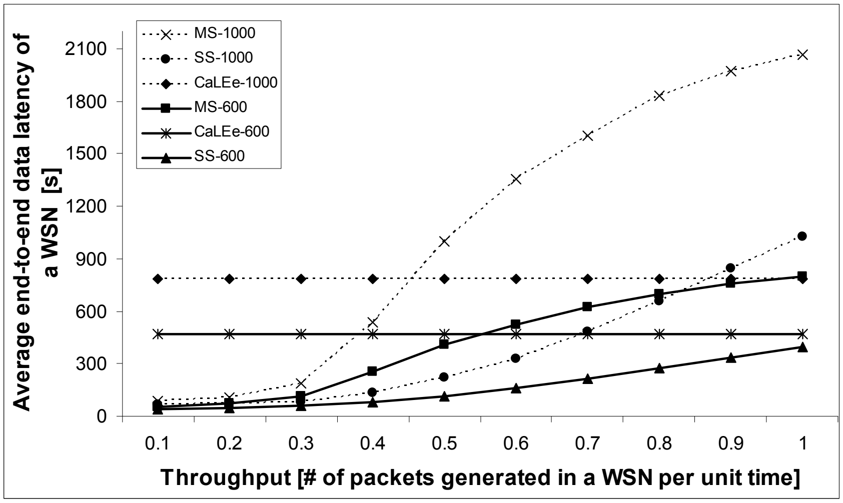
On the other hand, in the case of the *CaLEe* routing protocol increase in the throughput of a WSN increases the sojourn time of the sink at each data collector node. However, it is known from Section 5.3.3.1 that slight change in the sum of the sojourn time (128[s]) of the sink at all data collector nodes is not significant compared to average end-to-end data latency caused by slow mobility speed of the sink (952[s]). Therefore, in Figure 5.11(a) we plotted an upper bound of average end-to-end data latency $\overline{\Delta}$ for *CaLEe* (considering maximum sojourn time) irrespective of the throughput of the WSN.

Figure 5.11(b) shows average energy dissipation Ψ per node when the throughput of the sensor field varies from 0.1 to 1.0. It can be observed from Figure 5.11(b) that *CaLEe* has least average energy dissipation per node compared to both *SS* and *MS* because of the shortest average path length from source to the destination (data collector node). With an increase in the throughput of a WSN data relaying load increases at the nodes that leads to an increase in the average energy dissipation per node for *CaLEe*, *SS* and *MS*. Also larger the queue grow at a relay node higher will be time that this node has to spent in extended active mode to forward all the data in its buffer thus resulting in increased average energy dissipation per node. This phenomenon of *SS* and *MS* can be seen from Figure 5.11(b).

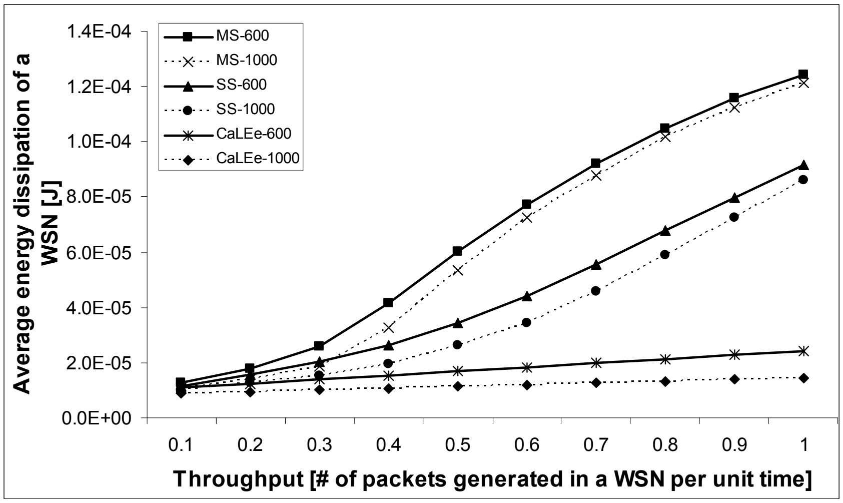
5.4.6 VARYING THE SIZE OF A SENSOR FIELD

In this section, we discuss how change in the size of a sensor field affects the performance of the *CaLEe* routing protocol compared to *SS* and *MS*. In order to achieve the goal we set up two medium sized sensor fields of radius $R = 600m$ and $R = 1000m$ in our simulator. Sensor node density=20nodes (required to ensure connected coverage of the field) and communication range r = 80m (for micaz maximum r equals 100m) and mobility trajectory of the sink is set to 0.5*R (average path). We then repeated the simulations discussed in Section 5.4.5 for the two fields.

It can be seen from Figure 5.12 that increase in the size of a WSN leads to an increase in average end-to-end data latency and average energy dissipation per node for all three considered protocols. In the cases of *SS* and *MS,* increase in the size of the sensor field leads to an increase in average path length between source node and sink. Furthermore, it is known from Section 5.4.3 that even under constant throughput of a sensor field increase in the number of hop counts between source and destination leads to an increase in average end-to-end data latency as well as average energy dissipation per node. These two arguments justify the increase in $\overline{\Delta}$ and Ψ for both *SS* and *MS*. However, amplitude of this increase depends on three factors that are communication range r of the node, sensor node density Γ, throughput G of the sensor field. Based on our discussion in Section 5.4.3, 5.4.4 and 5.4.5 it can be stated that $\overline{\Delta}$ and Ψ are inversely proportional to r, Γ and directly proportional to G. Thus, if sensor field is configured with low values of r, Γ and high values of G then increase in the size of the sensor field results in large increase in both $\overline{\Delta}$ and Ψ and vice versa.



(a): Size of a WSN vs. average end-to-end data latency $\overline{\Delta}$



(b): Size of a WSN vs. average energy dissipation Ψ per node

Figure 5.12 Analyzing the effect of size of a WSN on the performance of SS, MS and CaLEe

(Note: Bold line represents the sensor field with $R=600m$ and dotted line is for $R=1000m$.)

In the case of *CaLEe*, under constant throughput of the sensor field increase in the delay is directly proportional to an increase in the length of the mobility trajectory of the sink (as sojourn time of the sink remains constant due to fixed *G*). So, with an increase in the size of a WSN delay increases linearly for *CaLEe*. On the other hand, similar to *SS* and *MS* increase in the size of a WSN leads to an increase in the hop count between source node and data collector node that leads to increased energy dissipation from the sensor field. It is also known from Section 5.4.3, 5.4.4 and 5.4.5 that similar to *SS* and *MS* parameters *r,* Γ and *G* also influence $\overline{\Delta}$ and Ψ in *CaLEe* routing scheme.

In order to analyze the impact of increase in the size of the WSN on *SS, MS* and *CaLEe* in a real world scenario we provide some numbers in Figure 5.12. It shows that for *R* = 600m and the throughput of the sensor field is 0.9 then compared to *CaLEe SS* reduces the average end-to-end data latency $\overline{\Delta}$ by the factor of 1.41and *MS* increases $\overline{\Delta}$ by the factor of 1.61. On the other hand, *SS* and *MS* result in increased average energy dissipation Ψ per node compared to *CaLEe* by the factor of 3.5 and 5.0 respectively. In addition, when the size of a WSN is increased *(R=1000m)* then compared to *CaLEe* both *SS* and *MS* results in increased $\overline{\Delta}$ by the factor of 1.07 and 4.18 respectively. Similarly, *SS* and *MS* result in increased average energy dissipation Ψ per node compared to *CaLEe* by the factor of 4.2 and 7.0 respectively. Thus, we conclude that irrespective of the network configuration (*r,* Γ and G) increase in the size of the sensor field results in performance gains of *CaLEe* over *SS* and *MS* in terms of both average end-to-end data latency $\overline{\Delta}$ and average energy dissipation Ψ per node.

5.5 SUMMARY

In this chapter, we analyzed the performance of two major classes of routing protocols *SS* and *MS* in addition to a newly developed routing protocol *CaLEe* by varying six network parameters *r*, Γ, *G*, size of the WSN, number of sectors in a WSN and radius of the mobility trajectory of the sink. We obtain following results:

- Number of sectors have high influence on the performance of *CaLEe* in terms of Ψ. For small and medium sized sensor fields 16-20 sectors are sufficient to minimize average energy dissipation per node. However, for large size of the WSN increase in the number of sectors is required. On the other hand, considering low number of sectors (16-20) $\overline{\Delta}$ is independent of number of sectors. *SS* and *MS* do not partition the field.
- Radius of the mobility trajectory of the sink strongly influence both $\overline{\Delta}$ and Ψ in *MS* and *CaLEe*. We observe that smaller the radius of the mobility trajectory lower will be $\overline{\Delta}$ and Ψ, and vice versa. When radius of the mobility trajectory is set to zero in then it becomes the case of *SS* that has minimum $\overline{\Delta}$ and Ψ. However, Φ^{WSN} is minimum in the case of *SS* (nodes located in the vicinity of the sink have high energy depletion rate), followed by *MS* (sink mobility helps to balance energy dissipation amongst the nodes). While *CaLEe* has

maximum Φ^{WSN} because multiple data collector nodes helps to minimize the average path length from source to data collector node that result in reduced energy dissipation. Note that reduction in the average energy dissipation Ψ per node does not imply increase in the lifetime of a WSN, as maximum lifetime of a WSN is achieved only when the mobility trajectory of the sink is set to $\sqrt{2}R/2 \cong 0.7 * R$ [140] (refer to Section 5.4).

- *r,* Γ and *G* also have strong influence on $\overline{\Delta}$ and Ψ of the three routing protocols. Small values of *r* and Γ results in high $\overline{\Delta}$ and Ψ in the cases of *SS* and *MS* compared to *CaLEe* and vice versa. On the other hand, under low *G* both *SS* and *MS* outperform *CaLEe* in terms of $\overline{\Delta}$ and Ψ and vice versa.
- Considering fixed values for *r,* Γ, *G, number of* sectors and *radius of the mobility trajectory of the sink* we observe that increase in the size of the WSN improves the performance of the *CaLEe* compared to *SS* and *MS* both in terms of $\overline{\Delta}$ and Ψ.

One thing is clear from the discussion that there is no such thing as *"one for all routing protocol"* meaning each of the mentioned protocols has its pros and cons under different network settings. Thus, we need to identify operational regions based on varying network configurations where one protocol *(SS, MS and CaLEe)* performs better than the other.

CHAPTER 6: THE SOLUTION: AN ADAPTIVE ROUTING PROTOCOL

6.1 RESEARCH CONTRIBUTION

Discussion in Chapter 5 showed there is no *"One fits all"* type of routing protocol for WSNs that can provide best-case solution under varying network scenarios. Furthermore, due to error prone nature of the sensor nodes network configuration may change during the operation of the network hence, resulting in variable performance (energy efficiency, data latency) from same routing protocol. Based on this finding we will address following two questions in this chapter,

Q.1 Given some initial network configuration and user requirements, how one can decide which protocol (*SS, MS or CaLEe*) to use?

Q2. During the operation of the WSN, changing user requirements $(\overline{\Delta}, \Psi)$ and/or changing network configuration can effect the performance of the network. In such dynamic environment, how one can sustain the performance of a WSN at desired level.

In order to answer Q1 we will analyze (in Section 6.2) how change in various network parameters affects the performance of three considered routing schemes *SS, MS* and *CaLEe*. During the simulations we analyze the impact of six dimensional input space (communication range *r* of the nodes, node density Γ, throughput *G* of the WSN, size of a WSN, number of sectors in a WSN and radius of the mobility trajectory of the sink), which defines a network configuration on the performance (energy efficiency, data latency) of the routing protocol. By doing this we identify the operational regions (network configurations) were one routing scheme performs better than the other thus leading to optimal performance of the WSN.

We provide an answer to Q2 based on the concept of adaptation (in Section 6.3). We propose a road map for the development of a meta-routing protocol that depending on changing network configuration and user requirements switches between sub-routing schemes *(SS, MS and CaLEe)* to sustain the performance of the network at desired level.

6.2 WHEN TO USE *SS, MS* AND *CALEE*

Our discussion in Chapter 5 showed that considered network configuration has high influence on the performance of the routing protocols. Furthermore, our considered network configuration can be defined using six parameters namely, communication range *r* of the nodes, throughput *G* of the WSN, node density Γ, size of the WSN, radius of the mobility trajectory of the sink and number of sectors. These parameters can be divided into static and dynamic parameters. Those parameters, which are fixed at the deployment of the WSN, are called static parameters. If the value of a parameter changes during the lifetime of a WSN (course of the simulation), it is called dynamic parameter.

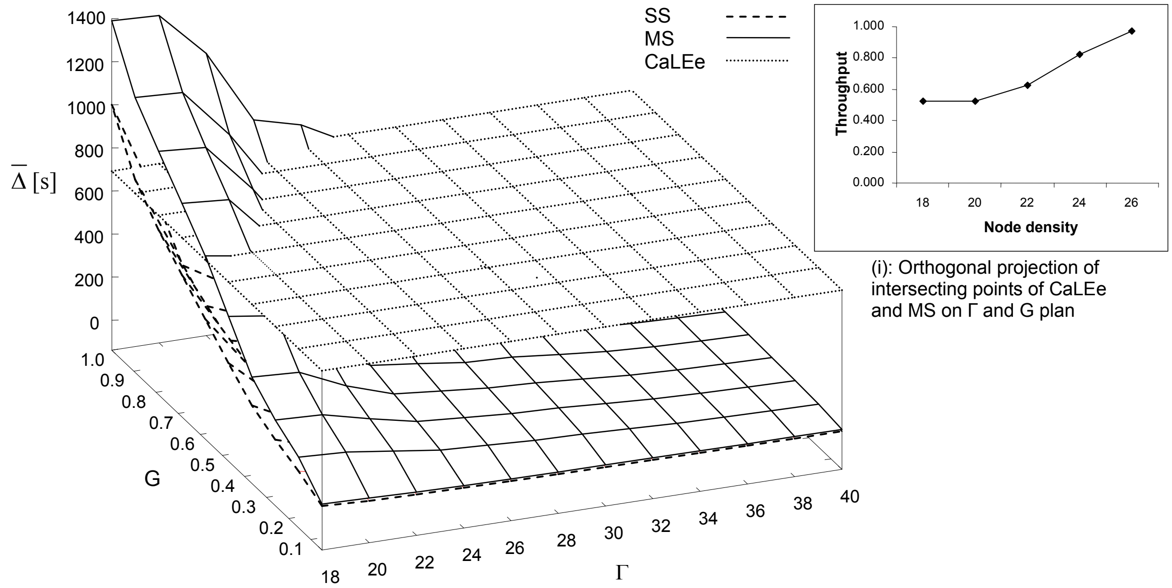
Simulation results in Section 5.4.6 have shown that larger the size of a WSN higher will be the performance gains of *CaLEe* compared to *SS* and *MS*. However, for a given application size WSN is usually known before the deployment of the WSN and it remains fixed during the operation of the WSN. Utilizing the information about the size of a WSN one can calculate the optimal number of sectors for *CaLEe* routing scheme. For example, it has been observed that for a WSN having R=800 meters energy gains saturate as the number of sectors approaches 12 (refer to Section 5.4.2 for details). Thus, for a given sensor field, size of a WSN and number of sectors are two static variables. On the other hand, communication range *r* of a node, throughput *G* of the WSN, sensor node density Γ and radius of the mobility trajectory of the sink are the parameters which can be configured to different values during the course of the simulation and thus are called dynamic variables. By performing this categorization of input variables, we manage to reduce our inputs from six to four-dimensional space.

In Chapter 5, we analyzed the effect of change in the network parameter individually (while keeping rest of the network parameters fixed) on the performance of the WSN. In this chapter we go one step further and study them in pairs (*r, G*), (*r,* Γ) and (*G,* Γ). In these simulations, we fix the size of a WSN and hence the number of sectors for *CaLEe* routing protocol. Also for each simulation, we vary two variables simultaneously. Furthermore, in Section 6.2.4 we study the impact of three input parameters (*radius of the mobility trajectory of the sink,* Γ *and G*) simultaneously on the performance of *SS*, *MS* and *CaLEe*. These simulations help us to better understand the impact of six dimensional input space on the performance of considered routing protocols. In addition, based on the obtained results we can identify the network configurations where one protocol performs better than the others in terms of latency $\overline{\Delta}$ and energy efficiency Ψ. Basic network setup for these simulations is the same, which we considered in Chapter 5.

6.2.1 THROUGHPUT OF THE WSN AND SENSOR NODE DENSITY

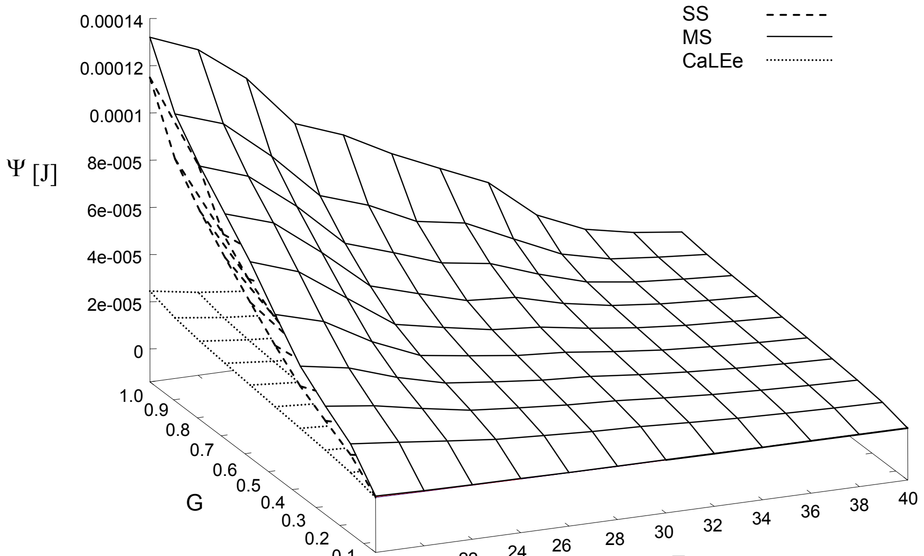
In this section, we analyze how change in the throughput *G* of the sensor field and sensor node density Γ affects average end-to-end data latency $\overline{\Delta}$ of WSN and average energy dissipation Ψ

per node of *SS, MS* and *CaLEe*. For the simulation, we varied the node density from 18 to 40 nodes (a standard range of densities assumed during ad-hoc deployment of the nodes to ensure connected coverage) and the throughput of the WSN is varied from 0.05 to 1.0 packets per time slot (since G is bounded by (0, 1]), refer to our discussion in Chapter 5). In Figure 6.1(a) and (b) three-dimensional graphs show the data latency $\overline{\Delta}$ of WSN and average energy dissipation Ψ per node as a function of both throughput G of the WSN and node density Γ.



(i): Orthogonal projection of intersecting points of CaLEe and MS on Γ and G plan

(a): $\overline{\Delta}$ expressed as a function of G and Γ



(b): Ψ expressed as a function of G and Γ

Figure 6.1 Analyzing the impact of G and Γ on the performance of the routing protocols

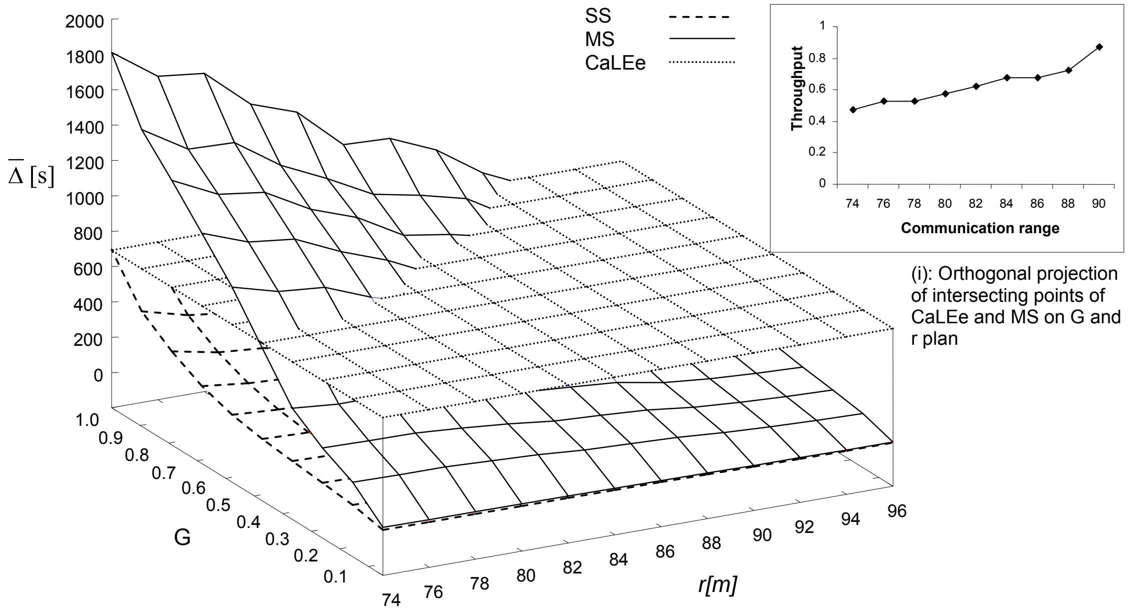
Figure 6.1(a) shows that data latency $\overline{\Delta}$ remain constant for *CaLEe* because latency in this case is independent of *G* and Γ but depends on the mobility speed of the sink (see Section 5.3.3.1). On the other hand, for *SS* and *MS* it is known from Section 5.4.4 and 5.4.5 that small density and high throughput result in high data latency $\overline{\Delta}$. Figure 6.1(a) also validates these findings and show that for low values of node density increase in the throughput of the WSN causes a step rise in data latency $\overline{\Delta}$. For example, with the node density fixed at 18 nodes, end-to-end data latency in *MS* and *SS* becomes more than *CaLEe* as the throughput is increased beyond 0.55 and 0.85 packets per time slot respectively. However, with increase in the sensor node density average end-to-end data latency $\overline{\Delta}$ starts to reduce in *SS* and *MS* and becomes lower than *CaLEe* for all values of the *G* when the node density approaches 20 and 28 nodes respectively. Furthermore, Figure 6.1(a)-(i) which plots the orthogonal projection of intersection point of *CaLEe* and *MS* on *G* and Γ plan highlights how change in the node density and throughput of the WSN actually causes a change in the intersection points of the two curves (*MS* and *CaLEe*).

On the other hand, similar to our findings in Section 5.4.4 and 5.4.5 average energy dissipation Ψ per node remains minimum in the case of *CaLEe* (shortest average path length from sensor to the sink) followed by *SS* and then the *MS* for all considered cases as shown in Figure 6.1(b).

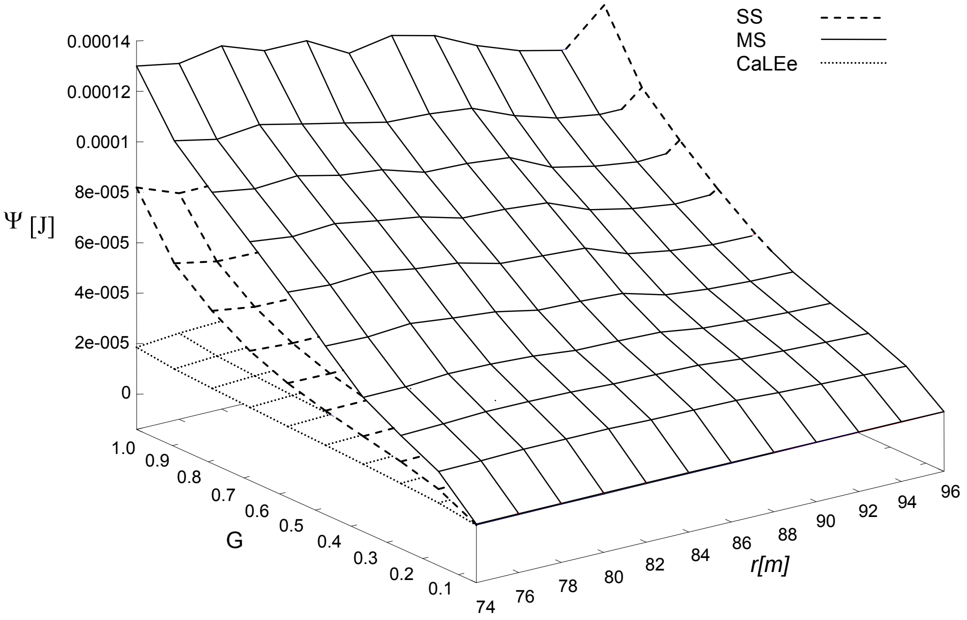
So, it can be inferred from above discussion that *CaLEe* has minimum average energy dissipation Ψ per node for all considered pairs of (G, Γ). *CaLEe* also provides minimum data latency $\overline{\Delta}$ compared to both *SS* and *MS* for G = 1.0 packets per time slot and Γ = 18 nodes thus providing optimal performance both in terms of energy dissipation Ψ and data latency $\overline{\Delta}$ amongst the three considered protocols. Moreover, having G fixed at 1.0 packet per time slot decrease in sensor node density Γ (below 18 nodes) will result in further degraded performance of both *SS* and *MS* compared to *CaLEe*. Also, use of *MS* is not advised for those pairs of (G, Γ) for which *MS* has higher delay than *CaLEe* (see Figure 6.1(a)) because in these cases *MS* gives worst results for both energy dissipation Ψ and data latency $\overline{\Delta}$ compared to *SS* and *CaLEe*. Thus, in these cases either *SS* or *CaLEe* must be used depending on the user requirements. To conclude for all pairs of (G, Γ) *CaLEe* can be used if minimum average energy dissipation per node is desired and *SS* can be used if minimum average end-to-end data latency $\overline{\Delta}$ is required.

6.2.2 THROUGHPUT OF THE WSN AND COMMUNICATION RANGE OF THE NODES

In this section, we analyze how change in the throughput *G* of the sensor field and communication range *r* of the nodes affects average end-to-end data latency $\overline{\Delta}$ of WSN and average energy dissipation Ψ per node of *SS, MS* and *CaLEe*. For the simulation, we varied the communication range of the nodes from 74m to 96m (maximum outdoor communication range of micaz nodes is 100m) and the throughput of the WSN is varied from 0.05 to 1.0 packets per time slot (since G is bounded by $(0,1]$, refer to our discussion in Chapter 5). In Figure 6.2(a) and (b) three-dimensional graphs shows the average end-to-end data latency $\overline{\Delta}$ of WSN and average energy dissipation Ψ per node as a function of both *G* and *r*.



(a): $\overline{\Delta}$ expressed as a function of *G* and *r*



(b): Ψ expressed as a function of *G* and *r*

Figure 6.2 Analyzing the impact of *G* and *r* on the performance of the routing protocols

Figure 6.2(a) shows that data latency $\overline{\Delta}$ remain constant for *CaLEe* because latency in this case is independent of *G* and *r* but depend on the mobility speed of the sink (see Section 5.3.3.1). On the other hand, for *SS* and *MS* it is known from Section 5.4.3 and 5.4.5 that small communication range and high throughput result in high data latency $\overline{\Delta}$. These results are revalidated in Figure 6.2(a) where *SS* and *MS* show high data latency $\overline{\Delta}$ under small communication range and high throughput of the WSN. For example, in the considered scenarios if *r* is set to 74m then end-to-end data latency in *MS* becomes more than *CaLEe* when the throughput of the WSN is increased beyond 0.5 packets per time slot. Similarly, if we fix the throughput of the WSN at 1.0 then data latency $\overline{\Delta}$ in *MS* remains higher than *CaLEe* until the communication range of the nodes is increased beyond 90m. Furthermore, Figure 6.2 (a)-(i) which plots the orthogonal projection of intersection point of *CaLEe* and *MS* on *G* and *r* plan highlights how change in the throughput and communication range of the nodes actually causes a change in the intersection points of the two curves (*MS* and *CaLEe*). Figure 6.2(a) also shows that *SS* has least average end-to-end data latency $\overline{\Delta}$ compared to *MS* and *CaLEe* for all the considered cases. So, it can be inferred from above discussion that under small communication range of the nodes and high throughput of the WSN *CaLEe* performs better than *MS* in terms of data latency $\overline{\Delta}$*,* though, *SS* has minimum data latency $\overline{\Delta}$ for all possible pairs of *(G, r)*.

Average energy dissipation Ψ per node remains minimum in the case of *CaLEe* (because it has the shortest average path length from sensor to the sink) followed by *SS* and then the *MS* as shown in Figure 6.2(b). However, when communication range of the nodes becomes high (*r* approaches 96m and G is greater than 0.5 packets per time slot) then the average energy dissipation Ψ per node in *SS* grows greater than *MS* as shown in Figure 6.2(b). Increase in Ψ is caused by prolonged active periods of the nodes which result from increased channel contention caused by increase in *r* (refer to Section 5.4.3) and throughput of the WSN (refer to Section 5.4.5).

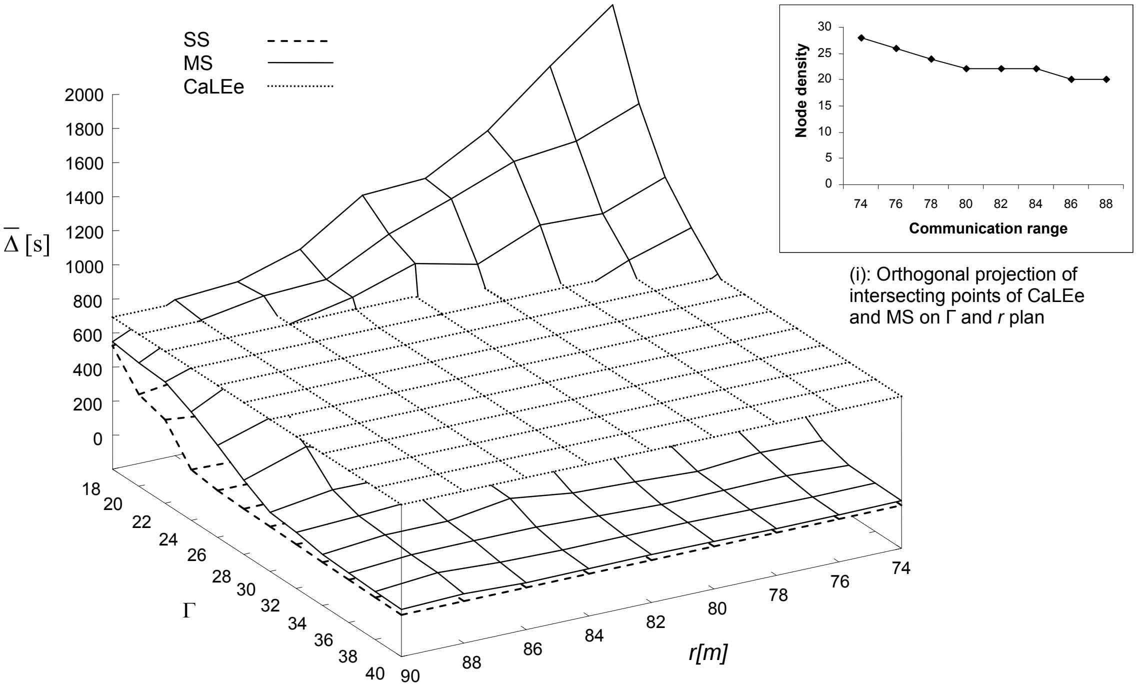
Thus, if minimum data latency $\overline{\Delta}$ is desired and energy dissipation Ψ is not the parameters of interest then *SS* should be adopted. If minium energy dissipation Ψ is required and data latency $\overline{\Delta}$ is not the parameter of interest then *CaLEe* should be adopted. In the considered network setup optimal performance (minimum energy dissipation Ψ per node and minimum end-to-end data latency $\overline{\Delta}$) can be achieved using *CaLEe* when the network is configured to operate with *G=1.0* packets per time slot and *r=74m* as shown in Figure 6.2(a)/(b).

6.2.3 SENSOR NODE DENSITY AND COMMUNICATION RANGE OF THE NODES

Now we analyze how change in the sensor node density Γ and communication range *r* of the nodes affects average end-to-end data latency $\overline{\Delta}$ and average energy dissipation Ψ per node of *SS, MS* and *CaLEe*. For the simulation, we varied the sensor node density Γ from 18 to 40 nodes (that is the standard range of densities assumed during ad-hoc deployment of the nodes to ensure connected coverage) and communication range of the nodes from 74 to 90m (maximum outdoor communication range of micaz node 100m). In Figure 6.3(a) and (b) three-dimensional graph show the latency $\overline{\Delta}$ and energy dissipation Ψ as a function of both Γ and *r*.

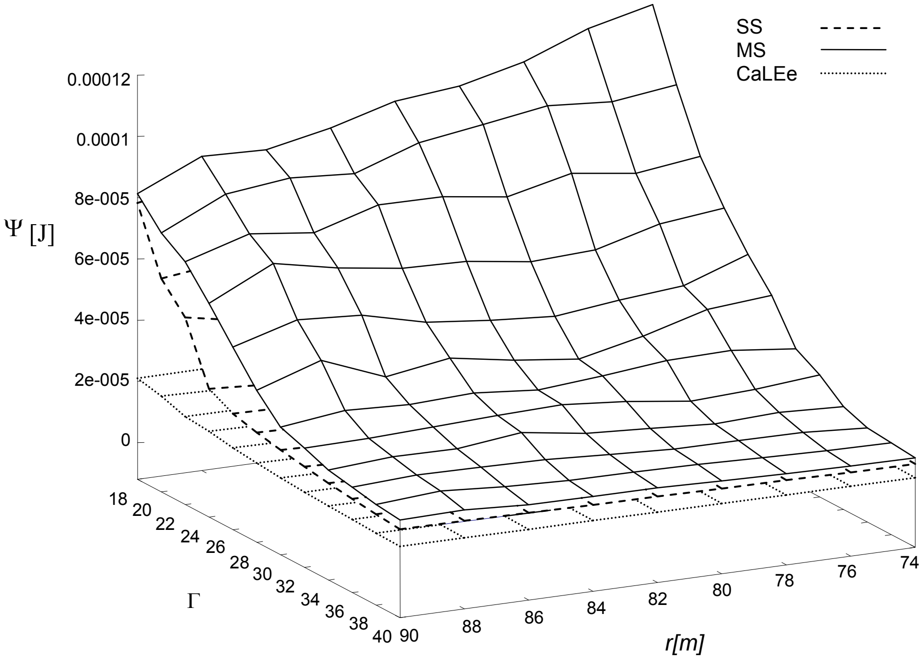
Average end-to-end data latency $\overline{\Delta}$ remain constant in *CaLEe* because average end-to-end data latency in this case is independent of *r* and Γ but depends on the mobility speed of the sink (see Section 5.3.3). On the other hand, it is known from our discussion in Section 5.4.3 and 5.4.4 that under low sensor node density and small communication range of the nodes both *SS* and *MS* leads to high data latency $\overline{\Delta}$. Figure 6.3(a) also validates these findings. In addition, it can be seen from Figure 6.3(a) that for both *SS* and *MS* sensor node density Γ has greater influence on data latency $\overline{\Delta}$ compared to communication range of the nodes, as reduction in node density Γ results in increased data latency $\overline{\Delta}$ even when communication range of the nodes is simultaneously increased. Furthermore, Figure 6.3(a)-(i) (plots the orthogonal projection of intersection point of *CaLEe* and *MS* on *r* and Γ plan) highlights how change in the node density and communication range of the nodes actually causes a change in the intersection points of the two curves (*MS* and *CaLEe*).

Similar to Section 6.2.1 and 6.2.2 here we observe that *CaLEe* leads to minimum energy dissipation Ψ and *SS* results in minimum latency $\overline{\Delta}$ amongst the three considered protocols. For example, consider the case when the system is operating at r=88m, Γ=20 nodes, *CaLEe* routing protocol is in use and user wants to reduce latency $\overline{\Delta}$. Now the question arises to which protocol the should we switch to reduce $\overline{\Delta}$ while losing minimum on energy dissipation Ψ. Results in Figure 6.3(a) shows that for considered values of *r* and Γ (88m, 20 nodes respectively) *MS* and *SS* have higher and lesser latency $\overline{\Delta}$ compared to *CaLEe* respectively. So, *SS* should be adopted to reduce latency $\overline{\Delta}$. Similarly, *MS* is not recommended for those pairs of *(r,* Γ*)* that leads to higher latency $\overline{\Delta}$ than *CaLEe* (part of *MS* curve above *CaLEe* in Figure 6.3(a)) because the use of *MS* in these cases leads to high latency $\overline{\Delta}$ as well as high energy dissipation Ψ compared to both *CaLEe* and *SS* thus improving nothing. Thus it can be concluded from Figure 6.3(a) and (b) that for all considered cases *SS* has minimum latency $\overline{\Delta}$ compared to *MS* and *CaLEe*, while *CaLEe* has minimum average energy dissipation Ψ per node compared to both *MS* and *SS*.



(i): Orthogonal projection of intersecting points of CaLEe and MS on Γ and *r* plan

(a): $\overline{\Delta}$ expressed as a function of Γ and *r*



(b): Ψ expressed as a function of Γ and *r*

Figure 6.3 Analysing the impact of r and Γ on the performance of the routing protocols

6.2.4 MOBILITY RADIUS OF THE SINK AND WSN DYNAMICS

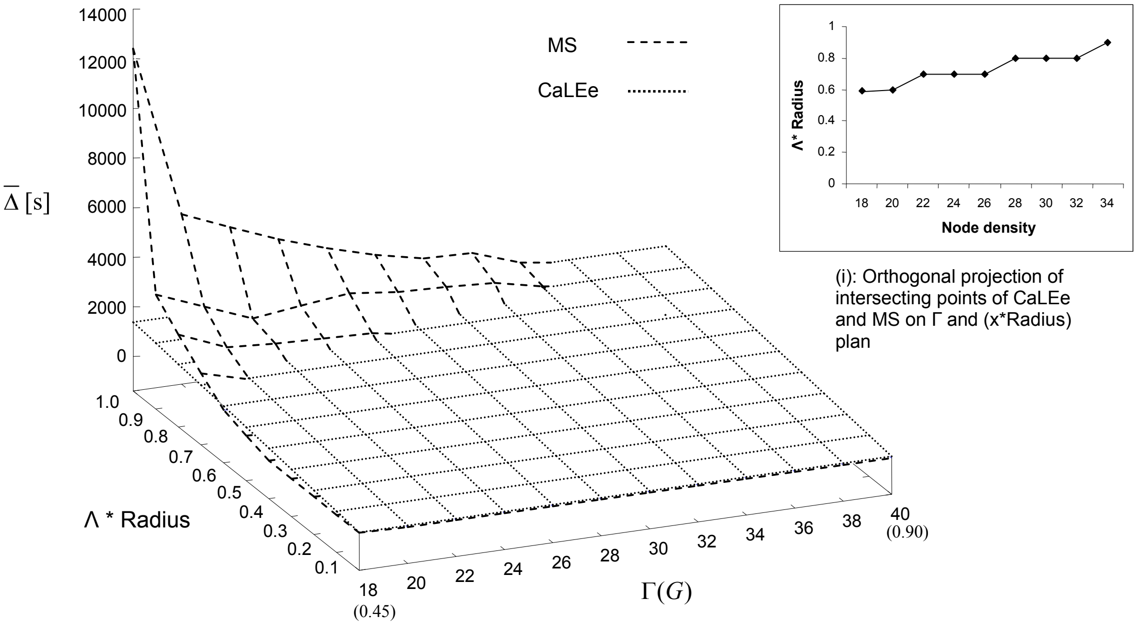
In this section, we analyze a scenario that is very close to real world operation of the WSN. We discuss the effect of three parameters (sensor node density Γ, throughput G of the WSN and radius of the mobility trajectory of the sink) simultaneously on the average end-to-end data latency $\overline{\Delta}$ and average energy dissipation Ψ per node of *SS*, *MS* and *CaLEe*.

It is known that during ad-hoc deployment of the WSN sensor nodes are densely deployed to fully cover the desired area. However, with the passage of time sensor node density reduces due to node failures. Considering constant data generation rate of the nodes reduction in the node density causes a drop in the throughput of the WSN hence resulting in variable performance of the deployed protocol ($\overline{\Delta}$, Ψ). In this section, we will analyze, given some initial configuration of the WSN and desired performance criterion, how one can ensure sustained performance of the network by varying radius of the mobility trajectory of the sink in the cases of *MS* and *CaLEe*.

In order to simulate above-mentioned scenario we varied the sensor node density Γ from 18 to 40 nodes (standard range of densities assumed during ad-hoc deployment of the nodes to ensure avoid coverage holes) under constant data generation rate of the nodes that also leads to change in the throughput G of the WSN from 0.45 to 0.9 packets per time slot. On the other hand, radius of the mobility trajectory of the sink is varied from *0m* (case of *SS*) to *800m* (periphery of the WSN). In Figure 6.4(a) and (b) three-dimensional graph show the average end-to-end data latency $\overline{\Delta}$ of WSN and average energy dissipation Ψ per node as a function of sensor node density Γ, throughput G of the WSN and radius of the mobility trajectory of the sink.

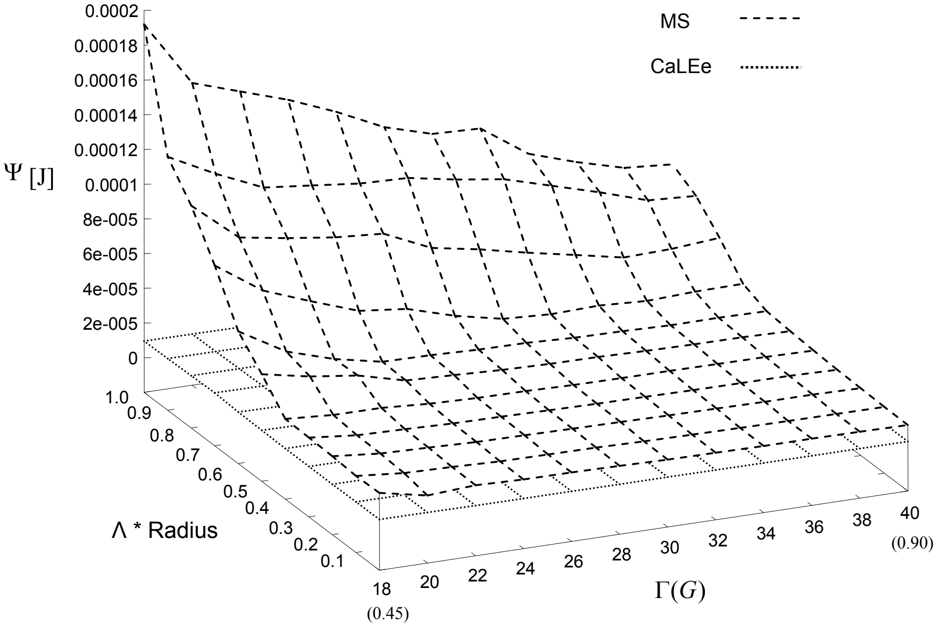
Figure 6.4(a) shows that after the deployment of the WSN with Γ = 40 node both the *SS* (Λ*Radius = 0) and *MS* show less end-to-end data latency $\overline{\Delta}$ compared to *CaLEe* for all radiuses of the mobility trajectories of the sink. However, when the node density falls below 36 nodes then *CaLEe* shows less data latency $\overline{\Delta}$ compared to *MS* for Λ*Radius = 1.0. Thus if the sink is moving along the periphery of the WSN and Γ<36 nodes then *CaLEe* should be used instead of *MS*. Moreover, further reduction in the Γ cause greater increase in data latency $\overline{\Delta}$ for *MS* as a result, even for small radius of the mobility trajectory of the sink *CaLEe* and *MS* have equal data latencies as shown in Figure 6.4(a)-(i) which plots the orthogonal projection of intersection point of *CaLEe* and *MS* on Λ*Radius and Γ plan. It also highlights how change in the node density and radius of the mobility trajectory of the sink actually causes a change in the intersection points of the two curves (*MS* and *CaLEe*). One way to ensure less data latency $\overline{\Delta}$ in *MS* compared to *CaLEe* is to reduce the radius of the mobility trajectory of the sink. It can be seen from Figure

6.4(a) that independent of the sensor node density if the radius of the mobility trajectory of the sink is below 0.6*R then *MS* shows less data latency $\overline{\Delta}$ compared to *CaLEe*. An interesting observation can be made in Figure 6.4(a)-(i), it shows that increase in the radius of the mobility trajectory of the sink has greater impact on the data latency compared to node density.



(i): Orthogonal projection of intersecting points of CaLEe and MS on Γ and (x*Radius) plan

(a): $\overline{\Delta}$ expressed as a function of R and $\Gamma(G)$



(b): Ψ expressed as a function of R and $\Gamma(G)$

Figure 6.4 Analysing the impact of R and $\Gamma(G)$ on the performance of the routing protocols

Figure 6.4(b) shows the results for average energy dissipation Ψ per node. It is observed that both decrease in the sensor node density Γ or increase in the radius of the mobility trajectory of the sink cause an increase in the average energy dissipation Ψ per node for *MS*, which revalidates our findings from Section 5.4.1 and 5.4.4 respectively (refer to these section for details). On the other hand, *CaLEe* achieves minimum Ψ amongst the three *(SS, MS and CaLEe)* routing schemes for all radiuses of the mobility trajectory of the sink as well as all considered node densities. *CaLEe* achieves these results because of the sensor field partitioning into sectors. This sectoring helps to reduce average path length that a data packet has to travel to reach the sink as well as channel contention and data congestion in the vicinity of the data collector nodes hence resulting in reduced average energy dissipation Ψ per node.

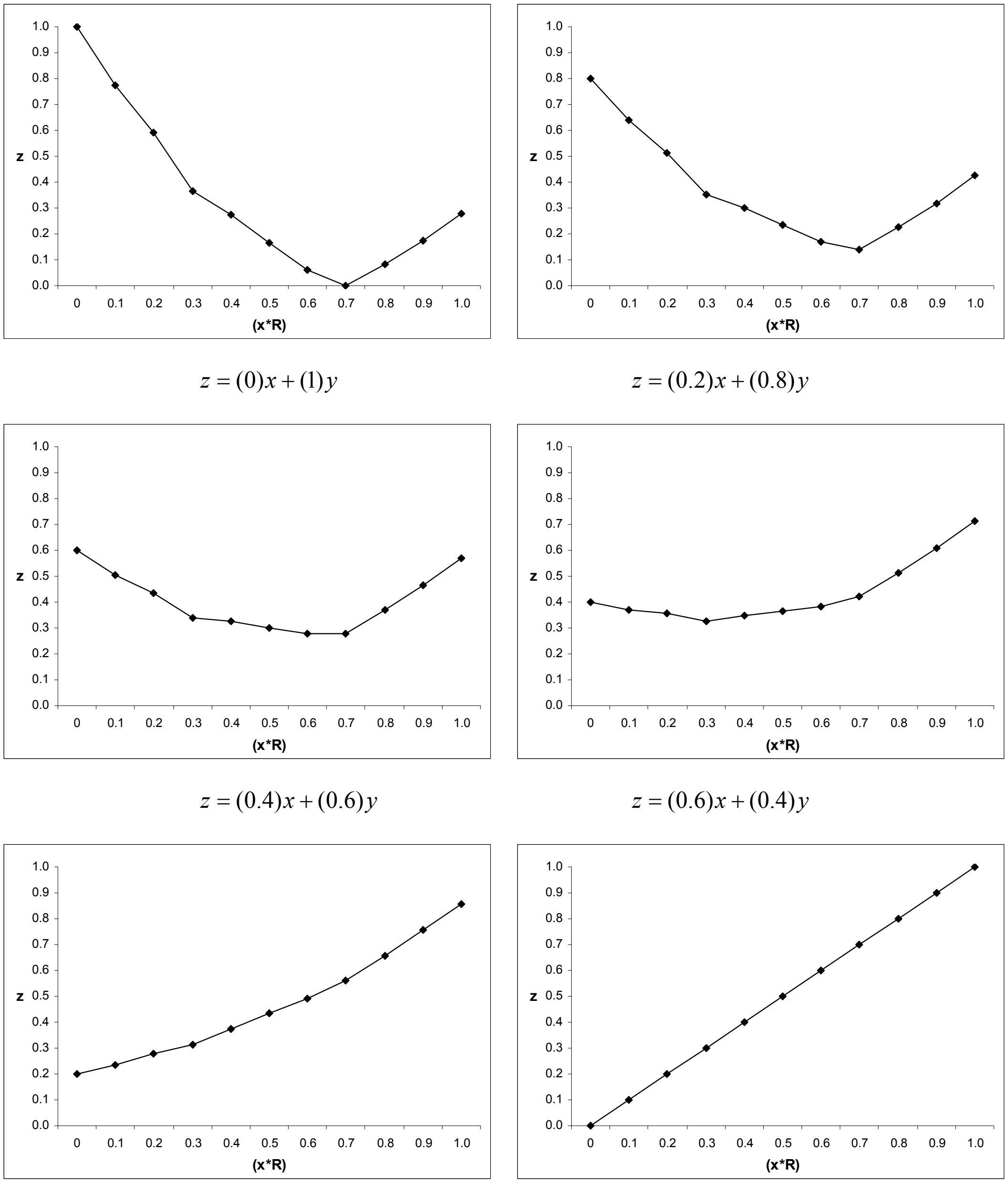
So, it can be inferred from above discussion that under high sensor node density *MS* provides less latency $\overline{\Delta}$ compared to *CaLEe* independent of the radius of the mobility path of the sink. However, reduction in the node density causes an increase in latency $\overline{\Delta}$ for *MS* and only way to keep it below the latency $\overline{\Delta}$ observed in *CaLEe* is to operate *MS* at small *sink mobility radius*.

6.3 COMBINED OBJECTIVE FUNCTION

In previous section, we have studied how change in the network configuration influences the performance of considered routing protocols *SS, MS* and *CaLEe*. However, for an end-user who does not have much technical knowledge it is very difficult to make an optimal selection of the routing protocol using graphs presented in Chapter 5 and 6. So, how can we assist him in making good decisions regarding the selection and configuration of the routing protocol? In the following, we address this issue with the help of an example about lifetime of the WSN and end-to-end data latency.

Discussion in Section 5.4 showed that for both *MS* and *CaLEe* the radius of the mobility trajectory of the sink greatly influences the end-to-end data latency and lifetime of the WSN. On the other hand, it has also been recognized that end-to-end data latency and lifetime of a WSN are two contradicting performance criteria. So, given some end-user requirements regarding end-to-end data latency and lifetime of the WSN, how can we identify optimum mobility trajectory of the sink. In order to address this issue, we investigate a combined objective function, which integrates end-to-end data latency *and* lifetime of the WSN:

$$z = \alpha x + (1-\alpha) y \qquad (6.1)$$



$z = (0)x + (1)y$

$z = (0.2)x + (0.8)y$

$z = (0.4)x + (0.6)y$

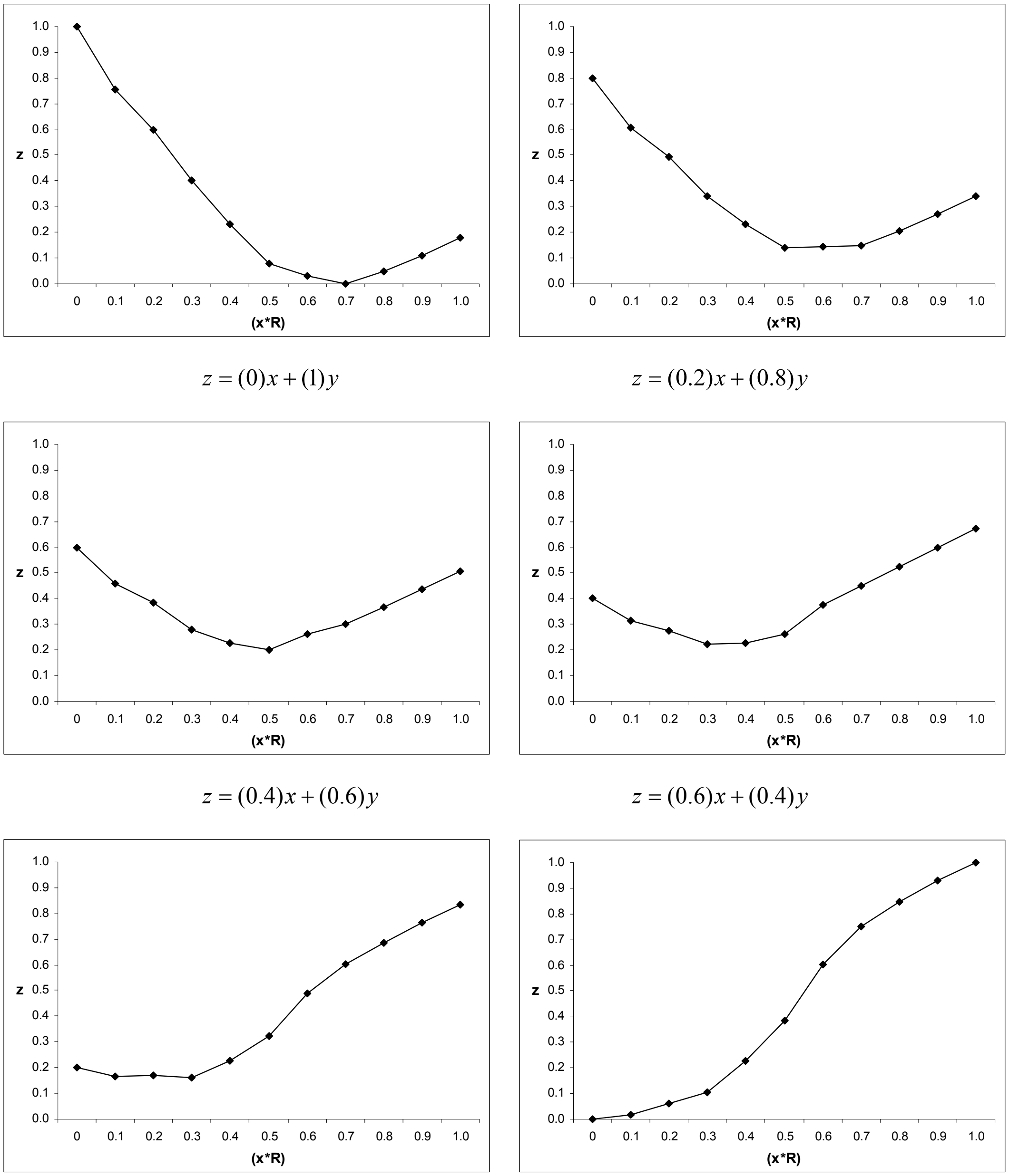
$z = (0.6)x + (0.4)y$

$z = (0.8)x + (0.2)y$

$z = (1)x + (0)y$

Figure 6.5 Performance results of *CaLEe* routing protocol

In Equation 6.1, x and y represent the observed end-to-end data latency and lifetime $(E_{\max} = \max_{i\in[1,N]} E_i$, where E_i is energy dissipation by the node $i)$ of the WSN, respectively, normalized to a scale between 0 (minimum observed value) and 1 (maximum observed value). α and (1- α) are the weights which end-user assigns to data latency and lifetime of the WSN, respectively. α can take values in (0, 1). z defines the performance of the considered routing protocol. Since z is the sum of energy dissipation *(E_{max})* and end-to-end data latency, smaller values of z indicate better performance of the WSN.



$z = (0)x + (1)y$

$z = (0.2)x + (0.8)y$

$z = (0.4)x + (0.6)y$

$z = (0.6)x + (0.4)y$

$z = (0.8)x + (0.2)y$

$z = (1)x + (0)y$

Figure 6.6 Performance results of *SS(x*R=0)* and *MS* routing protocols

In order to show the utility of Equation 6.1 we simulated *MS* and *CaLEe* routing protocols by varying the radius of the mobility trajectory of the sink from 0*R to 1.0*R. Observed lifetime *(E_{max})* and data latency are then normalized to a scale between 0 and 1. End-user requirements regarding the lifetime of the WSN and the end-to-end data latency can also vary. As a result, α can take any value between 0 and 1. In Figure 6.5 and Figure 6.6 we plotted the performance results of *CaLEe*, *SS* and *MS* (In the case of *MS* when radius of mobility trajectory is equal to zero then it becomes equivalent to *SS*) when the user requirements regarding end-to-end data latency and lifetime *(E_{max})* of the WSN varies ($\alpha = 0, 0.2, 0.4, 0.6, 0.8, 1.0$).

Both Figure 6.5 and Figure 6.6 revalidate results from state-of-the-art that when maximum lifetime of a WSN is desired independent of the end-to-end data latency $(z = (0)x + (1)y)$ then the radius of the mobility trajectory of the sink should be set to $(\sqrt{2}/2) * R$ (minium value of z is obtained at $(\sqrt{2}/2) * R$). However, with increase in the requirement of low data latency (increasing α) we observe that optimal performance (minium value z) can only be achieved by reducing the radius of the mobility trajectory of the sink for both *CaLEe* and *MS*. Furthermore, if minium end-to-end data latency is desired independent of the lifetime of a WSN $(z = (1)x + (0)y)$ then radius of the mobility trajectory of the sink should be set to 0 that is the case of *SS*.

Thus, we identify the optimal value of one input parameter in our simulation model that is radius of the mobility trajectory of the sink, under varying user requirements regarding lifetime of the WSN and end-to-end data latency. Similarly, formulation given in Equation 6.1 can also be used to identify the optimal values for other input parameters, such as communication range *r* of the nodes, number of sectors and size of the WSN, etc., for three routing schemes considered.

6.4 IMPLICATIONS OF THE RESULTS IN REAL WORLD SCENARIOS

End-to-end data latency $\overline{\Delta}$ and average energy dissipation Ψ per node (for precise definition see Section 2.3.1) are two basic performance criterions for routing protocols. However, our studies in Chapter 5 and Section 6.2 showed that these two performance criterions are contradictory in nature as reduction in end-to-end data latency $\overline{\Delta}$ results in increased average energy dissipation Ψ and vice versa. As a result, programmer has to make a trade-off in favor of one or another during network deployment. However, during the operation of a WSN sensor node failure or changing user requirements can cause change in the network configuration that leads to degraded performance of a WSN. In order to cater such situation an adaptive mechanism is required that can sustain the performance of a WSN at desired level under changing network scenarios.

This section presents an initial investigation for the development of a meta-routing scheme that is composed of various sub-routing protocols (*SS, MS* and *CaLEe*). Depending on current state of the WSN and user requirements regarding end-to-end data latency $\overline{\Delta}$ and average energy dissipation Ψ, the meta-scheme adaptively switches between *SS, MS* and *CaLEe* OR vary the *sink mobility radius* in the case of *MS/CaLEe* to sustain the performance of a WSN. Before we go into the details of the meta-routing protocol in the following, we outline assumption being made about the WSN. It is assumed that the sink knows the boundary of the WSN identified using the *MoSBoD* algorithm (see Chapter 3). Utilizing this information about the boundary the sink estimate coordinates of the center of the WSN, partition the field into Λ sectors and starts to collect data from the field using *CaLEe* routing scheme. Now depending on the user requirements and state of the network parameters the sink can initiate a switch between *CaLEe*, *MS* and *SS*. However, following three questions arises,

- When and to which routing protocol switching should take place.
- How the meta-routing scheme switches between *CaLEe, SS* and *MS*?
- What is the switching overhead?

When and to which routing protocol switching should take place. Answer to this question can be easily derived from Section 6.2 that identifies the operational regions based on network configurations where one protocol performs better than the other.

How the meta-routing scheme switches between CaLEe, SS and MS? We know from Chapter 4 and 5 that the major difference between *SS, MS* and *CaLEe* is the placement of sink in the WSN. In *SS* the sink is static, placed at the center of the WSN, while in *MS* and *CaLEe* the sink is mobile. In both *SS* and *MS* every node routes data directly to the current location of the sink. On the other hand, in *CaLEe* every node routes data to the collector node located in its sector and the sink periodically visit each data collector node to retrieve the stored data. In order to switch from *CaLEe* to *SS/MS* the sink broadcasts a CHANGE message to the WSN informing each node about its current location. In the case of *SS,* the sink broadcasts this message after it has positioned itself at the centre of the WSN (calculated using information about the boundary of the WSN). In the case of *MS* (considering, the mobility trajectory is same as in *CaLEe*) the sink can broadcast this message after sojourning at a node on its mobility trajectory. Transformation from *CaLEe* to *SS/MS* completes once each node knows the current position of the sink in the WSN and starts to route messages directly towards the current position of the sink instead of the data collector nodes. Moreover, transformation from *SS/MS* to *CaLEe* can be made by broadcasting a CHANGE message from the sink to instruct each node in the WSN to route data to the corresponding data collector node instead of the sink. Similarly, transformation from *SS* to *MS* and *MS* to *SS* can be made with one sensor field flooding from the sink once it reaches the desired mobility trajectory or the center of the WSN respectively. Time required for each transformation is the time taken by the CHANGE message from the sink to reach every node in the WSN.

What is the switching overhead? Each switching amongst the considered routing protocols require the sink to flood the field once and the switching time (time required to switching from one protocol to another) is the time taken by the CHANGE message from the sink to reach every node in the WSN. So, it can be concluded that both the switching effort and switching time heavily depends on the size of the WSN as greater the size of the WSN higher will be the switching effort and switching time.

Section 6.2.4 provides a best example for the implementation of such meta-routing protocol. Consider that a sensor field is deployed with a high sensor node density equal to 40 nodes and $G=0.9$ per time slot. Our aim is to acquire least end-to-end data latency $\overline{\Delta}$. It can be seen from Figure 6.4 that *MS* has less $\overline{\Delta}$ compared to *CaLEe* when $\Gamma=40$ thus *MS* can be adopted. However, with the passage of time sensor nodes deplete their energy and start to fail resulting in the reduction of the node density. It can be seen from Figure 6.4 that once node density fall below 36, $\overline{\Delta}$ in *MS* grows higher than *CaLEe* thus in order to achieve minium end-to-end data latency $\overline{\Delta}$ protocol switching is required from *MS* to *CaLEe*.

6.5 SUMMARY

In this chapter, we addressed a fundamental question, given some network configuration and user requirements about $\overline{\Delta}$, Ψ, how one can decide which protocol (*SS, MS or CaLEe*) to use? In order to answer this question we defined various operational scenarios based on different combinations of r, Γ, G and radius of the mobility trajectory of the sink. Our results showed that user requirements have a major role to play in the selection of the routing protocol. For example, if minimum average energy dissipation Ψ per node is desired (average end-to-end data latency $\overline{\Delta}$ is not an issue of interest) then *CaLEe* is the best option for network configuration. Similarly, *SS* provides the least end-to-end data latency $\overline{\Delta}$ except for the cases when the WSN is configured to operate at low Γ (<=18 nodes) and high G (=1.0 packets per time slot). Furthermore, for small r, low Γ and high G *CaLEe* showed competitive results in terms of end-to-end data latency $\overline{\Delta}$ compared to *SS* and *MS*. However, increase in r or Γ and reduction in G leads to reduction in end-to-end data latency $\overline{\Delta}$ for both *SS* and *MS* compared to *CaLEe*. However, average energy dissipation Ψ per node remains minimal in *CaLEe*.

During the operation of the WSN, changing user requirements regarding $\overline{\Delta}$, Ψ and/or changing network configuration (r, G, Γ) can effect the performance of the network. In order to sustain the performance of the network at certain desired level an initial discussion on the development of meta-routing protocol is also presented. The meta-routing scheme is capable of dynamically adapting to any of the sub-routing protocols depending on the user requirements and network dynamics to sustain the performance of the network ($\overline{\Delta}$, Ψ) at desired level.

CHAPTER 7: CONCLUSION

In this dissertation, we address the problem of energy efficient data routing with acceptable bounded end-to-end data latency in large-scale applications of wireless sensor networks. We approach the problem at two levels: post deployment initializations and route establishment/data routing protocol.

Post deployment initializations are required to transform randomly deployed isolated nodes into a powerful working unit that can be used to retrieve useful information about the parameter of interest. Execution of these initialization schemes leads to high-energy dissipation from the nodes leading to reduced lifetime of the network. We focused on one of the initialization requirement that is boundary identification of the WSN.

We presented a mobile sink based boundary identification scheme for WSN. Instead of relaying on sensor field flooding proposed scheme exploits sink mobility to identify the edge nodes that are connected linearly to obtain the boundary of a WSN. Analysis of the proposed scheme using analytical and simulation methods have shown that it leads to more than sixty percent increase in the lifetime of a WSN compared to current state-of-the-art. Furthermore, increase in the size of a sensor field results in further energy gains over conventional flooding based boundary identification schemes for WSN. One potential limitation of our scheme is high completion time that the sink needs to identify the complete boundary. However, it has also been shown that if high mobility speed of the sink and/or small average distance between the nodes is considered then *MoSBoD* shows comparable completion time with respect to the flooding based state-of-the-art schemes.

In the second phase of our research, we used the information obtained during sensor field initialization *(boundary identification)* for the development of energy efficient routing protocol for wireless sensor networks, called Congestion avoidance for Low latency and Energy efficiency (*CaLEe)*. Proposed protocol is based on sensor field partitioning into sectors and sink mobility along a fixed trajectory in a WSN. In each sector a data collector node is selected that is responsible of collecting data from the nodes located in its sector. On the other hand, sink moves from one data collector node to next for data retrieval from the sensor field.

Our analysis showed that *CaLEe* routing protocol improves the lifetime of a WSN compared to *SS* and *MS* by the factor of 4.5 and 3.25 respectively. Major limitation of the *CaLEe* routing protocol can be the large end-to-end data latency caused by the slow mobility speed of the sink. However, simulation results have shown that for small r and Γ, high G and large size of a WSN *CaLEe* has competitive end-to-end data latency compared to *SS* and *MS*. However, with an increase in r/Γ and/or reduction in G/size of a WSN average delay decreases in both *SS* and *MS* and becomes less than *CaLEe*. However, average energy dissipation per node remains least in the case of *CaLEe*.

During the analysis of *CaLEe* compared to current state-of-the-art routing schemes it has been observed that no single routing protocol is capable of providing best-case solution in all scenarios. Furthermore, due to error prone nature of the sensor nodes network configuration may change during the operation of the network hence, resulting in variable performance (energy efficiency, data latency) from same routing protocol. Therefore, last part of the thesis addresses issue such as, given some initial network configurations and user requirements how one can decide which routing protocol to use. In order to address this issue we presented a comparison of *SS, MS* and *CaLEe*. During the simulations we analyze the impact of six dimensional input space (communication range r of the nodes, node density Γ, throughput G of the WSN, size of a WSN, number of sectors in a WSN and radius of the mobility trajectory of the sink), which defines a network configuration on the performance (energy efficiency, data latency) of considered routing protocol. Obtained results are then used to identify the operational regions where each of the considered protocol performs better than the other.

7.1 FUTURE WORK

Sink mobility is mostly used to improve the lifetime of a WSN by evenly distributing the routing load amongst the sensor nodes. In contrast, we have exploited the sink mobility for boundary identification of a WSN. Similarly, sink mobility along the boundary of a WSN can be used easily for sensor field localization where sink localizes the edge nodes using its GPS and edge nodes act as beacon nodes for the rest of the sensor field. Thus, by utilizing a mobile sink equipped with a GPS one can localize the entire sensor field in a cost effective manner. Furthermore, in order to achieve the time efficiency in mobile sink based solutions use of multiple mobile can also be exploited.

Although mobile sink based routing schemes extensively improves the lifetime of a WSN but they also leads to considerable increase in the end-to-end data latency. In this thesis, we have discussed certain network scenarios where end-to-end data latency in *CaLEe* is comparable to *SS*. However, other techniques such as prediction based routing etc. to reduce end-to-end data latency in a mobile sink based WSN are still largely open for research. In such a scenario, nodes can be made intelligent enough to predict the location of the sink in a WSN and route data directly to the sink. This avoids control messages amongst the nodes for position tracking of the sink that result in increased lifetime of a WSN. Furthermore, it also helps to reduce the delay by eliminating the need of flooding the sink-position-update-message as each node can predict the current position of the sink in WSN.

APPENDIX – I: TABLE OF FIGURES

Figure 2.1 A typical wireless sensor network (WSN) 5
Figure 2.2 Antenna types 7
Figure 2.3 Sensor nodes 8
Figure 2.4 Habitat Monitoring 11
Figure 2.5 Environmental monitoring 12
Figure 2.6 Shooter localization in urban terrain [39] 12
Figure 2.7 Golden Gate Bridge monitoring using WSN [42] 13
Figure 2.8 Revolutionizing medical science [47] 14
Figure 2.9 Sensor node deployment strategies 17
Figure 2.10 Data exchange in BMAC protocol [65] 18
Figure 2.11 Transition between different sensor modes [70] 20

Figure 3.1 A 5-flower structure [84] 31
Figure 3.2 Perimeter-based boundary node detection approaches [78] 33
Figure 3.3 Network model 34
Figure 3.4 BiGDog 35
Figure 3.5 Identification of the starting edge node 38
Figure 3.6 Neighboring edge node identification 40
Figure 3.7 Special cases for neighbor edge node identification in Module-2 41
Figure 3.8 Edge node identification 42
Figure 3.9 Obtained boundary shapes using MoSBoD and M- MoSBoD 43
Figure 3.10 Modified boundary definition 44
Figure 3.11 Removing redundant edge nodes 45
Figure 3.12 Model sensor field for comparing completion time 50
Figure 3.13 Edge nodes and their neighboring node 55
Figure 3.14 $E_{M\text{-}MoSBoD}$ and $E_{Flooding}$ as the percentage of E_{MoSBoD} 56

Figure 4.1 A three tier architecture [109] 62
Figure 4.2 A two-tier data dissemination approach [33] 63
Figure 4.3 Sink mobility patterns in [113] 64
Figure 4.4 Tracking the position of randomly moving sink in WSN [114] 65
Figure 4.5 Interest dissemination using directional antenna at a mobile sink [115] 65
Figure 4.6 A mobile sink traversing the sensor field along a fixed path [116] 66
Figure 4.7 Joint mobility and routing strategy 67

Figure 4.8 Controlled sink mobility in a WSN [122] 68
Figure 4.9 Routing schemes for WSN 71
Figure 4.10 Partitioned sensor field 72
Figure 4.11 Clustered sensor field with data congestion at node N 78
Figure 4.12 Data storage and retrieval from the buffer node 79

Figure 5.1 Temporal evolution of the sensor state [135] 82
Figure 5.2 Simulation set up of a WSN 89
Figure 5.3 Energy dissipation patterns per node in SS, MS and CaLEe 91
Figure 5.4 End-to-end data latency experienced by each node in SS, MS and CaLEe 93
Figure 5.5 End-to-end data latency experience by each node 94
Figure 5.6 Effect of sink mobility radius on Ψ and $\overline{\Delta}$ 96
Figure 5.7 Sector of a WSN with various positions of a data collector node 97
Figure 5.8 Number of sectors vs. Ψ for *CaLEe* routing protocol 98
Figure 5.9 Varying communication range of the sensor nodes 100
Figure 5.10 Effect of sensor node density on performance of a WSN 102
Figure 5.11 Effect of throughput on performance of a WSN 104
Figure 5.12 Analyzing the effect of size of a WSN on the performance of SS, MS and CaLEe 106

Figure 6.1 Analyzing the impact of *G* and Γ on the performance of the routing protocols 111
Figure 6.2 Analyzing the impact of *G* and *r* on the performance of the routing protocols 113
Figure 6.3 Analysing the impact of r and Γ on the performance of the routing protocols 116
Figure 6.4 Analysing the impact of *R* and $\Gamma(G)$ on the performance of the routing protocols ... 118
Figure 6.5 Performance results of *CaLEe* routing protocol 120
Figure 6.6 Performance results of *SS(x*R=0)* and *MS* routing protocols 121

APPENDIX – II: LIST OF VARIABLES

N	Total number of sensor nodes in the field
N_{edge}	Total number of edge nodes in the sensor field
$N_{sectors}$	Total number of sectors
n_i	Sensor node i
n_{en}	Edge node
(x_i, y_i)	Position coordinates of node i
t	Time interval
Γ	Neighbor node density
$\overline{d}$	Average distance between two sensor nodes
η	Average path length from source to sink
ρ	A generic route that includes the source and the relays but not the sink
$M(i, .., i+n)$	Message M with parameters $i, .., i+n,$ where n is an arbitrary number
NL_i	List of neighboring nodes of node i
V_{sink}	Mobility speed of the sink
$t_{edgeNode}$	Time required by the sink to identify the next edge node
$e_{edgeNode}$	Total energy dissipation of the nodes during next edge node identification
$t_{i \rightarrow i+1}$	Time taken by the sink to travel from node i to node $i+1$
t_{1hop}	Time taken by a message to travel 1 hop distance
d_{max}	Maximum hop count distance of a node from the sink
R	Radius of the circular sensor field
r	Communication range of a sensor node
t_{flood}	Time required to flood the sensor field once
Num_{floods}	Number of sensor field flooding
T	Completion time of an algorithm
E	Total energy dissipation of a WSN
e_i	Energy dissipation for performing task i
E^t	Energy spent by a node during transition from sleep to active mode
E^{ele}	Energy spent by a node in transceiver electronics
E^{proc}	Energy spent by a node during message processing
E^{rx}	Energy spent by a node during message reception = $E^{ele} + E^{proc}$

E^{tx}	Energy spent by a node during message transmission = $E^{ele} + E^{proc} +$ energy consumption at amplifier
g	Data generation at each node follows a Bernoulli process with parameter g
q, p	Geometrically distributed random variables that define sleep and active transition of a node
G	Sum of the throughputs of all sensor nodes = throughput of the sensor field
L	Length of the mobility trajectory of the sink
S_{sink}	Sojourn time of the sink
e^i	Total energy dissipation of an arbitrary node i
Ψ	Average energy dissipation per sensor node = $\sum_{i=0}^{N} e^i / N$
Φ^{WSN}	Lifetime of a WSN = time until the first node fails (or runs out of energy)
Δ^i	End-to-end data latency experienced by an arbitrary node i = time taken by the data from node i to sink
$\overline{\Delta}$	Average end-to-end data latency of WSN = $\sum_{i=1}^{N} \Delta^i / N$
$\Delta_{\max}$	Maximum end-to-end data latency of WSN = maximum end-to-end delay experienced by a node

REFERENCES

1. C.S. Raghavendra, K.M.S.a.T.Z., *“Wireless Sensor Networks”*,. 2004: Kluwer Academic Publishers.
2. Akyildiz, I.F., Weilian Su, Sankarasubramaniam, Y., Cayirci, E., , *A survey on sensor networks.* IEEE Communications Magazine, August 2002. **40**(8): p. 102- 114.
3. Eugene Shih, S.C., N. I. R. M. A. S. A. W. and Chandrakasan, A. . *Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks*. in *MOBICOM 01*. 2001.
4. M.A. Youssef, M.Y.a.A., K., . *A constrained shortest-path energy-aware routing algorithm for wireless sensor networks*. in *WCNC*. March 2002.
5. P. Agrawal, T.T.a.A., A., . *A lightweight protocol for wireless sensor networks*. in *proceedings of WCNC*. March 2003.
6. S.-J. Park, R.V., R. S. and I.F.Akyildiz, . *A scalable approach for reliable downstream data delivery in wireless sensor networks*. in *proceedings of MOBIHOC*. 2004.
7. Wei Ye, J.H., D. E. *An energy-efficient mac protocol for wireless sensor networks*. in *proceedings of INFOCOM*. 2002.
8. Wendi R. Heinzelman, A.C.a.B. *H., Energy-efficient communication protocol for wireless microsensor networks*. in *proceedings of the 33rd Hawaii International Conference on System Sciences*. 2000.
9. Majid I. Khan, W.N.G., Günter Haring. *Energy consumption vs. latency in a new boundary identification method for WSNs with a mobile sink*. in *proceedings of the 6th ACM international Symposium on Mobility Management and Wireless Access, MobiWac '08*. October 30 - 31, 2008. Vancouver, British Columbia, Canada: ACM, New York, NY.
10. Majid I. Khan, W.N.G., Günter Haring. *Identifying the Boundary of a Wireless Sensor Network with a Mobile Sink*. in *proceedings of the 7th international conference on AD-HOC Networks & Wireless*. September 10-12, 2008. Nice, France.
11. Majid I. Khan, W.N.G., Guenter Haring, *Congestion Avoidance and Energy Efficient Routing Protocol for Wireless Sensor Networks with a Mobile Sink.* Journal of Networks (JNW), 2007. **2**(6): p. 42-49.
12. Majid I. Khan, W.N.G., Guenter Haring. *In-Network Storage Model for Data Persistence under Congestion in Wireless Sensor Networks*. in *First International Conference on Complex, Intelligent and Software Intensive Systems*. 2007. Vienna, Austria.
13. Zhu, Y., *Energy-efficient communication strategies wireless sensor networks*, in *School of Electrical and Computer Engineering*. 2007, Georgia Institute of Technology.
14. N. B. Priyanath, A.C., and H. Balakrishna. *The cricket location support system*. in *proceedings of Mobile Computing and Networking*. 2000.
15. A. Savvides, C.H., and M. Srivastava. *Dynamic fine-grained localization in ad-hoc networks of sensors. .* in *proceedings of ACM MobiCom ’01*. July 2001.
16. A. Savvides, H.P., and M. Srivastava. . *The bits and flops of the n-hop multilateration primitive for node localization problems.* in *proceedings of ACM WSNA ’02,* . September 2002.
17. A. Harter, A.H., P. Steggles, A.Ward, and P.Webster *The anatomy of a context-aware application*. in *proceedings of MOBICOM*. 1999.
18. *Antenna Model, http://www.myerseng.com/Patron/SL-5001-B357-Elevation.gif, [cited. 21.01.2009].*
19. *Antenna Model, http://www.hdtvprimer.com/ANTENNAS/cm4228el.gif, [cited, 21.01.2009].*
20. Lee, Y.a.C., H. *A Light-weight and Scalable Localization Technique Using Mobile Acoustic Source*. in *Sixth IEEE international Conference on Computer and information Technology (Cit'06)*. September 20 - 22, 2006. Washington, DC: IEEE Computer Society.
21. Chia-Ho Ou, K.-F.S. *A Range-free Localization Scheme Using Mobile Reference Nodes for Wireless Sensors*. in *International Computer Symposium*. Dec 15-17, 2004. Taipei, Taiwan.
22. Akcan, H., Kriakov, V., Brönnimann, H., and Delis, A. *GPS-Free node localization in mobile wireless sensor networks*. in *proceedings of 5th ACM international Workshop on Data Engineering For Wireless and Mobile Access MobiDE '06*. June 25 - 25, 2006. Chicago, Illinois, USA ACM, New York, NY, .
23. Chatzigiannakis, I., Kinalis, A., Nikoletseas, S., and Rolim, J. *Fast and energy efficient sensor data collection by multiple mobile sinks*. in *proceedings of 5th ACM international Workshop on Mobility Management and Wireless Access, MobiWac '07*. October 22 - 22, 2007. Chania, Crete Island, Greece ACM, New York, NY,.
24. Holger Karl, A.W., *Protocols and Archtectures for Wireless Sensor Networks*. 2005: John Wiley & Sons, Ltd. 497.

25. T. Liu, C.S., P. Zhang, and M. Martonosi. *Implementing Software on Resource-Constrained Mobile Sensors: Experiences with Impala and ZebraNet*. in *The Second International Conference on Mobile Systems, Applications, and Services, Mobisys 2004* June, 2004.
26. Pei Zhang, C.M.S., Steve A. Lyon, and Margaret Martonosi. *Hardware Design Experiences in ZebraNet* in *proceedings of Second ACM Conference on Embedded Networked Sensor Systems, SenSys*. Nov, 2004. .
27. Arsalan Tavakoli, J.Z., Sang H. Son,. *Group-Based Event Detection in Undersea Sensor Networks* in *proceedings of 2nd International Workshop for Networked Sensing Systems* June 2005. Los Angeles, CA,.
28. Ding, M.C., D. Xing, K. Cheng, X. , . *Localized fault-tolerant event boundary detection in sensor networks*. in *IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*. . 13-17 March 2005.
29. Yingqi Xu, J.W., Wang-Chien Lee. *Prediction-Based Strategies for Energy Saving in Object Tracking Sensor Networks*. in *IEEE International Conference on Mobile Data Management (MDM'04)*. 2004.
30. Wen-Chih Peng, Y.-C.T., Chih-Yu Lin, *Efficient In-Network Moving Object Tracking in Wireless Sensor Networks.* in journal of IEEE Transactions on Mobile Computing, Aug. 2006. **5**(8): p. 1044-1056.
31. Lee, S., Cha, H., and Ha, R., *Energy-aware location error handling for object tracking applications in wireless sensor networks.* in journal of Computer Communication, May 2007. **30**(7): p. 1443-1450.
32. Chalermek Intanagonwiwat , R.G., Deborah Estrin. *Directed diffusion: a scalable and robust communication paradigm for sensor networks*. in *proceedings of 6th annual international conference on Mobile computing and networking*. August 06-11, 2000. Boston, Massachusetts, United States
33. Ye, F., Luo, H., Cheng, J., Lu, S., Zhang, L. *A two-tier data dissemination model for large-scale wireless sensor networks*. in *8th Annual international Conference on Mobile Computing and Networking, MobiCom '02*. September 23 - 28, 2002. Atlanta, Georgia, USA: ACM, New York, NY.
34. *Great Duck Island, http://www.coa.edu/html/historyofgreatduck.htm [cited. 21.01.2009]*. [cited.
35. *The ultimate on the fly network, www.wired.com/wired/archive/11.12/network.html, [cited, 28.09.2008]*.
36. *Zebra Net, http://www.princeton.edu/~csadler, [cited, 15.06.2007]*.
37. Hu, W., Tran, V. N., Bulusu, N., Chou, C. T., Jha, S., and Taylor, A. . *The design and evaluation of a hybrid sensor network for Cane-Toad monitoring*. in *proceedings of 4th international Symposium on information Processing in Sensor Networks*. April 24 - 27, 2005. Los Angeles, California: IEEE Press, Piscataway, NJ,.
38. Werner-Allen, G., Lorincz, K., Welsh, M., Marcillo, O., Johnson, J., Ruiz, M., and Lees, J. . *Deploying a Wireless Sensor Network on an Active Volcano*. in *proceedings of IEEE Internet Computing* March 2006.
39. Maroti M., S.G., Ledeczi A., Sztipanovits J. *Shooter Localization in Urban Terrain,*. in *Computer, 37, 8*. August, 2004.
40. Volgyesi P., B.G., Nadas A., C. Nash, Ledeczi A., *Shooter Localization and Weapon Classification with Soldier Wearable Networked Sensors,* . January 15, 2007.
41. Nemeroff, J.G., L. Hampel, D. DiPierro, S. . *Application of sensor network communications,* . in *proceedings of IEEE military communication conference, MILCOM,* . 2001.
42. Kim, S., Pakzad, S., Culler, D., Demmel, J., Fenves, G., Glaser, S., and Turon, M. . *Health monitoring of civil infrastructures using wireless sensor networks.* . in *proceedings of 6th international Conference on information Processing in Sensor Networks. IPSN '07*. April 25 - 27, 2007. (Cambridge, Massachusetts, USA,: ACM, New York, NY.
43. C. Grosse, S.D.G., M. Krüger. *Condition monitoring of concrete structures using wireless sensor networks and MEMS.* . in *proceedings of Smart Structures and Materials: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*. 2006.
44. Li, M.a.L., Y *Underground structure monitoring with wireless sensor networks.* . in *6th international Conference on information Processing in Sensor Networks, IPSN '07* April 25 - 27, 2007. Cambridge, Massachusetts, USA,: ACM, New York, NY.
45. Essa, I.A., *Ubiquitous sensing for smart and aware environments.* IEEE Personal Communications, Oct. 2000: p. 47-49.
46. *http://www.onworld.com/smbldgs/index.html*. [cited 03-06-2008].
47. Ouwerkerk, M. *Display Applications &Technologies*. September 12, 2007 [cited.
48. E. Jovanov, D.R., J. Price, A. Moore, J. Chapman, A. Krishnamurthy. *Patient Monitoring Using Personal Area Networks of Wireless Intelligent Sensors,* . in *Proc. 38th Annual Rocky Mountain Bioengineering Symposium, Biomedical Sciences Instrumentation*. April 2001. Copper Mountain.
49. Tia Gao, D.G., and Matt Welsh. . *Improving Patient Monitoring and Tracking in Emergency Response,*. in *Proceedings of the International Conference on Information Communication Technologies in Health,* . July 2005.

50. C. A. Otto, E.J., A. Milenkovic. *A WBAN-based System for Health Monitoring at Home*. in *Proceedings of the 3rd IEEE EMBS International Summer School and Symposium on Medical Devices and Biosensors (ISSS-MDBS 2006),* . September 2006. Boston, MA,.
51. L. Evers, M.J.J.B., M. Marin-Perianu, R. Marin-Perianu, P. J. M. Havinga *Wireless Sensor Networks and Beyond: A Case Study on Transport and Logistics*. in *International Workshop on Wireless Ad-hoc Networks (IWWAN 2005),* . May 23-26, 2005. London UK,.
52. A. Timm-Giel, K.K., M. Becker, C. Görg,. *Wireless Sensor Networks in Wearable and Logistic Application*. [cited.
53. Jedermann, R.B., C.; Westphal, D.; Lang, W. , *Applying autonomous sensor systems in logistics; Combining Sensor Networks, RFIDs and Software Agents.* In journal of Sensors and Actuators A (Physical), November 2006. **132**(1): p. 370-375.
54. Pierce, F.J.a.E., T. V. , *Regional and on-farm wireless sensor networks for agricultural systems.* Eastern Washington. Comput. Electron. Agric. 61, April 2008: p. 32-43.
55. A. Baggio. *Wireless Sensor Networks in Precision Agriculture*. in *ACM Workshop Real-World Wireless Sensor Networks*. 2005.
56. P. Corke and P. Sikka. *Results from the Farm*. in *Proc. 3rd Workshop Embedded Networked Sensors (EmNets 06)*. 2006.
57. I. F. Akyildiz, W.S., Y. Sankasubramaniam, and E. Cayirci, *Wireless Sensor Networks: A Survey.* Journal of Computer Networks, 2002. **38**: p. 393–422.
58. F. Koushanfar, M.P., A. Sangiovanni-Vincentelli, *Fault-Tolerance in Sensor Networks*, in *Handbook of Sensor Networks*. 2004, CRC press. p. Section VIII, no. 36.
59. N. Bulusu, D.E., L. Girod, J. Heidemann, . *Scalable coordination for wireless sensor networks: self-configuring localization systems*. in *International Symposium on Communication Theory and Applications (ISCTA 2001)*. July 2001. Ambleside, UK.
60. E. Shih, S.C., N. Ickes, R. Min, A. Sinha, A. Wang, A. Chandrakasan. *Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks*. in *ACM MobiCom*. July 2001. Rome, Italy.
61. E.M. Petriu, N.D.G., D.C. Petriu, D. Makrakis, V.Z. Groza, *Sensor-based information appliances.* IEEE Instrumentation and Measurement Magazine, December 2000: p. 31–35.
62. Cerpa, J.E., M. Hamilton, J. Zhao. *Habitat monitoring: application driver for wireless communications technology*. in *proceedings of ACM SIGCOMM'2000*. April 2001. Costa Rica.
63. J.M. Rabaey, M.J.A., J.L. da Silva Jr., D. Patel, S. Roundy, *PicoRadio supports ad hoc ultra-low power* wireless networking, IEEE Computer Magazine 2000: p. 42–48.
64. Polastre, J., Hill, J., and Culler, D. *Versatile low power media access for wireless sensor networks*. in *proceedings of the 2nd international Conference on Embedded Networked Sensor Systems (SenSys '04)*. November 03 - 05, 2004. Baltimore, MD, USA: ACM, New York, NY.
65. Kerri Stone, M.C., *Efficient duty cycling through prediction and sampling in wireless sensor networks.* Wireless Communications and Mobile Computing, 2007. **7**(9): p. 1087-1102.
66. Wang, Y., Liu, X., and Yin, J. . *Requirements of Quality of Service in Wireless Sensor Network*. in *international Conference on Networking, international Conference on Systems and international Conference on Mobile Communications and Learning Technologies, ICNICONSMCL* April 23 - 29, 2006: IEEE Computer Society, Washington, DC.
67. Fan, K., Liu, S., and Sinha, P. *Scalable data aggregation for dynamic events in sensor networks*. in *proceedings of the 4th international Conference on Embedded Networked Sensor Systems (SenSys '06)*. October 31 - November 03, 2006. Boulder, Colorado, USA: ACM Press, New York, NY.
68. Gao, J., Guibas, L., Milosavljevic, N., and Hershberger, J. . *Sparse data aggregation in sensor networks.* . in *proceedings of the 6th international Conference on information Processing in Sensor Networks (IPSN '07)*. April 25 - 27, 2007. Cambridge, Massachusetts, USA: ACM Press, New York, NY.
69. Laura Galluccio, A.T.C., Sergio Palazzo. *Concert: aggregation-based congestion control for sEnsoR networks*. in *proceedings of the 3rd international conference on Embedded networked sensor systems*. 2005. San Diego, California, USA.
70. Wang, L.a.X., Y. , *A survey of energy-efficient scheduling mechanisms in sensor networks.* Mobile Network Applications, Oct. 2006. **11**(5): p. 723-740.
71. Jana van, G. and R. Jan, *Lightweight time synchronization for sensor networks*, in *Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*. 2003, ACM: San Diego, CA, USA.
72. Nuo Wei, Q.G., Jia-liang Lv, Yuan-yuan Yang,. *A Spanning Subtree Based Multi-channel Time Synchronization Algorithm for Sensor Networks*. in *International Conference on Embedded Software and Systems*. 2009.

73. Philipp Sommer, R.W. *Gradient Clock Synchronization in Wireless Sensor Networks*. in *IPSN'09*. April 13–16, 2009. San Francisco, California, USA: ACM.
74. Yu, G. and H. Tian, *Data forwarding in extremely low duty-cycle sensor networks with unreliable communication links*, in *Proceedings of the 5th international conference on Embedded networked sensor systems*. 2007, ACM: Sydney, Australia.
75. A. Srinivasan and J. Wu, *A Survey on Secure Localization in Wireless Sensor Networks*, in *Encyclopedia of Wireless and Mobile Communications*, B.F. (ed.), Editor. 2008, CRC Press, Taylor and Francis Group.
76. Wang, Y., Gao, J., and Mitchell, J. S. *Boundary recognition in sensor networks by topological methods*. in *proceedings of MobiCom*. 2006: ACM Press, New York, NY.
77. Nezhad, A.A., Makrakis, D., and Miri, A. . *Anonymous topology discovery for multihop wireless sensor networks.* . in *proceedings of 3rd ACM Workshop on QoS and Security For Wireless and Mobile Networks, Q2SWinet '07*. October 22 - 22, 2007. Chania, Crete Island, Greece: ACM, New York, NY.
78. Zhang, C., Zhang, Y., and Fang, Y. . *Localized coverage boundary detection for wireless sensor networks*. in *Proceedings of the 3rd international Conference on Quality of Service in Heterogeneous Wired/Wireless Networks QShine 2006*. August 07 - 09, 2006 (Waterloo, Ontario, Canada: ACM Press, New York, NY, USA.
79. Q. Fang, J.G., and L. Guibas. . *Locating and bypassing routing holes in sensor networks*. in *Mobile Networks and Applications*. 2006.
80. P. Juang, H.O., Y. Wang, M. Martonosi, D. Rubenstein and L. Peh. *Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet*. in *proceedings of ASPLOS-X*. October 2002.
81. A. Beaufour, M.L.a.P.B. *Smart-tag based data dissemination*. in *WSNA*. Sept. 2002. Atlanta, GA.
82. A. Mainwaring, J.P., R. Szewczyk, D. Culler and J. Anderson. *Wireless sensor networks for habitat monitoring*. in *WSNA*. Sept. 2002. Atlanta, GA.
83. Robert Nowak, U.M. *Boundary Estimation in Sensor Networks: Theory and Methods*. in *proceedings of IPSN*. April 22-23, 2003. Palo Alto, CA, USA.
84. A. Kroller, S.P.F., D. Pfisterer, and S. Fischer. *Deterministic boundary recognition and topology extraction for large sensor networks*. in *proceedings of 17th ACM-SIAM Sympos. Discrete Algorithms*. 2006.
85. S. P. Fekete, M.K., A. Kroller, and N. Lehmann. *A new approach for boundary recognition in geometric sensor networks*. in *17th Canadian Conference on Computational Geometry*. 2005.
86. Andreas Savvides, J.F., and Dimitrios Lymberopoulos. *Using Mobile Sensing Nodes for Dynamic Boundary Estimation*. in *WAMES'04*. June 6, 2004. Massachusetts, US.
87. Zeinalipour-Yazti, D., Andreou, P., Chrysanthis, P. K., and Samaras, G. . *SenseSwarm: a perimeter-based data acquisition framework for mobile sensor networks*. in *4th Workshop on Data Management for Sensor Networks, DMSN '07*. September 24-24, 2007. Vienna, Austria: ACM, New York, NY.
88. Hou, H.Z.a.J., *Maintaining sensing coverage and connectivity in large sensor networks.* Wireless Ad Hoc and Sensor Network 2005. **1**(2): p. 89–123.
89. Tseng, C.F.H.a.Y.C. *The coverage problem in a wireless sensor network*. in *proceedings of 2nd ACM international Workshop on Wireless Sensor Networks and Applications (WSNA '03)*. September 2003. San Diego, CA.
90. M. Raibert, M.G.B., R. R. Playter, . *Boston Dynamics, "BigDog"*. in *proceedings of SPIE Defense & Security Symposium*. 9-13 April, 2007. Orlando, Florida, USA.
91. Nasipuri, A.a.L., K, . *A directionality based location discovery scheme for wireless sensor networks*. in *proceedings of WSNA*. 2002: ACM Press, New York, NY.
92. Eiman Elnahrawy, J.A.-F.a.R.P.M. *Adding Angle of Arrival Modality to Basic RSS Location Management Techniques*. in *proceedings of IEEE International Symposium on Wireless Pervasive Computing (ISWPC'07)*. February, 2007.
93. Kuo, S., Tseng, Y., Wu, F., and Lin, C. *A Probabilistic Signal-Strength-Based Evaluation Methodology for Sensor Network Deployment*. in *19th international Conference on Advanced information Networking and Applications - AINA.* . March 25 - 30, 2005: IEEE Computer Society, Washington, DC
94. J. Ash and L. Potter. *Sensor Network Localization via Received Signal Strength Measurements with Directional Antennas*. in *Forty-Second Annual Allerton Conference on Communication, Control, and Computing*. Sep 2004. Champaign-Urbana, IL.
95. Vaidya., R.C.a.N., *Capture-Aware Protocols for Wireless Multihop Networks Using Multi-Beam Directional Antennas*. March 2005, Technical report, UIUC.
96. Zhou, G., He, T., Krishnamurthy, S., and Stankovic, J. A., *Models and solutions for radio irregularity in wireless sensor networks.* ACM Trans. Sen. Netw., 2006. **2**(2): p. 221-262.

97. Ming Li, D.G., and Prashant Shenoy, . *PRESTO: Feedback-driven Data Management in Sensor Networks”.* in *3rd Symposium on Networked Systems Design & Implementation,* . May 8-10, 2006. San Jose, CA, USA.
98. K.-W. Fan, S.L., and P. Sinha, , *Convergent Anycast: A Low Duty- Cycle MAC Layer for Sensor Networks*. 2005, Technical Report OSU-CISRC-4/05–TR24,.
99. Tassiulas., J.-H.C.a.L. *Energy Conserving Routing in Wireless Ad-hoc Networks*. in *Proceedings of the 19th IEEE INFOCOM*. 2000.
100. K. Kar, M.K., T.V. Lakshman, and L. Tassiulas. . *Routing for Network Capacity Maximization in Energy-constrained Ad-hoc Networks.* . in *Proceedings of the 22rd IEEE INFOCOM*. 2003.
101. Liu., A.S.a.Z. *Maximum Lifetime Routing in Wireless Ad-hoc Networks*. in *Proceedings of the 23rd IEEE INFOCOM*. 2004.
102. L. Li, J.Y.H., P. Bahl, Y.-M. Wang, and R. Wattenhofer. . *Analysis of A Conebased Distributed Topology Control Algorithm for Wireless Multi-hop Networks*. in *Proceedings of the 20th ACM PODC*. 2001.
103. J. Pan, Y.T.H., L. Cai, Y. Shi, and S.X. Shen. . *Topology Control for Wireless Sensor Networks*. in *Proceedings of the 9th ACM MobiCom*. 2003.
104. Hou., N.L.a.J. *Topology Control in Heterogeneous Wireless Networks: Problems and Solutions*. in *Proceedings of the 23rd IEEE INFOCOM*. 2004.
105. W.R. Heinzelman, A.C., and H. Balakrishnan. , *An Application-specific Protocol Architecture for Wireless Microsensor Networks.* . IEEE Transactions on Wireless Communications, 2002. **1**(4): p. 660-670.
106. Kumar., V.K.a.P.R. *Power Control and Clustering in Ad Hoc Networks*. in *Proceedings of the 22nd IEEE INFOCOM*. 2003.
107. Z. Vincze, R.V., and A. Vidács. *Deploying multiple sinks in multi-hop wireless sensor networks*. in *Proceedings of ICPS IEEE International Conference on Pervasive Services,*. 2007. Istanbul, Turkey.
108. Chatzigiannakis, I., Kinalis, A., and Nikoletseas, S. *Sink mobility protocols for data collection in wireless sensor networks*. in *Proceedings of the international Workshop on Mobility Management and Wireless Access, MobiWac '06*. 2006. Terromolinos, Spain: ACM Press, New York, NY,.
109. R. C. Shah, S.R., S. Jain, and W. Brunette. *Data mules: Modeling a three-tier architecture for sparse sensor networks*. in *procedings of IEEE Workshop on Sensor Network Protocols and Applications (SNPA)*. 2003.
110. Krishnamachari, C.A.a.B. *The power of choice in random walks: an empirical study*. in *proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems (MSWiM '06)*. October 2006. New York, NY, USA: ACM press.
111. Volkov, S., *Vertex-reinforced random walk on arbitrary graphs.* journal of Ann. Probab 2001. **29**: p. 66-91.
112. Ren, B., Ma, J., and Chen, C. . *The hybrid mobile wireless sensor networks for data gathering*. in *Proceedings of the 2006 international Conference on Wireless Communications and Mobile Computing, IWCMC '06*. 2006. Vancouver, British Columbia, Canada: ACM Press, New York, NY.
113. Kinalis, A.a.N. *Scalable Data Collection Protocols for Wireless Sensor Networks with Multiple Mobile Sinks*. in *Proceedings of the 40th Annual Simulation Symposium* 2007. Annual Simulation Symposium. IEEE Computer Society, Washington, DC.
114. Shim, G.a.P., D. *Locators of Mobile Sinks forWireless Sensor Networks*. in *Proceedings of the 2006 international Conference Workshops on Parallel Processing* 2006: IEEE Computer Society, Washington, DC.
115. Wu, Y., Zhang, L., Wu, Y., and Niu, Z. *Interest dissemination with directional antennas for wireless sensor networks with mobile sinks*. in *proceedings of the 4th international Conference on Embedded Networked Sensor Systems, SenSys '06*. 2006. Boulder, Colorado, USA, : ACM Press, New York, NY.
116. Aazhang, A.C.a.A.S.a.B. *Using predictable observer mobility for power efficient design of sensor networks*. in *In procedings of second International Workshop on Information Processing in Sensor Networks (IPSN)*. 2003.
117. Hubaux, J.L.a.J.-P. *Joint Mobility and Routing for Lifetime Elongation in Wireless Sensor Networks*. in *procedings of 24th IEEE INFOCOM*. 2005. Miami, USA.
118. Jun Luo, J.P., Michal Piorkowski, Matthias Grossglauser, and Jean-Pierre Hubaux. *Mobiroute: Routing towards a mobile sink for improving lifetime in sensor networks*. in *procedings of IEEE International Conference on Distributed Computing in Sensor Networks (DCOSS)*. 2006: Lecture Notes in Computer Science (LNCS), Springer Verlag,.
119. Yaoyao Gu, D.B., Eylem Ekici, Fusun Ozguner, and Chang-Gun Lee. *Partitioning Based Mobile Element Scheduling in Wireless Sensor Networks*. in *Proceedings of the Second Annual IEEE Communications*

Society Conference on Sensor and Ad Hoc Communications and Networks (IEEE SECON 2005). 2005. Santa Clara, CA, USA: IEEE.

120. Prem Prakash Jayaraman, A.Z., Jerker Delsing. *Sensor data collection using heterogeneous mobile devices*. in *proceedings of IEEE International Conference on Pervasive Services (ICPS'07)*. July 15-20, 2007. Istanbul, Turkey.
121. Bi, Y.a.S., Limin and Ma, Jian and Li, Na and Khan, Imran Ali and Chen, Canfeng *HUMS: An Autonomous Moving Strategy for Mobile Sinks in Data-Gathering Sensor Networks.* EURASIP Journal on Wireless Communications and Networking, 2007. **2007**: p. Article ID 64574, 15 pages.
122. Stefano Basagni, A.C., Emanuel Melachrinoudis, Chiara Petrioli, Z. Maria Wang, *Controlled sink mobility for prolonging wireless sensor networks lifetime* Journal of Wireless Networks, December, 2008. **14**(6): p. 831-858.
123. Shigang Chen, N.Y., *Congestion avoidance based on lightweight buffer management in sensor networks.* IEEE Transactions on Parallel and Distributed Systems, Sept 2006. **17**(9): p. 934-946.
124. Sharaf, M.A., Beaver, J., Labrinidis, A., and Chrysanthis, P. K. *TiNA: a scheme for temporal coherency-aware in-network aggregation*. in *proceedings of the 3rd ACM international Workshop on Data Engineering For Wireless and Mobile Access (MobiDe '03)*. September 19, 2003. San Diego, CA, USA ACM Press, New York, NY.
125. Barton, R.J.Z., R. . *Order-Optimal Data Aggregation in Wireless Sensor Networks Using Cooperative Time-Reversal Communication*. in *40th Annual Conference on Information Sciences and Systems* 22-24 March 2006: Princeton, NJ.
126. Yoon, S.a.S., C. , *The Clustered AGgregation (CAG) technique leveraging spatial and temporal correlations in wireless sensor networks.* ACM Transactions Sensor Networks March 2007. **3**(1): p. 3.
127. Estrin., A.C.a.D., *ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies* IEEE Transactions on Mobile Computing Special Issue on Mission-Oriented Sensor Networks, July- September 2004. **3**(3): p. 272-285.
128. Zhu, J., Hung, K., and Bensaou, B. . *Tradeoff between network lifetime and fair rate allocation in wireless sensor networks with multi-path routing. .* in *proceedings of the 9th ACM international Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM '06)*. October 02 - 06, 2006. Terromolinos, Spain: ACM Press, New York, NY.
129. R Vidhyapriya, a.D.P.T.V., *Energy Efficient Adaptive Multipath Routing for Wireless Sensor Networks.* IAENG International Journal of Computer Science. **34**(1).
130. Deepak Ganesan, R.G., Scott Shenker, Deborah Estrin *Highly-Resilient, Energy-Efficient Multipath Routing in Wireless Sensor Networks.* Mobile Computing and Communications Review, 2002. **5**(2).
131. Wan, C., Eisenman, S. B., and Campbell, A. T. *CODA: congestion detection and avoidance in sensor networks*. in *proceedings of the 1st international Conference on Embedded Networked Sensor Systems (SenSys '03)*. November 05 - 07, 2003. Los Angeles, California, USA ACM Press, New York, NY.
132. Akan, Ö.B.a.A., I. F. , *Event-to-sink reliable transport in wireless sensor networks.* IEEE/ACM Trans. Network, Oct. 2005. **13**(5): p. 1003-1016.
133. Wang, C.L., B. Sohraby, K. Daneshmand, M. Hu, Y., , *Upstream congestion control in wireless sensor networks through cross-layer optimization* IEEE journal on selected areas in communications, May 2007. **25**(4).
134. Sun, K., Peng, P., Ning, P., and Wang, C. . *Secure Distributed Cluster Formation in Wireless Sensor Networks*. in *proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC)* December 11-15, 2006: IEEE Comp Society, Washington DC.
135. Chiasserini, C.a.G., M. , *An Analytical Model for Wireless Sensor Networks with Sleeping Nodes.* IEEE Transactions on Mobile Computing December 2006. **5**(12): p. 1706-1718.
136. W. Ye, J.H., and D. Estrin. *An Energy Efficient MAC Protocol for Wireless Sensor Networks*. in *Proceedings of the IEEE INFOCOM*. June 2002.
137. F. Bennett, D.C., J.B. Evans, A. Hopper, A. Jones, and D. Leask, *Piconet: Embedded Mobile Networking.* IEEE Personal Comm. Magazine, Octobor 1997. **4**(5): p. 8-15.
138. T. van Dam and K. Langendoen. *An Adaptive Energy-Efficient Mac Protocol for Wireless Sensor Networks*. in *Proceedings of First Int'l Conf. Embedded Networked Sensor Systems (SenSys '03)*. 2003.
139. W. Rabiner Heinzelman, A.C., and H. Balakrishnan. *Energy-Efficient Communication Protocol for Wireless Microsensor Networks*. in *Proceedings of 33rd International Conference System Sciences (HICSS '00)*. Janurary 2000.
140. Shi, G., Liao, M., Ma, M., and Shu, Y. *Exploiting sink movement for energy-efficient load-balancing in wireless sensor networks*. in *Proceeding of the 1st ACM international Workshop on Foundations of*

Wireless Ad Hoc and Sensor Networking and Computing, FOWANC '08. May 26 - 26, 2008. Hong Kong, Hong Kong, China ACM, New York, NY.

141. Kumar, P.G.a.P.R., *The capacity of wireless networks* IEEE Trans on Information Theory, 2000. **46**(2).

CURRICULUM VITAE

Majid Iqbal Khan

Research Group Distributed Systems
Department of Distributed and Multimedia Systems
Faculty of Computer Science
University of Vienna

Postal Address: Lenaugasse 2/8, 1080 Vienna, Austria
Phone: +43 1 4277 39605 Fax: +43 1 4277 39649
E-Mail: majid@ani.univie.ac.at

Personal

Born: December 9, 1981 in Rawalpindi, Pakistan
Citizenship: Pakistani

Education

2006-2009: Ph.D. in Computer Science, University of Vienna

2001-2004: Master’s in Software engineering, COMSATS Institute of Information Technology, Islamabad, Pakistan

Professional Experience

09.2004-06.2006: Research associate at COMSATS Institute of Information Technology, Islamabad, Pakistan

Publications

1. Majid I. Khan, Wilfried N. Gansterer, Günter Haring. Energy consumption vs. latency in a new boundary identification method for WSNs with a mobile sink. in proceedings of the 6th ACM international Symposium on Mobility Management and Wireless Access, MobiWac '08. October 30 - 31, 2008. Vancouver, British Columbia, Canada: ACM, New York, NY.
2. Majid I. Khan, Wilfried N. Gansterer, Günter Haring. Identifying the Boundary of a Wireless Sensor Network with a Mobile Sink. in proceedings of the 7th international conference on AD-HOC Networks & Wireless. September 10-12, 2008. Nice, France.
3. Majid I. Khan, Wilfried N. Gansterer, Guenter Haring, Congestion Avoidance and Energy Efficient Routing Protocol for Wireless Sensor Networks with a Mobile Sink. Journal of Networks (JNW), 2007. 2(6): p. 42-49.
4. Majid I. Khan, Wilfried N. Gansterer, Guenter Haring. In-Network Storage Model for Data Persistence under Congestion in Wireless Sensor Networks. in First International Conference on Complex, Intelligent and Software Intensive Systems. 2007. Vienna, Austria.