Graph-Theoretic and Geometric Optimizations in Communication Networks

Can Fang

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2009

To my parents

Acknowledgements

This thesis could not be finished without the help and support of many people who are gratefully acknowledged here.

At the very first, I am honored to express my deepest gratitude to my dedicated supervisor, Dr. Low Chor Ping for helping me so much from a bachelor degree holder to a Ph.D. I believe I am not able to complete this thesis without his continuous support and inspiring encouragement. He has offered me valuable ideas, suggestions and criticisms with his profound knowledge in forensic linguistics and rich research experience. His patience and kindness are greatly appreciated. I have learnt from him a lot not only about dissertation writing, but also the professional ethics. I am very much obliged to his efforts of helping me complete the thesis.

Thanks are also due to my postgraduate friends, who never failed to give me great encouragement and suggestions. I benefited from the collaboration and discussions with the postgraduate students of ICIS InfoComm Research Lab. I am grateful to Xie Feng, Yang Xun, Xue Daojun, Zhang Yan and Wang Ting. I had a great time working with them and learned a lot from the group interactions.

At last but not least, I would like to thank my parents for their support all the way from the very beginning of my postgraduate study. I am thankful to all my family members for their thoughtfulness and encouragement.

Table of Contents

Acknowledgements v

Table of Contents v

Summary ix

List of Figures xi

List of Tables xiii

List of Abbreviations xiv

Chapter 1 Introduction 1
1.1 Motivations 1
1.2 Background and Approaches 5
1.2.1 Optimization and Algorithms 5
1.2.2 Graph Theory and Graph Theoretic Optimizations 7
1.2.3 Geometry and Geometric Optimizations 9
1.3 Research Directions and Contributions 10
1.3.1 Optimal Wavelength Converter Placement with Bounded Wavelength Usage (OPWB) 10
1.3.2 Load-Balanced Clustering Problem (LBCP)in Wireless Sensor Networks 12
1.3.3 The Worst-case and Best-case k-coverage Problems in Wireless Sensor Networks 13
1.3.4 The Generalized Study of Movement-Assisted Sensor Deployment Problem 14
1.4 Thesis Organization 14

Chapter 2 Optimal Wavelength Converter Placement with Bounded Wavelength Usage (OPWB) 16
2.1 Introduction 17
2.2 Related Work 18
2.3 Theoretical Preliminaries 23
2.3.1 Network Model 23
2.3.2 The Computational Intractability of OPWB 24

2.3.3 Graph Decomposition . . . 26
2.4 Networks with Special Topologies . . . 28
2.4.1 Network with Path Topology . . . 28
2.4.2 Network with Star Topology . . . 28
2.4.3 Network with Bridges . . . 32
2.4.4 Network with Tree Topology . . . 35
2.5 Proposed Algorithm . . . 37
2.5.1 Algorithm for OPWB . . . 37
2.5.2 Performance Analysis . . . 38
2.6 Simulation Results . . . 40
2.7 Summary . . . 44

Chapter 3 Load-Balanced Clustering Problem in Wireless Sensor Networks 46
3.1 Introduction . . . 47
3.2 System Architecture . . . 49
3.3 Related Work . . . 49
3.4 Problem Formulation . . . 54
3.5 The Load-Balanced Clustering Problem with Uniform Traffic Load (LBCP-UTL) . . . 55
3.6 The Load-Balanced Clustering Problem (LBCP) . . . 61
3.6.1 The intractability of LBCP . . . 61
3.6.2 Some Observations about LBCP . . . 61
3.6.3 An Approximation Algorithm . . . 64
3.6.4 The Greedy Load-Balanced Clustering Algorithm (GLBCA) . . . 65
3.6.5 Performance Ratio . . . 68
3.7 Simulation Results . . . 72
3.8 Summary . . . 73

Chapter 4 Worst-Case and Best-Case k-coverage in Wireless Sensor Networks . . . 74
4.1 Introduction . . . 75
4.2 Related Work . . . 77
4.2.1 Optimal Covered Path . . . 77
4.2.2 k-coverage . . . 78
4.3 Preliminaries . . . 80
4.3.1 Problem Formulation . . . 80
4.3.2 Growing Disks . . . 82
4.4 Proposed Solution . . . 85
4.4.1 Segments and Borders . . . 85
4.4.2 Worst-Case k-coverage Problem . . . 90
4.4.3 Best-Case k-coverage Problem . . . 96
4.4.4 Performance Analysis . . . 96
4.5 Extensions of Proposed Solution . . . 97
4.5.1 Irregular Coverage Region . . . 97

4.5.2 Distributed Algorithm . . . 101
4.6 Applications . . . 103
4.6.1 Coverage Evaluation . . . 103
4.6.2 Redundancy Analysis . . . 104
4.6.3 Energy Saving . . . 105
4.7 Experimental Results . . . 105
4.8 Summary . . . 109

Chapter 5 A Generalized Study of The Movement-Assisted Sensor Deployment Problem . . . 110
5.1 Introduction . . . 111
5.2 Related Work . . . 112
5.3 Generalized Movement-Assisted Sensor Deployment Problem . . . 114
5.3.1 Preliminaries . . . 115
5.3.2 Problem Formulation . . . 116
5.4 Problem Analysis . . . 117
5.4.1 Attributes for MCRP Scheme . . . 118
5.4.2 Attributes for MFDP Scheme . . . 119
5.5 Proposed Algorithm for MCRP . . . 119
5.5.1 Movement Methodology . . . 119
5.5.2 Virtual Force . . . 120
5.5.3 Attractive Force of Target Field . . . 120
5.5.4 Repulsive Force of Boundary . . . 121
5.5.5 Repulsive Force between Sensors . . . 123
5.5.6 Attractive Force of Potential Field . . . 123
5.5.7 Movement of Sensors . . . 128
5.6 Proposed Algorithm for MFDP . . . 128
5.6.1 Relationship between MCRP and MFDP . . . 128
5.6.2 Proposed Algorithm for MFDP . . . 130
5.7 Optimizations . . . 131
5.7.1 Factor Settings . . . 131
5.7.2 Terminating Condition . . . 132
5.8 Case Study and Simulation Results . . . 132
5.8.1 Performance of algorithm for MCRP . . . 133
5.8.2 Performance of algorithm for MFDP . . . 135
5.9 Summary . . . 139

Chapter 6 Conclusions and Further Work . . . 140
6.1 Conclusions . . . 140
6.2 Recommendations for Further Research . . . 144

Author's Publications . . . 147

Bibliography . . . 149

Summary

With advances in the technology of micro-electronics and communication, some new types of networks have been introduced in recent years. Typical examples of these networks include optical network, Mobile Ad hoc NETwork (MANET) and Wireless Sensor Network (WSN). These brought about many optimization problems which need to be solved to increase the efficiency of the network and to provide better Quality of Service (QoS).

Interestingly but not surprisingly, many of these new optimization issues can be solved efficiently by using some classic mathematic techniques. In this thesis, we address some of the optimization issues that arise in optical networks and sensor networks by using two distinct mathematical approaches, namely the graph-theoretic approach and the geometric approach.

For the graph-theoretic approach, we first address the wavelength converter placement problem in optical networks. We note that existing converter placement schemes do not take into account the availability of resources, such as the number of wavelengths and converters that are available for utilization, in a given network. In this thesis, we take above-mentioned issues into consideration. By applying some graph theoretic results, we propose a wavelength converter placement scheme that is able to provide a flexible trade-off between the number of wavelength converter nodes and the number of wavelengths required. Next, we move on to consider the clustering problem in sensor networks. Particularly, we address the problem of assigning sensor nodes to gateways to form clusters with the objective of minimizing the maximum load of each gateway in the given network. We show that a special case of this problem (whereby the traffic loads contributed by all sensor nodes are the same) is optimally

solvable in polynomial time. We next prove that the general case of this problem is NP-hard. We then propose an efficient $\frac{3}{2}$-approximation algorithm for the problem.

For the geometric approach, two optimization issues in sensor networks are addressed. We first study the k-coverage problem in sensor networks. This is a generalization of the existing work where only k=1 is assumed. By combining geometric techniques with some algorithmic methods, we establish optimal algorithms to solve two variants of this problem, namely the worst-case and the best-case k-coverage problem in polynomial time. Next we study the problem of enhancing sensor coverage by using the mobility of sensors. We adopt the movement-assisted sensor model in this study. We found that most of existing work only consider the case that the target field is a 2-dimensional space. In this thesis, we study a generalized case of this problem whereby the target field can be a space which ranges from 1-dimensional to 3-dimensional. By combining the geometric techniques with the notion of "virtual force", we propose new efficient algorithms for the generalized movement-assisted sensor deployment problem.

List of Figures

2.1 A case that converter is not necessary 23
2.2 Derive a new graph by splitting operation 27
2.3 Constructing the edge compatibility graph for a star network 30
2.4 Two singly connected networks:G_1 and G_2 32
2.5 Reassignment of wavelengths in Scheme 2 34
2.6 Constructing a tree by singly connecting a set of stars and paths 36
2.7 Number of converter nodes in NSFnet 41
2.8 Number of converter nodes in USA long haul network 42
2.9 Number of converter nodes in 4×4 mesh network 42
2.10 Number of converter nodes in 7×7 mesh network 43

3.1 Multi-gateway Clustered Sensor Network 50
3.2 An Optimal Assignment using $\mid M \mid$ Gateways 65
3.3 Further normalization of assignment A 70
3.4 Swapping the assignment of the second sensors in A 71
3.5 A Problem Instance: Performance Bound is Tight 71
3.6 Performance ratio of GLBCA 72

4.1 The definition of k-th distance 81
4.2 A path P with 2-breach larger than r 83
4.3 Segments and their borders 86
4.4 Inner arc and outer arc 87
4.5 Calculating the end points of borders 89
4.6 Exploration process of borders 94
4.7 The shape of coverage region is a sector 98
4.8 Sensing region increases as power increase 98
4.9 A path connecting I and F in the region $\overline{D(S,k,p)}$, where $k = 2$ 100
4.10 Maximal k-breach 106
4.11 Minimum k-support 106
4.12 Comparison of maximal k-breach 108
4.13 Comparison of minimum k-support 108

5.1 Examples of extended boundaries 122
5.2 Attractive force of a 1-D potential field 125
5.3 Attractive force of a 2-D potential field 126

5.4 Attractive force of a 3-D potential field 127
5.5 Coverage ratio and average total moving distance of scenario A 134
5.6 Coverage ratio and average moving distance of scenario B 135
5.7 Coverage ratio and average moving distance of scenario C 136
5.8 Convergence process of scenario B' 138

List of Tables

2.1 A two-step algorithm for OPWB 39
2.2 Statistics of some typical networks 41

3.1 Load-Balanced Clustering Algorithm 58
3.2 Greedy Load-Balanced Clustering Algorithm 67

4.1 The calculation of all borders 91
4.2 The BFS of borders 93
4.3 The binary search procedure 95

5.1 Proposed algorithm for MFDP 131
5.2 Parameter settings of scenarios 133
5.3 Parameter settings and simulation results 137

List of Abbreviations

3hBAC	3-hop Between Adjacent Cluster Heads
BFS	Breadth First Search
CDS	Connected Dominating Set
EGCS	Edge Compatibility Graph Construction Scheme
GLBCA	Greedy Load-Balanced Clustering Algorithm
GPS	Global Positioning System
ILP	Integer Linear Programming
LBCA	Load-Balanced Clustering Algorithm
LBCP	Load-Balanced Clustering Problem
LBCP-UCL	Load-Balanced Clustering Problem with Uniform Traffic Load
LCC	Least Cluster Change
MANET	Mobile Ad Hoc NETwork
MCRP	Maximizing Coverage Ratio Problem
MFDP	Minimizing maximal First Distance Problem
MMSPIM	Minimum Makespan Scheduling Problem on Identical Machines
MMSPUM	Minimum Makespan Scheduling Problem on Unrelated Machines
OPWB	Optimal Wavelength Converter Placement with Bounded Wavelength Usage
PSO	Particle Swarm Optimization
QoS	Quality of Service
RNG	relative neighborhood graph

RWA	Routing and Wavelength Assignment Problem
SNCS	Star Network Construction Scheme
TCS	Tree Construction Scheme
VFA	Virtual Force based sensor deployment Algorithm
WCDS	Weakly-Connected Dominating Set
WDM	Wavelength Division Multiplexing
WSN	Wireless Sensor Network

Chapter 1

Introduction

1.1 Motivations

The usage of communication networks, including computer networks, telephone networks, wired and wireless networks, is growing dramatically in scale and importance in the past decade. For example, it is reported in [1] that the traffic of the Internet doubled every year since 1990's. In addition, various novel networks-based applications such as environment monitoring, home automation and traffic control have been developed in recent years. In general, a communication network is the infrastructure that allows two or more hosts to communicate with each other. The network achieves this by providing a set of rules for communication, called protocols, which should be observed by all participating hosts. To guarantee that the network can be designed, managed and operated in the efficient way, a larger number of optimization problems should be solved in the protocol design of communication networks. These problems include medium access control, congestion control, channel assignment, power management, etc. In this thesis, we focus our research on two types of networks: optical networks and Wireless Sensor Networks (WSNs). We found that they pose many design challenges as they are emerged in recent decades.

Optical networks are high-capacity telecommunications networks based on optical tech-

nologies and components that provide routing, grooming, and restoration at the wavelength level as well as wavelength-based services. Compared to traditional copper-based networks, optical networks have many advantages. Firstly, through Wavelength Division Multiplexing (WDM) [2, 3], an optical fiber network can make high-speed, high-bandwidth tera-bit transmission possible, which is far greater than the bandwidth that can be provided by a copper-based network. Secondly, optical fiber features minimum loss of signal power, which in turn enables it to transmit data, voice, and video at higher speeds and greater distances with minimum usage of repeaters. Thirdly, due to the dielectric properties of optical fiber, it does not conduct electricity nor radiate electromagnetic pulses. Thus the transmission is secure and noise-free. Because of the above-mentioned advantages, optical networks are considered as candidates for the next generation wide-area networks which are required to meet the increasing traffic demand in the foreseeable future.

Ad hoc connectivity is a network connection method which is most often associated with wireless devices. In an ad hoc network, the connection is established for the duration of one session. An ad hoc network requires no base station, devices discover others within range to form a network. Devices may search for target nodes that are out of range by flooding the network with broadcasts that are forwarded by each node [4–6]. Each node in an ad hoc network is willing to forward data for other nodes, and so the determination of which node to forward data is made dynamically based on network connectivity. This is in contrast to wired networks in which routers perform the task of routing. It is also in contrast to managed wireless networks, in which a special node known as an access point manages communication among other nodes.

WSN is a typical type of ad hoc networks [7–9]. A WSN is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations [10, 11]. Sometimes the sensor nodes are equipped with mobility devices so they can move to explore a larger field. The development of wireless

sensor networks was originally motivated by military applications such as battlefield surveillance. However, wireless sensor networks are now used in many civilian application areas, including environment and habitat monitoring, health-care applications, home automation, and traffic control [10,12]. WSNs pose many unique challenges in terms of network protocol design, under the conditions of extremely limited resources, various node capabilities, and dynamic topology. In addition, the importance of distributed optimization algorithms is specially emphasized in WSNs due to the flexibility and scalability of distributed asynchronous algorithms [13].

We note that optical networks and WSNs present some distinctive features listed as follows:

1. *Limited resources.* For most of the networks, bandwidth is one of the most important resources. In wireless networks, the radio bandwidth of certain frequency of wireless mobile networks is scarce as compared with wired networks. Although some networks such as optical networks can provide huge bandwidth, it still cannot satisfy some data transmission requirements since such networks are usually implemented as backbone networks and have to be shared by a large number of users. In the WSNs, power is another important but scarce resource since each each device is powered by battery. Other resources needed by the nodes in networks may include computation capability and storage capacity. Since all these resources are limited, they should be utilized efficiently to provide better Quality of Service (QoS) [14,15].

2. *Consist of some expensive hardware.* The cost of various communication hardware has been reducing in recent years because of the increase of production and the improvement of manufacturing technologies. However, some of the hardware remain to be costly. For example, in the WDM networks, wavelength converters are expensive devices and it has been anticipated that they will continue to be so in the foreseeable future [16]. As another example, consider a WSN which is composed of a set of mobile

robots. Such robots typically cost thousands of US dollars each [17]. Since the budget of network users is limited, these hardware can only be implemented in limited number and they should be utilized in an efficient manner.

3. *Bring new problems.* Many new problems have been generated by the introduction of new types of network structures and applications. In WSNs, the nodes are powered by batteries. A node may fail to communicate with other nodes when it moves out of the communication range or when it runs out of power. Since this node may act as an intermediate node between some other nodes, the disconnection caused by the mobility and failure of this node may lead the disconnection of a part of the network. This problem is referred to as the partition problem [18, 19]. Although the partition problem may also occurs in wired networks, such occurrences are much less likely as compared to wireless networks. In addition, it is no need to address the issue of network partitioning due to node mobility in wired networks. As another example, the wavelength converter is an effective but costly device. Thus the network designers need to determine which nodes should be equipped with wavelength converters so that the network performance can be enhanced most. This problem is referred to as the wavelength converter placement problem [20, 21]. Hence the introduction of optical networks and wireless networks give rise to new optimization issues and challenges that need to be addressed.

Considering the above features of these communication networks, the designers and users of these networks may face many optimization problems which need to be addressed. In this Ph.D research, we concentrate on solving some of these optimization problems by using two approaches: the graph-theoretic approach and the geometric approach. The objective is to design efficient algorithms and schemes which can be effectively applied in optical networks and wireless sensor networks.

1.2 Background and Approaches

1.2.1 Optimization and Algorithms

In mathematics, optimization problems refers to a class of problems which seeks to minimize or maximize an *objective value*. The objective value is determined by systematically choosing the values of real or integer variables from an allowed set, which is referred to as the *search space*.

There are two important subgroups of optimization problems: namely *integer optimization* and *continuous optimization*. We say a problem belongs to the class of integer optimization if the variables in the search space are constrained to take on integer values and a problem belongs to the class of continuous optimization if all variables can be chosen from the set of real numbers.

Integer optimization problems exists widely in the communication networks since many of optimization problems in the communication networks are composed of some integer variables. A typical example is the channel assignment problem, in which each link should select a channel from a set of candidate channels, subject to a set of constraints. Since the set of candidate channels can be considered as a set of integers, the channel assignment problem can be formulated as an integer optimization problem.

Although integer is a subset of real numbers, it does not imply that integer optimization is more tractable than continuous optimization. It can be shown that many of the integer optimization problems that arise in the design of communication networks belong to the class of NP-complete problems. The class of NP-complete problems represents a large collection of such problems, which are all related in the sense that a polynomial time solution of one of them implies the polynomial time solvability of the whole class [22]. Up to now, no polynomial time algorithm for an NP-complete problem is known. Thus it is widely believed that NP-completeness, or more general, NP-hardness of a problem is a certificate of computational intractability. In this thesis, we will focus on solving some of the integer

optimization problems by using graph-theoretic techniques. We will show that some of the problems can be solved optimally in polynomial time while some others belong to the class of NP-hard where only approximate solutions can be found.

On the other hand, many optimization problems arise in the communication networks have real value variables, such as the location coordinates, power levels, distance, etc. These optimization problems are classified as continuous optimization problems. In this thesis, we will concentrate on solving some of these problems by using geometric techniques.

Optimization problems may be solved by using algorithms. An algorithm is a well-defined procedure for solving a problem in a finite number of steps. An algorithm is based on a model that characterized the essential features of the problem. The algorithm specifies a methodology to solve the problem using the model representation. Algorithms are characterized by a number of properties [23–25]. These properties are necessary to ensure that the algorithm correctly solves the problem for which it is intended, or correctly identifies when a solution is impossible to find, in a finite number of steps. These properties include:

- *Specified inputs*: The inputs to an algorithm must be from a pre-specified set.
- *Specified outputs*: For each set of input values, the algorithm must produce outputs from a pre-specified set. The output values produced by the algorithm comprise the solution to the problem.
- *Finiteness*: An algorithm must produce a desired output after a finite number of steps.
- *Effectiveness*: It must be possible to perform each step of the algorithm exactly as specified.
- *Generality*: The algorithm should be applicable to all problems of the desired form. It should not be limited to a particular set of input values or special cases.

As noted earlier, not all of the optimization problems can be solved optimally in polynomial time, such as the NP-complete problems. Thus, especially for problems of practical

interest, one would like to compromise on the quality of the solution for the sake of computing nearly optimal solution quickly. This leads in a natural way to approximation algorithms. An approximation algorithm runs in polynomial time and outputs a solution which performs closely to the optimal solution. The most common way to measure this deviation from optimality is by using performance guarantee, which means the maximum of the difference between the approximate solution and the optimal solution over all instances of a problem. An algorithm A is said to be an α-approximation algorithm, if the ratio of the solution generated by A and an optimal solution, is bounded from above by α. The design of approximation algorithms with good performance guarantees has been a major focus of research in studies related to optimization issues.

1.2.2 Graph Theory and Graph Theoretic Optimizations

Graph theory is the study of "graphs". In this thesis, we will concentrate on solving the integer optimization problems by using the graph theoretic techniques. In mathematics and computer science, the word "graph" denotes the mathematical structure used to model pairwise relations between objects from a certain collection. A "graph" in this context refers to a collection of vertices or "nodes" and a collection of edges that connect pairs of vertices. A graph may be undirected, meaning that there is no distinction between the two vertices associated with each edge, or its edges may be directed from one vertex to another.

Graph theory is an important branch of mathematics and it has a long history. The origin of graph theory can be retrospect to the paper written by Leonhard Euler on the *Seven Bridges of Königsberg* published in 1736. In this paper, Euler proved that it was impossible to find a route through the town of Königsberg(now Kaliningrad) that would cross each of its seven bridges exactly once. Euler found that this result did not depend on the lengths of the bridges, nor on their distance from one another, but only on connectivity properties: which bridges are connected to which islands or riverbanks. Thus this paper is widely regarded as the first paper in the history of graph theory [26].

Many integer optimization problems can be modeled using graphs, and these problems are referred to as graph-theoretic optimization problems. Some of these optimization problems can be solved optimally with polynomial time algorithms, such as the *single source shortest path problem* [25, 26] which can be solved by Dijkstra's algorithm [27] and Bellman-Ford algorithm [28, 29]; some others were still unsolved or have been proven to be NP-complete, such as the *traveling salesman problem* [26, 30].

The most important step and challenge in graph-theoretic optimizations is problem modeling. A proper problem modeling will transforms the original optimization problem into a problem in graph theory. Then some of the graph theoretic results can be applied to solve the problem. As an example, consider the following problem: a host H need to retrieve some data from a server$S$ by using the minimum number of hops. To solve this problem, we can model the network as an un-weighted graph. Then by constructing the Breadth First Search (BFS) tree rooted at the host H, we can find a path connecting the host H and the server S which has the minimum number of hops. In Chapter 2 and Chapter 3, we will focus on some important optimization problems which can be addressed and effectively solved using the graph-theoretic approach. In particular, in Chapter 2 we model the wavelength converter placement problem as an edge-coloring problem. Based on this modeling, we propose a flexible, two-step algorithm which can place a minimal set of wavelength converters in a WDM network with arbitrary topology and the total wavelength usage is bounded. In Chapter 3, we address the problem of assigning sensors to gateways to form clusters so that the maximum load of each gateway is minimized. This problem is referred to as Load-Balanced Clustering Problem (LBCP). We first study a special case of LBCP whereby the offered traffic load from all sensor nodes are the same. We propose a scheme to construct a BFS tree which is rooted on certain sensor. By using this BFS tree, the traffic load of overly loaded gateways can be reassigned to other gateways. This is in turn reduce the maximum traffic load of each gateway and lead to an optimal solution for this special case of LBCP. Next we model the general case of problem as a matching problem in a bipartite graph. By

using some graph-theoretic results from maximal matching, we propose a greedy algorithm and prove its performance bound.

1.2.3 Geometry and Geometric Optimizations

Geometry is another branch of mathematics which is concern with questions of size, shape, and relative position of figures and with properties of space. Geometry is one of the oldest sciences. Its origin is from the practical knowledge concerning lengths, areas, volumes and angles. In the third century B.C., geometry was put into an axiomatic form by Euclid, whose treatment - Euclidean geometry - set a standard for many centuries to follow. One of the early applications of geometry is the field of astronomy, especially mapping the positions of the stars and planets on the celestial sphere.

Modern geometry is much more developed and enriched. Branches of geometry include algebraic geometry, non-Euclidean geometry, information geometry, finite geometry and numerical geometry. The advance of computer science in recent decades acts as an impetus for the development of modern geometry. Computational geometry, a branch of geometry, is the result of interdisciplinary study between geometry and computer science.

In general, geometry is mainly concerned with the measurable properties of point, line, plane and other figures in the space, such properties include size, shape and position. As a comparison, in graph theory, we study the abstract "graph" which is composed of vertices and edges. In this graph, the size, shape and other measurable properties of vertices and edges will not affect the results obtained.

Geometric optimization deals with problems of computing geometric objects which are optimal subject to certain criteria and constraints. In many applications, the objective function of the underlying optimization problem involves a small number of variables, and the constraints are induced by a family of geometric objects. These problems are referred to as geometric-optimization problems. In such cases one expects that the underlying geometry can be exploited to obtain faster and simpler algorithms. Similar to graph theoretic

optimizations, the most important step of geometric optimization is the problem modeling. A proper modeling can explore the relationship between the original problem and some geometric problems, which may lead to an efficient solution for the problem.

As we will show in Chapter 4 and Chapter 5, some optimization problems arisen in sensor networks can be solved effectively by using geometric techniques since these problems can be transformed into problems which deal with measurable or numerical variables such as locations, distances, shapes and movements. In Chapter 4, we apply a geometric technique which is referred to as *disk diagram* to calculate the worst-case and the best-case k-coverage path in sensor networks. In Chapter 5, we use geometric techniques to detect the region which is not covered or is redundantly covered and move sensors accordingly to enhance the coverage of the sensor networks.

1.3 Research Directions and Contributions

In this thesis, we aim at developing effective and efficient algorithms for some optimization problems that arise in communication networks by using two distinct approaches. The wavelength converter placement problem in the WDMs and the clustering problem in WSNs are addressed by using graph-theoretic approach. In another approach, namely the geometric approach, we address the k-coverage problem and movement-assisted sensor deployment problem in WSNs.

1.3.1 Optimal Wavelength Converter Placement with Bounded Wavelength Usage (OPWB)

Wavelength is one of the most important resources in WDM networks. In optical routing, we are given a set of communication paths (or ligthpaths) in a WDM network and we must assign a wavelength to each path so that paths sharing a link must be assigned with different wavelengths. With the introduction of wavelength converters, the number of wavelengths

which is required to support all lights paths can be much reduced. However, wavelength converters are scarce resources as they are expensive devices [16]. In this thesis, we aim to design a scheme that is able to provide a flexible trade-off between the number of wavelength converters and the number of wavelengths required to support the communications of all lightpaths in a given network. In particular, we study the problem of placing the minimum number of wavelength converters in a network to ensure that the number of wavelengths needed will not exceed a given bound αL, where L is the maximum link load in the network and α is a parameter defined by the network designer to reflect the overall availability of wavelength resources. Although similar problems have been addressed by some existing works [31–34], the schemes proposed by these works only provide fixed upper bounds for the wavelength usage. Thus they are not able to adapt to the availability of resources of different networks. Further, we take the set of lightpaths into consideration as a design input. We note that doing so will help to reduce the redundant deployment of wavelength converters in a given network.

We adopt the basic idea of decomposing a given network modeled using a graph into some sub-networks which have special topologies. We find that this graph can be constructed by properly connecting some sub-graphs which has tree topology. These trees can be further partitioned into sub-graphs with star and path topology. For the sub-graphs with path topology, it is proven in [32] that the number of wavelengths required to support all lightpaths is bounded by the maximum link load in the network, denoted by L. For sub-graphs with star topology, we find that the problem of assigning wavelengths to the lightpaths can be transformed into the problem of edge coloring an arbitrary graph by adopting the schemes we proposed. The edge coloring problem is known to be NP-hard and existing results which have been proposed in the literatures can be applied to obtain a bound for the number of wavelengths required to support all lightpaths in a network with star topology.

Based on these observations and analysis, we propose a flexible, two-step algorithm which can place a minimal set of wavelength converters in a WDM network with arbitrary topology

and the total wavelength usage is bounded. Its correctness is verified by using theoretical analysis and its effectiveness is evaluated by both theoretical and experimental studies.

1.3.2 Load-Balanced Clustering Problem (LBCP)in Wireless Sensor Networks

In the cluster-based WSNs, sensor nodes are grouped into distinct clusters by using the corresponding gateway as the cluster-head. Each sensor node only belongs to one cluster and communicates with the command node through the gateway (or cluster-head) in the cluster. We note that if the sensor nodes and the gateways are not "well distributed", some gateways may be overloaded with increase in sensor density and detected phenomenons/targets. Such overload may increase latency in communication and cause degradation of overall network performance. Being aware of this issue, we address the problem of assigning sensor nodes to gateways to form clusters with the objective of minimizing the maximum load of each gateway in the given network. The effort to minimize that maximum load of each gateway will result in a more balanced distribution of loads among the set of gateways. We refer problem as the Load-Balanced Clustering Problem (LBCP).

We first consider a special case of LBCP whereby the offered traffic loads from all sensor nodes are the same. We refer to this problem as the Load-Balanced Clustering Problem with Uniform Traffic Load (LBCP-UTL). We first show that by using the Breadth First Search (BFS) tree which is constructed by the scheme we proposed, the traffic load of a gateway can be reassigned to another gateway which has less traffic load. This in turn will help to distribute the traffic load among the gateways. We then propose a polynomial time algorithm which solves LBCP-UCL optimally.

For the general case of LBCP, we study its computational tractability and prove this problem is NP-hard, which indicates that it is highly unlikely to solve the problem optimally in polynomial time. Thus we propose an approximation algorithm for the general case of

LBCP and prove that our proposed algorithm is able to guarantee a performance ratio of $\frac{3}{2}$. Empirical studies shown that our proposed algorithm is able to perform even better on the average as compared to the worst-case performance ratio derived.

1.3.3 The Worst-case and Best-case k-coverage Problems in Wireless Sensor Networks

Coverage is an essential problem in sensor networks. Sensor coverage, which reflects how well a sensor network is monitored by sensors, is an important measure for the QoS that a sensor network can provide. In this thesis, we address the coverage problem from two different view points and refer to them as the worst-case and best-case coverage problems. Existing work on these two problems assumed that the coverage degree is one (i.e. the target area falls within the sensing range of at least one sensor), and is referred to as the 1-coverage problem. By noticing that 1-degree coverage may not be sufficient in some scenarios, we address the k-coverage problem, where the coverage degree is a user-defined parameter k. This is a generalization of the earlier work where only k=1 is assumed.

By adopting the basic idea of a geometric technique called growing disks and adopting a series of transformations, we are able to reformulate both cases of the *k-coverage problem* into the problem of finding a sequence of adjacent borders. We then propose optimal polynomial time algorithms to solve both variants of the problem.

Two important extensions of the study on the problem are also addressed. Firstly, we address the problem by removing the assumption that the coverage region of each sensor is a unified disk and propose an approach to solve the problems under such scenario. Secondly, we discuss how our proposed algorithms can be adapted to solve the problems in a distributed manner. Both extensions help in applying our algorithms in more practical scenarios.

1.3.4 The Generalized Study of Movement-Assisted Sensor Deployment Problem

In mobile sensor networks, sensors are capable to move within a certain distance. This mobility can be used to enhance the sensor coverage of the target field. This technique is referred to as the movement-assisted sensor deployment. Existing movement-assisted sensor deployment schemes usually consider the problem of deploying a set of sensors in a 2-*dimensional* (2-D) field. Further, they assume that the field in which the sensors are initially placed is identical to the target field. However, we find these assumptions may not be valid under some scenarios. Thus we addressed this problem in a more general way. The initial deployment field and target field can be a 1-D, 2-D or 3-D space. Further, the initial deployment field and the target field are not required to be identical to each other.

By noticing that the sensor deployment schemes for 1-D, 2-D and 3-D cases share some common attributes, we study the general problem of movement-assisted sensor deployment in the context of achieving good coverage. By using two distinct objectives, the problem is formulated in two variations, namely Maximizing Coverage Ratio Problem (MCRP) and Minimizing Maximal First Distance Problem (MFDP). For both variations of the problem, we identify a set of basic attributes which can be used as guidelines for designing efficient movement-assisted sensor deployment schemes. Based on these attributes and combined with the basic idea of "virtual force", we first propose an efficient algorithm for MCRP. Next, we analyze the relationship between these two optimization objectives and propose an algorithm for MFDP based on the solution of MCRP. Experimental study shows that our algorithms for both variants of the problem work efficiently in enhancing the sensor coverage.

1.4 Thesis Organization

The structure of this thesis is as follows. In Chapter 2, we address the wavelength converter placement problem and propose an algorithm which can make a flexible tradeoff between

the number of converters and the number of wavelengths required. The problem of load-balanced clustering is address in Chapter 3, an optimal algorithm for a special case of the problem and a greedy algorithm with performance ratio of $\frac{3}{2}$ are proposed in this chapter. In Chapter 4, we address the k-coverage problem of WSN in two cases, namely the worst-case and the best-case. Algorithms for both variants of the problem are proposed. A unified framework which captures some basic attributes of movement-assisted sensor deployment schemes is proposed in Chapter 5, and algorithms which are based on this framework are also presented in Chapter 5. We present our conclusions and discuss the future research directions in Chapter 6.

Chapter 2

Optimal Wavelength Converter Placement with Bounded Wavelength Usage (OPWB)

Wavelength is one of the most important resources in Wavelength Division Multiplexing (WDM) networks. In optical routing, we are given a set of communication paths (or ligth-paths) in a WDM network and we must assign a wavelength to each path so that paths sharing a link must be assigned with different wavelengths. By properly choosing a set of nodes that are equipped with wavelength converters, the number of wavelengths which is required to support all lightpaths can be much reduced. In this chapter, we study the problem of placing the minimum number of wavelength converters in a network to ensure that the number of wavelengths needed will not exceed a given bound αL, where L is the maximum link load in the network and α is a parameter defined by the network designer to reflect the overall availability of wavelength resources. We show that this problem is closely related to the edge coloring problem which is proved to be NP-hard. Based on some theorems in graph theory, we develop an efficient heuristic algorithm for the problem and extensive theoretical analysis and experimental studies are carried out to verify the effectiveness and performance of the algorithm.

2.1 Introduction

In Wavelength Division Multiplexing (WDM) networks [2, 3], the bandwidth of an optical fibre is divided into multiple wavelength channels so that multiple users can transmit data at distinct wavelengths through the same fibre concurrently. Since all-optical WDM networks can provide communication service with huge bandwidth and low latency, such networks are considered as candidates for the next generation wide-area networks which are required to meet the increasing traffic demand in the foreseeable future.

A *lightpath* is an optical communication path between a pair of source and destination nodes which may span multiple hops. In WDM networks, any pair of lightpaths (traffic demand) must be assigned with different wavelengths if they share the same link in any hops. Hence it is easy to see that the number of wavelengths required in a network is at least equal to the *natural congestion bound* or *maximum link load*, defined to be the maximum number of paths passing through any link in the network.

Wavelength converter is an essential device in multi-hop WDMs that enhances the scalability of the network. In WDMs without any wavelength conversion, the same wavelength must be assigned to all links in a lightpath (this is often referred to as the *wavelength continuity constraint*). If a node is equipped with a wavelength converter, any lightpath that passes through this node may change its wavelength. Clearly wavelength assignments in networks with wavelength converters can be more efficient (uses less wavelengths) than wavelength assignments for the same set of paths where no wavelength converter is available. However, wavelength converters are expensive devices and it has been anticipated that they will continue to be so in the foreseeable future [16]. In addition, densely placed converters may cause the signal distortion [35]. Hence, it is not practical to equip every node with a wavelength converter.

Several wavelength converter placement schemes [31–33] have been proposed in the literature to reduce the overall wavelength requirements of a given network by employing a

minimal set of converters nodes. However, we note that existing converter placement schemes do not take into account the availability of resources, such as the number of wavelengths and converters that are available for utilization, in a given network; hence they are not able to adapt to the availability of resources of different networks.

In this chapter, we aim to take above-mentioned issues into consideration in the design of efficient wavelength converter placement schemes for WDM networks. Furthermore, we aim to design a scheme that is able to provide a flexible trade-off between the number of wavelength converter nodes and the number of wavelengths required to support the communications of all lightpaths in a given network. In particular, the problem that we interested in is as follows: given the traffic demand (a set of lightpaths) in a network, determine the placement of the minimum number of wavelength converters in the network so that the number of required wavelengths does not exceed a given upper bound αL, where L is the maximum link load in the network and α is a parameter that can be defined by the network designer to reflect the overall availability of wavelength resources.

The rest of this chapter is organized as follows: Section 2.2 reviews the related work. Section 2.3 presents the problem assumptions, problem formulation and the methodology used in this work. Section 2.4 addresses the problem of determining the wavelength requirements for the networks with special topologies. The results we obtained in Section 2.4 are applied in Section 2.5 and a two-step algorithm is proposed and analyzed. Results from our simulation studies are discussed in Section 2.6. Section 2.7 concludes this chapter.

2.2 Related Work

The introduction of wavelength converter brings huge impact on wavelength routing. Being aware of this, many studies concentrate on increasing the throughput of the network by using a limited number of wavelength converters. In [36], Yang *et al.* considered the problem of designing WDM optical interconnects which can provide maximum connectivity

while keeping a minimum hardware cost. A WDM optical interconnects structure which achieve full connectivity by employing the low-cost limited range wavelength converters was proposed. In [37], Ngo *et al.* studied the problem of determining the minimum number of limited-range wavelength converters needed to construct a strictly, wide-sense, and rearrangeably nonblocking optical cross-connects for both unicast and multicast traffic patterns. The minimum number of limited-wavelength converter needed for wide-sense and rearrangeably nonblocking unicast and multicast optical cross-connects was given in the exact formula. One of the structures to achieve such a minimum number has also been proposed by Ngo *et al.* In [20], Venugopal *et al.* investigated the problem of placing limited range wavelength converters in arbitrary mesh optical networks to minimize the blocking probabilities and to reduce the distortion of optical signal. An efficient heuristics which place limited converters at a few nodes was proposed. Observation showed that placing converters by using proposed heuristics can provide almost the entire improvement in blocking probability as the full range wavelength converter placed at all the nodes. Given the number of nodes in the network which is equipped with full wavelength converter, Thiagarajan *et al.* [21] provide the formulation of overall network blocking probability. Based on this formulation, an optimal converter placement algorithm was proposed to minimize the blocking probability by employing a given number of converters.

In general, the number of wavelength required to support all lightpaths can be determined by solving the Routing and Wavelength Assignment Problem (RWA). This problem, however, has been proved to be NP-complete [38]. Chu and Li did a series work [39–41] on the joint study of RWA and wavelength converter placement problem. In [39], Chu *et al.* demostrated that RWA and the wavelength converter placement problem are closely related. A well-designed wavelength converter placement mechanism for a particular RWA algorithm might not work well with a different RWA algorithm. Therefore, the wavelength converter placement and the RWA have to be considered jointly. Based on this observation, Chu *et al.* proposd the heuristics for the fixed-alternate routing algorithm and the least-loaded routing

algorithm, which can achieve almost the same performance compared with full wavelength conversion in terms of blocking probability. The benefit of this joint study is further verified by [40], which showed that the blocking probablity can be much reduced by adopting the proposed algorithms for RWA and wavelength converter placement problem. In [41], Chu *et al.* further demonstrated that the existing dynamic RWA algorithms do not work well in the presence of wavelength conversion. Chu *et al.* then propose a dynamic RWA algorithm which considers both the current traffic load and the route lengths jointly. Simulation results confirmed that the proposed algorithm can achieve a lower blocking probablity and can make a good tradeoff between the route length and the link utilization ratio.

Given a network and a set of lightpaths, wavelength converters can also be used to reduce the number of wavelength requited to support all lightpaths. In [42], Baroni *et al.* studied the relationship between wavelength requirement and network topology. In particular, they evaluated the number of required wavelengths as a function of the physical connectivity, which is a parameter that can be calculated from the network topology.

Some studies focus on determining the wavelength requirements of the networks with special topologies. These studies usually consider two types of communication channels: *duplex* and *unidirectional*. In duplex channels, data can be transmitted in both directions in one fiber; in unidirectional channels, data are transmitted in one direction from the source to the destination. Wilfong *et al.* [34] proved that $2L-1$ wavelengths are sufficient and necessary for a ring network with unidirectional channels, where L is the maximum link load in the network. Here "*sufficient*" means the routing and wavelength assignment can be realized with at most $2L-1$ wavelengths, without wavelength conversion and "*necessary*" means that for any integer L there is an instance in which routing and wavelength assignment requires $2L-1$ wavelengths. For the network with tree topology, Erlebach *et al.* [43] proved that $\frac{5}{3}L$ wavelengths are sufficient and $\frac{5}{4}L$ wavelengths are necessary in unidirectional channels; Raghavan *et al.* [44] proved that $\frac{3}{2}L$ wavelengths are sufficient and necessary in duplex channels.

In [34], Wilfong *et al.* defined a set S of nodes in a network to be *sufficient* if placing converters at the nodes in S can ensure that the number of wavelengths required by any set of lightpaths is equal to its maximum linkload L. They showed that the problem of finding a sufficient set of minimum size for an arbitrary WDM network with unidirectional channels, referred to as the *minimum sufficient set problem*, is NP-complete. In addition, they showed that for networks with unidirectional channels, (i) the empty set (i.e. $S = \emptyset$) is sufficient if and only if the topology of network is a spider (i.e. a tree with at most one vertex of degree greater than two) and (ii) for any ring networks, the size of a minimum sufficient set S is equal to 1.

In [31], Kleinberg *et al.* extended the splitting technique to arbitrary directed graphs and proposed a 2-approximation algorithm for finding the minimum sufficient set for an arbitrary directed (unidirectional channel) network. They also showed that the minimum sufficient set problem is as hard as the minimum vertex cover problem, which is believed to be unlikely to have an approximation algorithm with performance ratio less than 2 [45].

The problem of finding a minimum sufficient set for a network with tree or tree of rings topology has been investigated by Wan *et al.* [46]. Wan *et al.* found in networks with these two special topologies, the minimum sufficient set can be determined optimally in polynomial time.

Erlebach *el.al* [47] considered the case in which all lightpaths will be routed by the shortest path algorithm and they gave the complete characterization of duplex networks for which $S = \emptyset$ holds. They also noted that the restriction to shortest-path routing could reduce the converter requirements of a given network significantly.

The problem of finding a minimum sufficient set for an arbitrary network has also been addressed by Jia *et al.* [32]. In particular, Jia *el.al* [32] considered the problem of placing a minimum number of wavelength converters in a network such that the wavelength needed by the network does not exceed the maximum link load L. They refer to this feature of wavelength converter placement and wavelength assignment as L-assignability. Jia *et al.*

proved that the problem of finding a minimum sufficient set for a network with duplex channel is optimally solvable in polynomial time. In addition, they proved that the problem of finding minimum sufficient set for a network with unidirectional channel is NP-complete (also proven by Kleinberg *et al.* [31])and they proposed a 2-approximation algorithm for this problem.

By noticing that achieving L-assignability usually requires a large number of converters in some typical network topologies, another work of Jia *et al.* [33] aims at striking a trade-off between the number of wavelengths and the number of wavelength converters. In [33], Jia *et al.* introduced the notion of αL-assignability, which means that the number of required wavelengths will not exceed the maximum linkload by a factor of α. They showed that the problem of placing a minimum set of converters to achieve αL-assignability is NP-complete when α is fixed at $\frac{3}{2}$ (for duplex channel) and $\frac{5}{3}$(for unidirectional channel). Jia *et al.* also proposed a 2-approximation for both duplex and unidirectional cases.

Our work differ from the works in [31–34] in two aspects:

1. The given set of lightpaths are taken into consideration as a design input. We note that doing so will help to reduce the redundant deployment of wavelength converters in a given network. For example, consider the scheme proposed in [32] (for duplex channels) whereby converters are placed at each node whose degree is larger than two. In applying this scheme for a star network, a converter will be placed at the central node. Let's consider a case whereby there are only two lightpaths that pass through this network (as shown in Figure 2.1). It is easy to see that it is not necessary to place any converter in this case. Thus a redundant wavelength converter will be placed if the scheme proposed in [32] is adopted.

2. The schemes proposed by existing work only provide fixed upper bounds for the wavelength usage. Although Jia *et al.* introduced the notion of αL-assignability, where $1 < \alpha < 2$, they only address the case where $\alpha = \frac{3}{2}$ (for duplex channels) [33]. By

noticing that there is a gap between L and $\frac{3}{2}L$, we adopt a variable upper bound αL, whereby α is a variable which ranges between 1 to $\frac{3}{2}$. By considering the availability of converters and wavelengths, we can tighten this bound(use smaller α) if the wavelength is scarce and converter is comparably abundant and vice versa. It is easy to see that this feature provides additional flexibility for the network designers in the overall network design process.

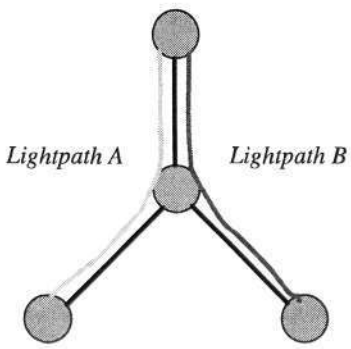


Figure 2.1: A case that converter is not necessary

2.3 Theoretical Preliminaries

2.3.1 Network Model

We model the network as an undirected simple graph G(V, E), where V is the vertex set and E is the edge set. The traffic demand is represented by a set of lightpaths $D = \{l_1, l_2, l_3 \ldots l_k\}$. In this chapter, we consider the case of static routing where all connections (lightpaths) are known in advance and stay for an infinite period of time in the network. The number of wavelengths needed to support all lightpaths in D is denoted by $W(G, D)$.

We assume that all communications support *duplex communication channels*, whereby data can transmit in both directions in the same fibre. The set of lightpaths that occupy the same link must be assigned with different wavelengths on this link regardless of their transmitting direction.

In this chapter, we assume all converters have *full conversion capability* [48, 49]. This means the converter can translate an incoming wavelength into any outgoing wavelength. We adopt the *shared by node model* [49] which allows converters placed at a node to be shared by any lightpaths that pass through this node. In addition, we assume the capacity of each converters is large enough to support all lightpath pass though it.

Next we formally define the problem addressed in this chapter as follows: given the network G and a set of traffic demand $D = \{l_1, l_2, l_3 \ldots l_k\}$, locate a minimum set of nodes $S \subseteq V$ so that if we place wavelength converters at each node in S, the number of required wavelengths will not exceed the given bound αL, where L is the maximum linkload in the network and α is a parameter that can be defined by the network designer in the range of $[1, \frac{3}{2}]$. We refer this problem as *Optimal Wavelength Converter Placement with Bounded Wavelength Usage Problem*(OPWB).

2.3.2 The Computational Intractability of OPWB

We formulate the corresponding decision problem of OPWB, which is referred as OPWB', as follows:

Instance: A network $G(V, E)$, a lightpath set D, non-negative integer i_1 and positive real number r_1, where $i_1 = |S|$, $r_1 = \alpha L$.

Question: Is it possible to place i_1 converters in G so that the number of wavelengths required to support all lightpaths in D will not exceed r_1?

Now we consider a special case where $i_1 = 0$ and G is a *star*. A *star* $G_{star}(V, E)$ is a graph whereby each vertex in G_{star} is of degree one except for one vertex whose degree is at least three. We note this special case is equivalent to the wavelength assignment problem in a star network:

Instance: A star network $G_{star}(V, E)$, a lightpath set D, positive real number r_1, where $r_1 = \alpha L$.

Question: When no converter is placed, is it possible establish all lightpaths in D so

that the number of required wavelengths will not exceed r_1?

Lemma 2.1 The wavelength assignment problem in a star network is NP-hard.

Proof: We found the wavelength assignment problem in a star network is closely related to the *edge coloring problem* which is defined as follows:

Definition 2.1 Let G be a graph without loops. A $k-$*edge coloring* of G is an assignment of k colors to the edges of G in such a way that any two edges meeting at a common vertex are assigned with different colors. If G has a k-edge coloring, then G is said to be $k-$*edge colorable*. The chromatic index of G, denoted by $\chi'(G)$, is the smallest value of k for which G is k-edge colorable. The problem of finding a k-edge coloring of G whereby $k = \chi'(G)$ is called the *edge coloring problem*.

We will show that an edge coloring problem can be transformed into a wavelength assignment problem of a star network. This transformation can be done by the scheme described below:

Star Network Construction Scheme(SNCS)

Input: A graph $G(V, E)$, $V = \{v_1, v_2, \ldots v_n\}$, $E = \{e_1, e_2, \ldots e_m\}$.

Output: A star network $G_{star}(V^* \cup v_c, E^*)$, $V^* = \{v_1^*, v_2^*, \ldots v_n^*\}$, $E^* = \{e_1^*, e_2^*, \ldots e_n^*\}$, lightpath set $D = \{l_1, l_2, l_3 \ldots l_m\}$.

1. $V^* = \emptyset$; $E^* = \emptyset$; $D^* = \emptyset$;
2. Create a vertex v_c as the center of the star network;
3. For each vertex $v_i \in V$, create a vertex $v_i^* \in V^*$ and insert an edge $e_i^* \in E^*$ by connecting v_c and v_i^*;
4. For each edge $e_i \in E$, where $e_i = (v_a, v_b), v_a \in V, v_b \in V$, we create a 2-hop lightpath $l_i \in D$ which traversing over edge $e_a^* \in E^*, e_b^* \in E^*$ that correspond to $v_a \in V, v_b \in V$.

It is easy to see that the star network G_{star} constructed by the scheme described above satisfies the following properties:

- Each vertex $e_i^* \in E^*$ in G_{star} corresponds to a vertex $v_i \in V$ in G;
- Each lightpath $l_i \in D$ in G_{star} corresponds to an edge $e_i \in E$ in G;

- Any two lightpaths in G_{star} will share an edge(link) $e_i^* \in E^*$ if and only if their corresponding edges are adjacent to the same vertex $v_i \in V$ in G.

The task of edge coloring is to assign colors to all edges so that any pair of edges which are adjacent to the same vertex will be assigned with different colors; On the other hand, the task of wavelength assignment is to assign wavelengths to all lightpaths so that any pair of lightpaths which occupying the same link will be assigned with different wavelengths. It's easy to see the edge coloring problem in G is equivalent to the wavelength assignment problem in G_{star} when $r_1 = \chi'(G)$, i.e. the edge coloring problem of an arbitrary graph G is reducible to the wavelength assignment problem by SNCS, which can be done in polynomial time. Since the edge coloring problem is known to be NP-hard [22, 50], the wavelength assignment problem in a star network is also NP-hard.

■

Theorem 2.1 OPWB is NP-hard

Proof: As mentioned above, the wavelength assignment problem in a star network is a special case of OPWB'. We have proven in Lemma 1 that the wavelength assignment problem in a star network is NP-hard, hence OPWB' and OPWB are also NP-hard. ■

2.3.3 Graph Decomposition

Consider the case whereby node v_i is equipped with wavelength converters. All lightpaths that pass though v_i can convert their wavelength at v_i. The set of lightpaths which shared these converters are thus split into two parts, one from source node to the converter node v_i while another one from v_i to the destination node. The wavelength assignments for these two parts are independent from each other; thus placing a set of wavelength converters at a set of nodes S will result in the splitting of lightpaths that pass through the nodes in S into shorter lightpaths. This feature can be described by the splitting operation which is defined as follows:

Given a graph $G(V, E)$ and subset $S \subseteq V$, let $G_S(V', E')$ be a new graph derived from G by splitting each node $x \in S$ into $deg(x)$ nodes in V', where $deg(x)$ denote the degree of node x in G. Each edge (x, y) in $G(V, E)$ becomes edge (x^*, y) in $G_S(V', E')$, where x^* is a new node that is generated by splitting x in $G(V, E)$. Let $W_x \subseteq V'$ denote the set of vertices in G_S which are derived from node x in G. Note that each node in W_x is of degree one. The process of decomposing node x in G into a new set of nodes W_x in G_S is referred to as the *splitting operation*(as in [31–34]).

Figure 2.2 illustrates the decomposition of a given graph G into a new graph G_S by splitting nodes in the set S, where $S = \{3, 4\}$. Given a graph $G(V, E)$ and a set $S \subseteq V$, the process of decomposing a graph $G(V, E)$ by splitting the nodes in S can be expressed as follows: $G_S(V', E) = split[G(V, E), S]$

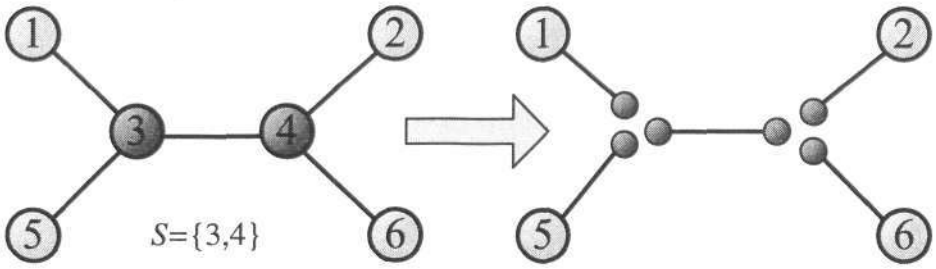


Figure 2.2: Derive a new graph by splitting operation

Since the task of wavelength assignment in networks with special topologies, such as paths, stars and trees, can be done more easily than in network with arbitrary topologies, we adopt the approach of decomposing a given network into edge-disjoint subgraphs with special topologies which include paths, stars and trees. The decomposition process is carried out by using the splitting operation described above. We note that such an approach has also been used in [31–34]. However the objectives of our approach differ from those in [31–34] as follows: the objectives of the work in [31–34] are to select a set of converters for placement to satisfy L-assignability or $\frac{3}{2}L$-assignability, i.e. fixed bounds on wavelength usage; on the other hand, the objectives of our approach is to place a minimal set of wavelength converters to satisfy αL-assignability, where α is a parameter that may be specified by the user. Hence

the problem addressed in this work is a generalization of those addressed in [31–34].

2.4 Networks with Special Topologies

2.4.1 Network with Path Topology

Theorem 2.2 [32] Given a network with path topology (which is often referred to as *linear network*), denoted by G_{path}, then $W(G_{path}, D) = L$ holds for arbitrary D, where L is the maximum linkload of G_{path}.

Theorem 2.3 [32] Given a network G, if every connected component of G is a path, then $W(G_{path}, D) = L$ holds for arbitrary D, where L is the maximum linkload of G.

It follows from Theorem 2.3 that if we split an arbitrary network into a set of linear networks, then L-assignability can always be achieved for the network. However, a major drawback of this approach is that a large number of nodes will have to be split in the process, thus resulting in high usage of wavelength converters.

2.4.2 Network with Star Topology

As described in Section 2.3.2, a *star* $G_{star}(V, E)$ is a graph whereby each vertex in G_{star} is of degree one except for one vertex whose degree is at least three. The vertex whose degree is three or above is referred to as the *center node* and all edges that adjacent to this vertex are called *legs*. We note that each lightpath in $G_{star}(V, E)$ has at most 2 hops. We will show that the wavelength assignment problem for a network with star topology can be transformed into an edge coloring problem.

In Section 2.3.2, we have shown that the edge coloring problem can be transformed to the wavelength assignment problem of a star network by employing the SNCS. In this section we will show that complement is also true, i.e. an wavelength assignment problem of a star network can be transformed into a edge coloring problem. Given a star network $G_{star}(V, E)$,

we can construct a new graph $H^*(V^*, E^*)$ which we refer to as the *edge compatibility graph*, as follows:

Edge Compatibility Graph Construction Scheme (EGCS)

Input: A star network $G_{star}(V, E)$, $V = \{v_1, v_2, \ldots v_n\}$, $E = \{e_1, e_2, \ldots e_m\}$ and traffic demand $D = \{l_1, l_2, l_3 \ldots l_k\}$.

Output: Edge compatibility graph $H^*(V^* \cup W^*, E^*)$.

1. $V^* = \emptyset; W^* = \emptyset; E^* = \emptyset$;
2. For each edge $e_i \in E$, create a vertex $v_i^* \in V^*$(shown as Figure 2.3-i);
3. For each lightpath $l_i \in D$, we do the following:

Case(i): l_i is a 2-hop lightpath.

In this case, l_i will occupy two edges, say e_x and e_y in G_{star}. Insert an edge $e_i^* \in E^*$ in H^* that connects the two vertices v_x^* and v_y^* in H^* that correspond to the edges e_x and e_y(shown as Figure 2.3-ii).

Case(ii): l_i is a 1-hop lightpath.

In this case, l_i will occupy an edge, say e_x in G_{star}. Insert a new vertex $w_i^* \in W^*$ in H^* and insert an edge $e_i^* \in E^*$ in H^* that will connect the pair of vertices $v_x^* \in V^*$ and $w_i^* \in W^*$in H^*(shown as Figure 2.3-iii).

Based on the construction scheme described above, it is easy to see that the *edge compatibility graph* H^* of a star network G_{star} satisfies the following properties:

- Each vertex $v_i^* \in V^*$ in H^* corresponds to an edge $e_i \in E$ in G_{star};
- Each edge $e_i^* \in E^*$ in H^* corresponds to a lightpath $l_i \in D$ in G_{star};
- Any two edges in H^* are adjacent if and only if their corresponding lightpaths occupy the same edge in G_{star}.

Next we note that if G_{star} is a part of a larger network G, i.e. G_{star} is a subgraph of G, then there may exist more than one lightpaths traversing through the same pair of links in G_{star}. This in turn implies that there may exist more than one edges connecting the same

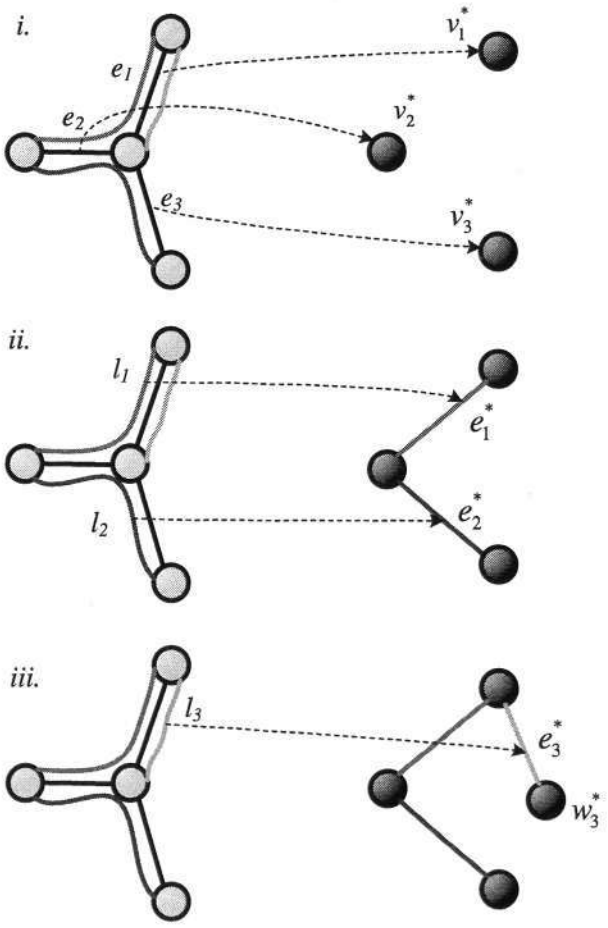


Figure 2.3: Constructing the edge compatibility graph for a star network

pair of vertices in H^*, i.e.H^* is a multi-graph. Hence for the rest of the discussion in this chapter, we shall assume that H^* is a multi-graph.

Since each pair of lightpaths in G_{star} must be assigned with different wavelengths if they occupy the same link, it is easy to see the task of assigning wavelengths to lightpaths in G_{star} is equivalent to that of assigning colors to the edges in H^* such that any two adjacent edges are assigned with different colors, i.e. solving the edge coloring problem on H^*. The edge coloring problem is known to be NP-hard [22, 50] and various results have been proposed in the literatures to provide upper bounds on the chromatic index of a given graph. Some of these results are listed as follows.

Bounds on the chromatic index:

König's Theorem [51] If G is a bipartite multi-graph whose maximum vertex degree is d, then its chromatic index $\chi'(G) = d$.

Shannon's Theorem [52] If G is a multi-graph whose maximum vertex degree is d, then $d \leq \chi'(G) \leq \frac{3}{2}d$.

Vizing's Theorem (extended version) [53] If G is a multi-graph whose maximum vertex degree is d, and if h is the maximum number of edges joining a pair of vertices, then $d \leq \chi'(G) \leq d + h$.

Bounds on the wavelength requirement of a given network:

Theorem 2.4 Given a star network G_{star} with traffic demand D and it's maximum link load is denoted by L, let H^* be its edge compatibility graph constructed using EGCS. If H^* is a bipartite graph, then $W(G_{star}, D) = L$.

Proof: We note the maximum link load L of G_{star} is equal to the maximum degree d of H^*. Thus it follows from König's theorem the chromatic index of H^* is equal to L. This in turn implies that the wavelength requirement of G_{star} is L. ■

Theorem 2.5 Given a star network G_{star} with traffic demand D, let h denote the maximum number of lightpaths occupying the same pair of edges(links) in G_{star}, and let L denote the maximum linkload of G_{star}. Then $W(G_{star}, D) \leq Min(\frac{3}{2}L, L + h)$.

Proof: Let H^* be the edge compatibility graph of G_{star} constructed by EGCS. The maximum linkload L of G_{star} is equal to the maximum degree d of H^*. The maximum number of edges joining a pair of vertices in H^* is equal to the maximum number of lightpaths traversing the same pair of edges in G_{star}, i.e. h. Hence it follows from Shannon's Theorem and Vizing's Theorem that the chromatic index of H^* is bounded from above by $\frac{3}{2}L$ and $L+h$, respectively. This in turn implies that the wavelength requirement of G_{star} is bounded by $Min(\frac{3}{2}L, L + h)$. ■

2.4.3 Network with Bridges

Definition 2.2 Given a network $G(V, E)$, an edge e is called a *bridge* if $G-e$ is disconnected. Let C_1 and C_2 denote the two connected components of $G-e$, let $G_1 = C_1 \cup e$ and $G_2 = C_2 \cup e$, Then we say the two networks G_1 and G_2 are *singly connected* by bridge e (use Figure 2.4 as an illustration).

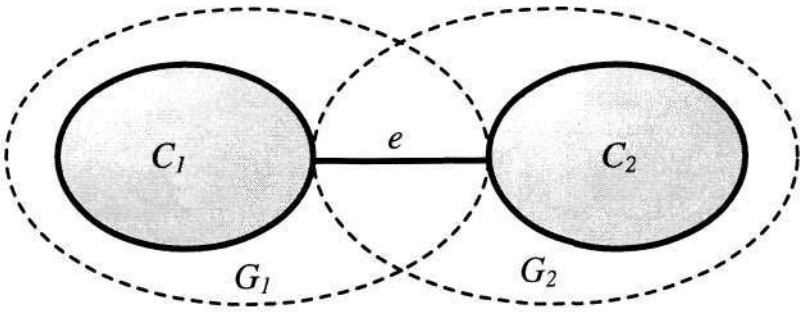


Figure 2.4: Two singly connected networks:G_1 and G_2

Theorem 2.6 Given two networks G_1 and G_2, if G_1 and G_2 are singly connected by bridge e, then $W(G, D) = max[W(G_1, D_1), W(G_2, D_2)]$, where $G = G_1 \cup G_2, D = D_1 \cup D_2$; D_1 and D_2 are the set of lightpaths that traverses G_1 and G_2, respectively.

Proof: Without lost the generality, we assume that $W(G_1, D_1) \geq W(G_2, D_2)$. We note that e is the only common edge of G_1 and G_2. Let T denote the set of lightpaths over e and $T = \{l_1, l_2, \ldots l_k\}, |T| = k$. Consider the case whereby wavelengths have been assigned to all lightpaths in G_1 and G_2 using their respective assignment schemes, which we refer to as *Scheme* 1 and *Scheme* 2.

We note that the wavelengths that have been assigned to G_1 and G_2 will form a valid assignment for G if the two schemes assign the same set of wavelengths to each lightpaths in T. The overall wavelength requirement of G in this case is $W(G, D) = W(G_1, D_1) = max[W(G_1, D_1), W(G_2, D_2)]$.

Next consider the case whereby Schemes 1 and Scheme 2 assign different set of wavelengths to the lightpaths in T. In this case conflict will arise between Scheme 1 and Scheme 2 in the assignment of wavelengths to the common lightpaths in T. Without loss of gen-

erality, we can resolve this conflict by keeping Scheme 1 unchanged while reassigning the wavelengths in Scheme 2 as described below:

Wavelength reassignment for scheme 2:

Let the set of wavelengths assigned to G_1 and G_2 be denoted by A and B, respectively. For each lightpath $l_i \in T$, let $x_i \in A$, $y_i \in B$ denote the wavelengths that have been assigned to l_i using Scheme 1 and Scheme 2, respectively.

1. Construct a bipartite graph $G_R(V, E)$, $V = A \cup B$, $E = \emptyset$;
2. Insert the edge (x_i, y_i) into E for $i = 1, 2, \ldots, k$;
3. For each vertex $b_j \in B - \{y_1, y_2, \ldots, y_k\}$, insert the edge (a_j, b_j) into E, where a_j is a vertex in $A - \{x_1, x_2, \ldots, x_k\}$ which is not adjacent to any vertex in B, i.e. $deg(a_j) = 0$. Since $|A| > |B|$, it is always possible to find such a vertex a_j;
4. Reassign the wavelength in Scheme 2 as follows: Let l_p be a lightpath in D_2 which has been assigned with wavelength p, i.e. $p \in B$. Let the vertex which is adjacent to p in the graph G_R be denoted by q, i.e. $(p, q) \in E$. In this case the lightpath l_p will be reassigned with wavelength q.

From the process described above, we can ensure that the reassignment of Scheme 2 satisfies the following two properties:

- After the reassignment, the set of lightpaths in T will be assigned with the same set of wavelengths by Scheme 1 and Scheme 2. This property is guaranteed by step 2. We can also note since all lightpaths in T shared a common edge, each lightpath will be assigned with a distinct wavelength in both Scheme 1 and Scheme 2. Thus each edge $(x_i, y_i) \in G_R$ inserted in step 2 will not adjacent to any other edges.
- This reassignment will not destroy the validity of Scheme 2. In the step 2 and step 3 described above, we always choose a vertex whose degree is 0 to insert a new edge.

Hence every vertex in A is of degree 1 or degree 0. This implies that in the reassignment of step 4, each wavelength in set B will be replaced by a distinct wavelength. Any pair of lightpaths that are assigned with different wavelengths in Scheme 2 before reassignment will still be assigned with different wavelengths when the reassignment is finished.

Following the reassignment of wavelengths in G_2, the overall wavelength requirements of network G is again bounded by the $W(G_1, D_1) = max[W(G_1, D_1), W(G_2, D_2)]$.

■

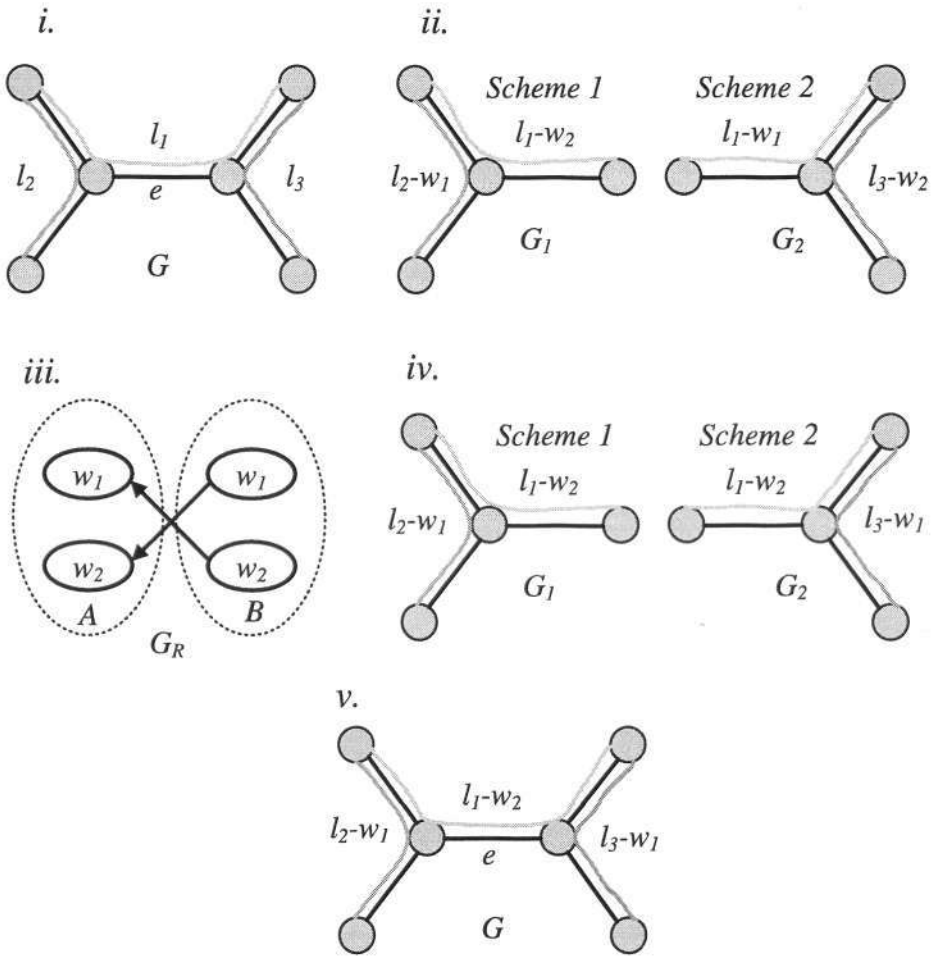


Figure 2.5: Reassignment of wavelengths in Scheme 2

In Figure 2.5, we illustrate this process by a small example. As shown in Figure 2.5-i,

network G is composed by two singly connected stars G_1 and G_2. Three lightpaths l_1, l_2, l_3 are traversing over G. l_1 is the common lightpath shared by G_1 and G_2. Scheme 1 and Scheme 2 assigned different wavelengths(shown in Figure 2.5-ii) to l_1. Thus we need to reassign the wavelengths in Scheme 2. We construct the bipartite graph G_R shown in Figure 2.5-iii, $w_1 \in B$ is connected to $w_2 \in A$ because w_1 is assigned to l_1 in Scheme 2 and w_2 is assigned to l_1 in Scheme 1. After the reassignment for Scheme 2, two schemes assigned same wavelength to their common lightpath l_1(Figure 2.5-iv); thus they can form a valid assignment for G(Figure 2.5-v).

2.4.4 Network with Tree Topology

Theorem 2.7 A tree $G_{tree}(V, E)$ can be constructed by taking a union of a series components $C_1, C_2, \ldots, C_r$, whereby the following conditions hold:

i). $G_{tree} = \bigcup_{i=1}^{r} C_i$;

ii). C_i is either a path or a star, for $i = 1, 2, \ldots, r$;

iii). Given two components: $C_a = \bigcup_{i=1}^{m} C_i$ and $C_b = C_{m+1}$, C_a and C_b are singly connected for $m = 1, 2, 3, \ldots, r-1$.

Proof: We prove this theorem by proposing a tree-construction scheme described as follows:

Tree Construction Scheme(TCS):

Input: A tree $G_{tree}(V, E)$.

Output: An ordered list $C = \{C_1, C_2, \ldots, C_r\}$ that satisfies conditions (i) to (iii) described above.

1. Create an empty set C^*;
2. For each vertex $v_i \in V$, if $deg(v_i) > 2$, then v_i is the center vertex of a star. Thus we insert a star $S_i(V_i, E_i)$ into C^*, where V_i is composed of v_i(center node) and all its neighboring vertices(terminal nodes) in $G_{tree}(V, E)$, E_i is composed by the edges that adjacent to v_i in $G_{tree}(V, E)$.

3. The rest of $G_{tree}(V, E)$, i.e. the linear sub-networks between two star centers or between one star center and one 1-degree node are also inserted into C^* as a set of paths.

4. Now we have a set C^* whereby each member of C^* is either a star or a path. We randomly choose a member in C^* and insert this member into C as C_1. Then for $m = 1$ to $m = r - 1$, we do following: $C_a = \bigcup_{i=1}^{m} C_i$, choose a member which is singly connected to C_a in C^* and insert it into C as C_{m+1}. This process will finish when all members from C^* are inserted into C.

Figure 2.6 gives an illustration for how a tree can be constructed by singly connecting a set of stars and paths. It is easy to verify that the ordered list of components derived by TCS satisfies conditions i-iii, so Theorem 2.7 holds.

■

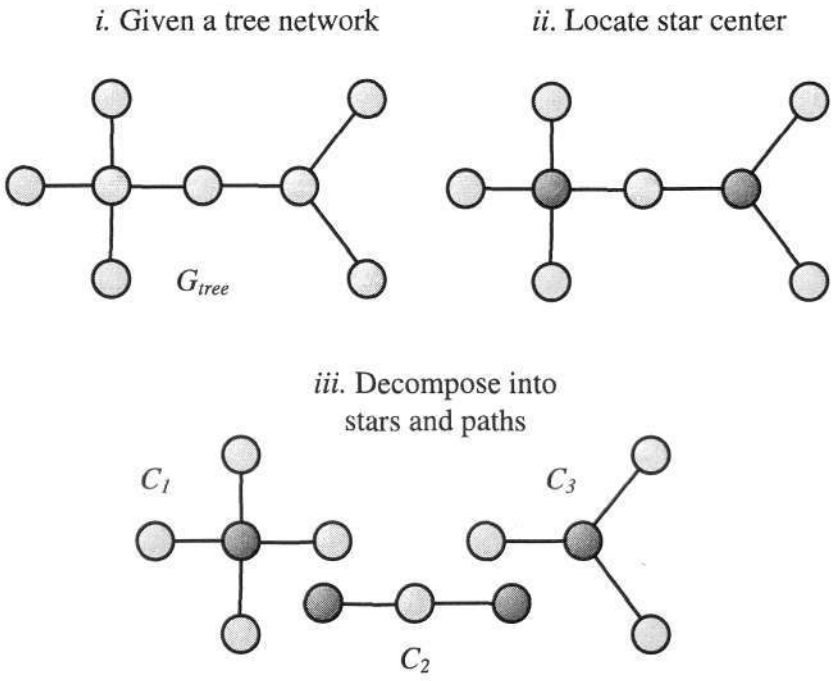


Figure 2.6: Constructing a tree by singly connecting a set of stars and paths

Theorem 2.8 For a tree network G_{tree}, $W(G_{tree}, D) \leq \alpha L$ if and only if for each star sub-network $C_i \subseteq G_{tree}$, $W(C_i, D_i) \leq \alpha L$, where D_i is the set of lightpaths that traverses C_i.

Proof: If: From Theorem 2.6 and Theorem 2.7 we have: $W(G_{tree}, D) = Max[W(C_1, D_1), W(C_2, D_2), \ldots, W(C_r, D_r)]$, where $C_1, C_2, \ldots, C_r$ is a set of star networks or linear networks

that satisfy conditions i-iii stated in Theorem 2.7 and $D_1, D_2, \ldots, D_r$ are the lightpath sets traversing over $C_1, C_2, \ldots, C_r$, respectively. For each linear network C_j, Theorem 2.2 states that $W(C_j, D_j) \leq L \leq \alpha L$. Thus $W(G_{tree}, D) \leq \alpha L$ holds if the wavelength usage of each star networks, $W(C_i, D_i)$, is bounded by αL.

Only if: Since C_i is a sub-network of G_{tree}, $W(C_i, D_i) \leq W(G_{tree}, D)$. Thus if $W(C_i, D_i) > \alpha L$, then we will have: $W(G_{tree}, D) \geq W(C_i, D_i) > \alpha L$. Hence $W(G_{tree}, D) \leq \alpha L$ holds only when $W(C_i, D_i) \leq \alpha L$.

■

2.5 Proposed Algorithm

2.5.1 Algorithm for OPWB

As proven in [54], the problem of determining the wavelength usage of a network is NP-hard. In fact, to the best of our knowledge, no efficient upper bound has been proposed for the wavelength usage of a network with arbitrary topology. In [33], Jia *et al.* showed that even for a network with simple topology(a 4-vertices graph), its wavelength requirement may exceed $\frac{3}{2}L$. Furthermore, in [34] Wilfong *et al.* showed that for a single ring network, its wavelength requirement may also exceed $\frac{3}{2}L$. Fortunately, if G is a tree network then its wavelength usage can be bounded by $\frac{3}{2}L$ regardless of the traffic demand [44]. Based on this fact, in the first step of our proposed algorithm, we aim to determine the minimum set S_1 so that $G_{S1}(V', E) = split[G(V, E), S_1]$ will be a tree or a forest(a set of disconnected trees). This problem is often referred to as the minimum feedback set problem and is known to be NP-complete [22]. However, as a well-studied problem, there exist some approximation algorithms with good performance guarantee. For example, in [55], a 2-approximate algorithm is proposed for this problem. Thus we can determine the vertex set S_1 which will be equipped with converters by applying these approximation algorithms.

After the converters are placed at each node in S_1, the wavelength usage of $G_{S_1}(V', E)$

is bounded by $\frac{3}{2}L$. We can further tighten this bound by applying step 2. In this step, for each star sub-network, we examine the upper bound for its wavelength requirement which is determined using Theorem 2.4 and Theorem 2.5. For those star sub-networks whose upper bounds on wavelength usage exceed αL, we will include their center nodes into set S_2. Converters will be placed at each node in S_2. After all these converters are placed, some stars are split into paths and we let the resultant network be denoted as G_S. We can guarantee that for all remaining stars $C_i \subseteq G_S$, $W(C_i, D_i) \leq \alpha L$. Thus from Theorem 2.8 we have $W(G_S, D) \leq \alpha L$, which implies that the total wavelength usage is bounded by αL and the total number of converter nodes is $|S_1| + |S_2|$. Our algorithm can be described by the pseudo code described in Table 2.1.

2.5.2 Performance Analysis

Computational complexity

Theorem 2.9 The computational complexity of proposed two-step algorithm is $O(|E||V| + |D||V|)$.

Proof: In the first step, finding the minimum feedback set by using the approximation algorithm proposed in [55] can be done in $O(|E||V|)$ time; splitting operation can be done in $O(|E|)$ time in the worst case. In the second step, each legs of the stars should be checked to determine the maximum linkload of stars, we note each edge can be included in two stars at most so the complexity of checking linkload is $O(|E|)$. Next we note the number of lightpaths after Step 1 is $|D||V|$ at most, thus building edge compatibility graph by EGCS can be done in $O(|E| + |D||V|)$; checking whether the edge compatibility graph is bipartite for all stars can be done in $O(|E| + |D||V|)$. Step 2 will cost $O(|E| + |D||V|)$ and the two-step algorithm we proposed will cost $O(|E||V| + |D||V|)$ in the worst case. ■

Table 2.1: A two-step algorithm for OPWB

Input: Network $G(V, E)$, $V = \{v_1, v_2, \ldots, v_n\}$, $E = \{e_1, e_2, \ldots, e_m\}$ with traffic demand set $D = \{l_1, l_2, \ldots, l_k\}$, the upper bound for the wavelength usage αL.
Output: Vertex set S.

1. ***Step 1*** (place converters at the feedback set nodes):

   $S = \emptyset, S_1 = \emptyset$

   find the minimum feedback set S_1 for G

   $S = S \cup S_1$

   $G_S(V, E) = split[G(V, E), S]$

2. ***Step 2*** (place converters at centers of star networks):

   $S_2 = \emptyset$

   for $i = 1$ to n: /* for each vertex v_i

   if $deg(v_i) \leq 2$ /*v_i is not a star center

   $i++$

   else

   build the edge-compatibility graph H_i^* of the star with center

   v_i by EGCS

   check whether H_i^* is a bipartite graph

   check the value of l and h, which denotes the maximum linkload and the maximum number of edges joining a pair of vertices, separately

   if H_i^* is not a bipartite graph **and** $Min(\frac{3}{2}l, l + h > \alpha L)$

   $S_2 = S_2 \cup v_i$, $i++$

   endfor

   $S = S \cup S_2$

   output vertex set S

The setting of α

As mentioned in Section 2.1, the size of converter nodes set S is determined by network topology, traffic demand and given bound for the wavelength usage. In this section we will investigate the relationship between $|S|$ and the value of α:

1. $\alpha = 1$: In this case the wavelength usage is the minimum possible. Thus the size of S would be large. In the worst case, every center node of stars will be equipped with converters so the network will be split into a set of linear networks by node set S; this is the case that studied by [32].

2. $\alpha = \frac{3}{2}$: It is proved in [44] and [33] that for the network with tree topology, this upper bound can always be met for arbitrary traffic demand. Thus we do not need to place any converter in the second step. We can also note that in this case the OPWB is equivalent to the minimum feedback set problem. The wavelength converter placement problem under this case is studied by [33].

3. $1 < \alpha < \frac{3}{2}$: This is the general case that we are addressing in this chapter, as shown our algorithm will generate a vertex set S with the size between case 1 and case 2.

2.6 Simulation Results

In this section, we adopt the experimental approach to study the relationship between the size of S and the value of α . The converter set S is constructed by the proposed algorithm. Three typical networks were studied which include NSFnet network, USA long haul network and mesh network. We also varied the size of mesh network from 4×4 to 7×7 to evaluate the effects of the network size. Some statistics of these networks are listed in Table 2.2.

Traffic demand are generated for each pair of vertices with probability p, where p is a parameter controlling the total traffic load of the network. In this study we defined three types of traffic load condition:

Table 2.2: Statistics of some typical networks

Topology	Number of vertices	Feedback set size	Number of vertices whose degree larger than two
NSFnet	14	3	10
USA long haul	28	8	21
4×4 mesh	16	4	12
7×7 mesh	49	13	45

i). Low traffic load, where p is set to 0.2;

ii). Moderate traffic load, where p is set to 0.5;

iii). High traffic load, where p is set to 0.8;

All traffic demands are routed by the shortest path algorithm. For each traffic load condition, we calculate size of the converter nodes set $|S|$ by taking the average of ten times repeated simulation. The results are shown in Figures 2.7 to 2.10.

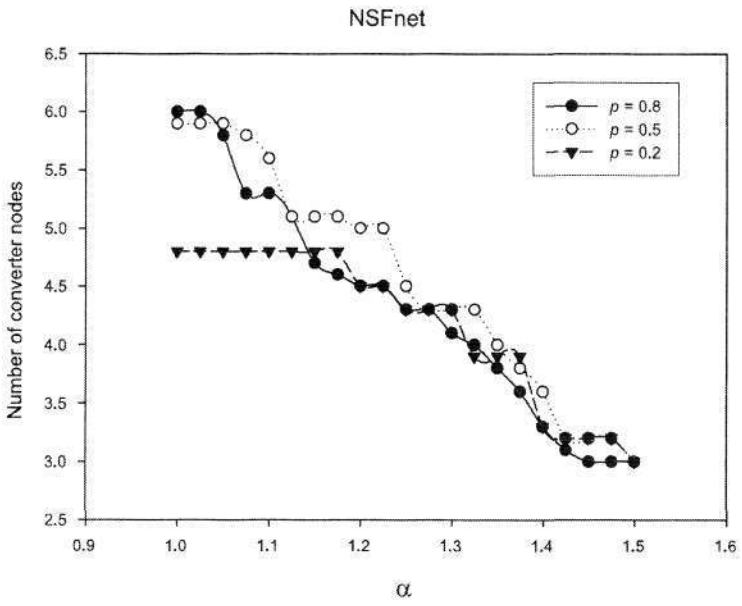


Figure 2.7: Number of converter nodes in NSFnet

The results show that the size of S decreases as α increases from 1 to $\frac{3}{2}$. This phenomenon is expected because larger number of available wavelengths results in less deployment of

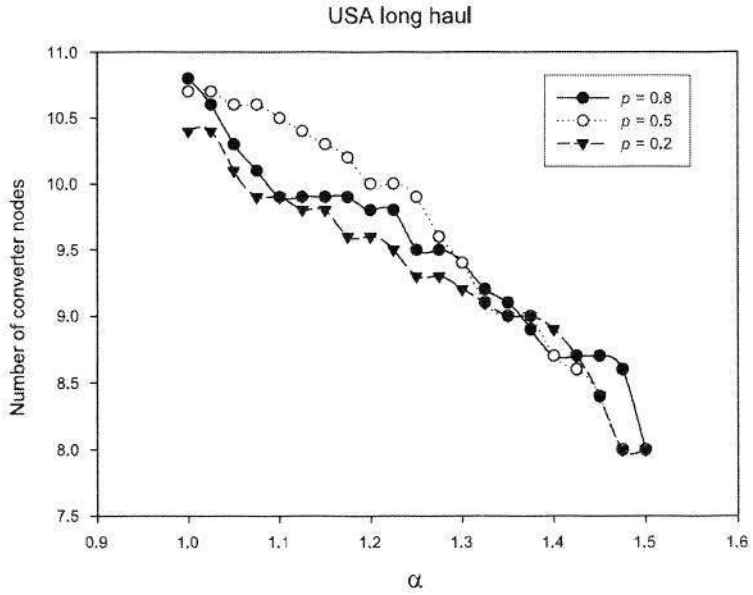


Figure 2.8: Number of converter nodes in USA long haul network

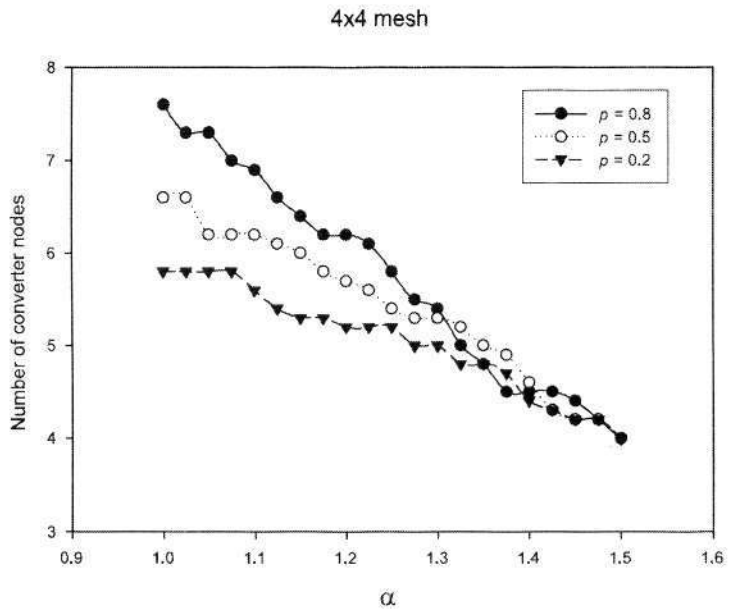


Figure 2.9: Number of converter nodes in 4×4 mesh network

converters. It indeed verify that correctness of the implementation of our proposed algorithm. With these $|S|$ *vs* α curves, the network designer can easily estimate the upper bound of wavelength usage when given the number of wavelength converters or estimate the number of required converters when given the upper bound for the wavelength usage.

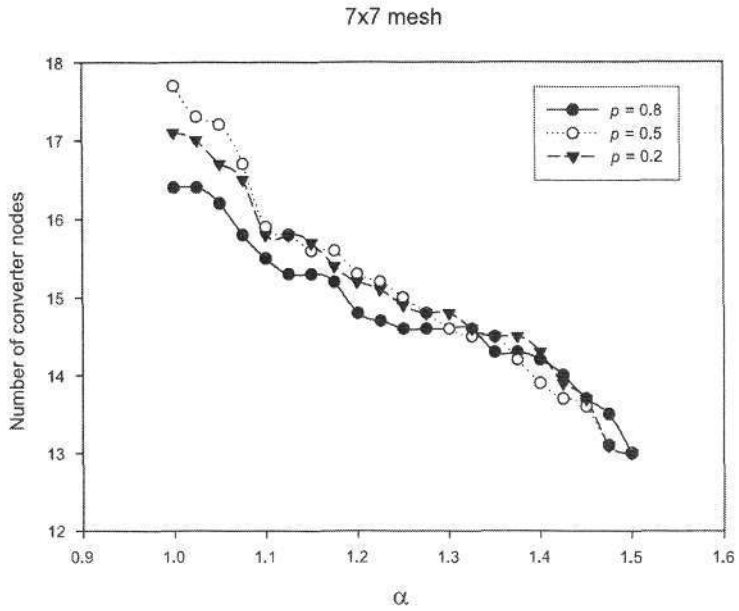


Figure 2.10: Number of converter nodes in 7×7 mesh network

Next we note when the traffic load of network increases, both the number of lightpaths and the maximum linkload L increase. This in turn results in an increase in the number of wavelengths needed (i.e αL) when the number of converters is fixed. However, in Figure 2.7, 2.8 and 2.10 we can see that the numbers of converters(i.e $|S|$) placed under three traffic load conditions differ from each other by no more than 1 when $\alpha \in [1.1, 1.5]$. In Figure 2.9, although the difference of $|S|$ under three traffic load conditions is around 2 when $\alpha = 1.0$, this difference decrease fast as α increases. Thus the experimental results shows that the traffic load parameter p has little effect on the number of converters placed by the proposed algorithm. In addition, we note that our proposed algorithm is able to achieve similar trend of performance (in terms of the trade-off between the number of wavelength converters needed and the value of α) in three traffic conditions. For instance, in Figure 2.8 we can see that the average value of $|S|$(i.e the number of converters placed) is around 10.5 when $\alpha = 1$ and it will decrease as α increases; when $\alpha = 1.5$, the average value of $|S|$ is decreased to 8, which is equal to the feedback set size of the network topology.

We could also note when $\alpha = 1$, which means the wavelength usage is the minimum

possible, the size of S constructed by the proposed algorithm is much smaller than the size of converter set constructed by the algorithm proposed in [32], which is equal to the number of vertices whose degree is larger than two (listed in the last column of Table 2.2). For instance, the number of wavelength converters needed for the case of NSFnet when $\alpha = 1$ and $p = 0.5$ in Figure 2.7 is equal to 6 while the number needed by the algorithm proposed by Jia *et. al* [32] is equal to 10. We have noted that the algorithm proposed by Jia *et. al* [32] does not consider a given set of lightpaths as its design input. The results obtained with the algorithm proposed in [32] gives the bound for the number of converters under the worst-case scenario. On the other hand, the given set of lightpaths are taken into consideration as a design input in our algorithm. Simulation results show that this will help to reduce the redundant deployment of wavelength converters and the number of converters placed can be much reduced.

2.7 Summary

In this chapter, we have studied the problem of placing a minimal set of wavelength converters in WDM networks with arbitrary topology and the total wavelength usage is bounded. The traffic demand is also taken into consideration. In this work, the network designer can set the upper bound for wavelength usage in the range of $[L, \frac{3}{2}L]$. Thus the proposed algorithm is more flexible compared to existing work in this area. A two-step algorithm is proposed for this problem, its correctness is guaranteed by a set of theorems and its effectiveness is evaluated by both theoretical and experimental studies.

This work can benefit WDM network design and development in several aspects. Firstly, by considering the traffic status, the number of converter required can be further reduced compared to earlier works. Secondly, it can help us to understand the relationship between the number of converters and the bound on wavelength usage, thus enabling more efficient utilization of wavelength converters. Thirdly, by adopting our two-step algorithm and wave-

length switching techniques, the wavelength assignment problem for a network with arbitrary topology can be reduced to a wavelength assignment problem in a set of independent stars and paths, which in turn helps in reducing the overall computational complexity.

Chapter 3

Load-Balanced Clustering Problem in Wireless Sensor Networks

Wireless sensor networks (WSNs) have been receiving increasing attention in recent years due to their potential applications in the establishment of dynamic communications for emergency/rescue operations, disaster relief efforts, and military networks. The cluster scheme is a hierarchy structure which is essential for achieving a basic performance guarantee in a large-scale sensor networks. In a clustering scheme the sensor nodes are divided into different virtual groups, and they are allocated to certain cluster according to some criterions.

In this chapter, we investigate the problem of grouping the sensor nodes into clusters to enhance the overall scalability of the network. A selected set of nodes, known as gateway nodes, will act as cluster-heads for each cluster and the objective is to balance the load among these gateways. Load balanced clustering increases system stability and improves the communication between the various nodes in the network. We call the problem addressed in this chapter as the *Load-Balanced Clustering Problem* (LBCP). We first show that a special case of LBCP (whereby the traffic loads contributed by all sensor nodes are the same) is optimally solvable in polynomial time. We next prove that the general case of LBCP is NP-hard. We then proposed an efficient $\frac{3}{2}$-approximation algorithm for the problem.

3.1 Introduction

The availability of cheap, low power, and miniature embedded processors, radios, sensors, and actuators, often integrated on a single chip, is leading to the use of wireless communications and computing for interacting with the physical world in applications such as security and surveillance applications, smart classroom, monitoring of natural habitats and eco-systems, and medical monitoring. The resulting systems, often called wireless sensor networks [7–9], differ considerably from current networked and embedded systems. They combine the large scale and distributed nature of networked systems such as the Internet with the extreme energy constraints and physically coupled nature of embedded control systems.

A sensor network is composed of a large number of sensor nodes (or sensors), which are densely deployed either inside the *phenomenon* (i.e. something known by sense perception) or very close to it. Sensors are generally equipped with data processing and communication capabilities. These sensing circuit measures parameters from the environment surrounding the sensor and transforms them into an electric signal. Processing such signals reveals some properties about phenomenon and/or objects located in the vicinity of the sensors. Data collected from each sensor are routed back to a base station or command node, either periodically or based on events. To avoid long-haul communication with the command node, some high-energy nodes called *gateways* are typically deployed in the network. Sensor nodes are group into distinct clusters by using each of these gateways as the cluster-head of a cluster. Each sensor node only belongs to one cluster and communicates with the command node through the gateway (or cluster-head) in the cluster. It is easy to see that by adopting a cluster-based network architecture, several benefits can be obtained:

1. The overall traffic load can be better distributed among the nodes in the various clusters and the end-to-end transmission delay between a sensor node and the command node can be reduced. This in turn enhances the overall scalability of the network.

2. The cluster structure is an effective topology control means [56, 57] which can fa-

cilitates the spatial reuse of resources to increase the system capacity [58]. In the non-overlapping multi-clusters, two clusters may deploy the same frequency or code if they are not neighboring clusters [59].

3. Routing problem may be better solved. The set of gateway nodes can form a virtual backbone for inter-cluster routing. Thus the generation and spreading of routing information can be restricted in the set of nodes with bounded size [60, 61].

4. A cluster structure makes the sensor network appear smaller and more stable in the view of each sensors [62]. When a sensor node fails or move out of a cluster, only the nodes which belong to the corresponding cluster need to update the information. This will results in a great reduction of the information stored and processed in each sensor node [63, 64].

A multi-gateway architecture is required to cover a large area of interest without degrading the service of the system. But, if the sensor nodes and the gateways are not "well distributed", some gateways may be overloaded with increase in sensor density and detected phenomenons/targets. Such overload may increase latency in communication and cause degradation of overall network performance.

Hence, in this chapter we address the problem of assigning sensor nodes to gateways to form clusters with the objective of minimizing the maximum load of each gateway in the given network. We note that the effort to minimize that maximum load of each gateway will result in a more balanced distribution of loads among the set of gateways. We refer to this problem as the *Load-Balanced Clustering Problem* (LBCP). The problem here is to determine for each sensor node, the gateway to which it should be assigned, under the constraint that each sensor node must be connected to one and only one gateway, in order to minimize the overall maximum load of the gateways.

We first consider a special case of the problem whereby the offered traffic loads from all sensor nodes are the same. We refer to this special case as the *Load-Balanced Clustering*

Problem with *Uniform Traffic Load* (LBCP-UTL). We show that this special case is optimally solvable in polynomial time. We next consider the general case of the problem in which the traffic load from each sensor node may differ from one another. We show that the problem in this case is NP-hard and we propose $\frac{3}{2}$-approximation algorithm for the problem.

The rest of this chapter is organized as follows. In the next two sections we describe the architectural model of sensor network and summarizes related work. In Section 3.4, we give a formal definition of the Load-Balanced Clustering Problem. A polynomial time algorithm that optimally solves LBCP-UTL, a special case of LBCP is described in Section 3.5. The general case of LBCP is considered in Section 3.6. This problem is shown to be NP-hard and an approximation algorithm for the problem is given in this section. Simulation results to evaluate the performance of our proposed approximation algorithm will be described in Section 3.7. The Chapter concludes with Section 3.8.

3.2 System Architecture

The system architecture for the sensor network is shown in Figure 3.1. The gateway node also act as cluster head for each clusters. Gateway nodes are less-energy constrained compared to sensor nodes. All communication is over wireless links. A wireless link is established between two nodes only if they are in range of each other. Gateways are capable of long-haul communication compared to sensor nodes and all gateway nodes are assumed to be in communication range of one another. In this chapter, we assume that the sensor nodes and gateways are stationary.

3.3 Related Work

Clustering is an important problem in WSNs and MANETs and thus has been addressed in many research works. These works can be classified according to the optimization objectives.

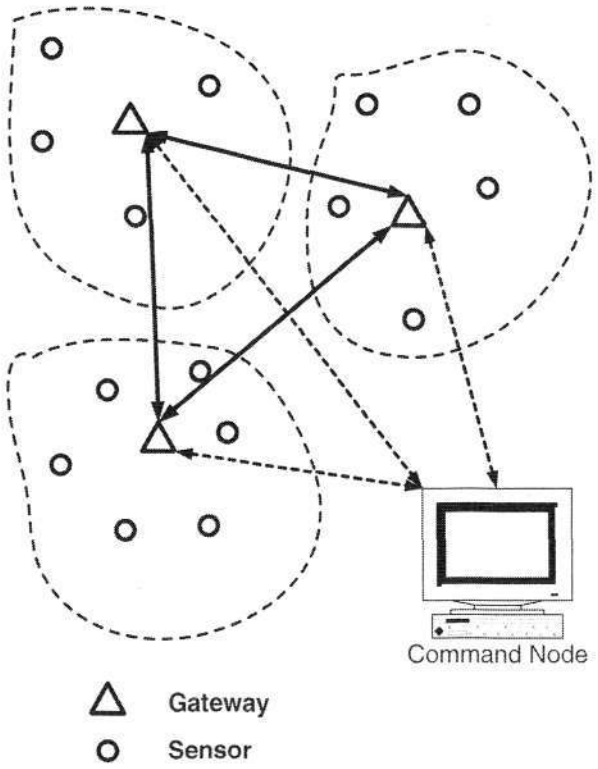


Figure 3.1: Multi-gateway Clustered Sensor Network

A typical approach to cluster formation is to compute a Connected Dominating Set (CDS) of the network topology graph. The nodes which belong to the CDS will be elected as cluster heads. This set of cluster heads will be used as a backbone of the network to relay routing information and data packets. A good backbone should first and foremost be small; additionally, it should have other characteristics such as robustness to node failures (i.e., hard to disconnect) and low stretch, i.e., routes in the backbone should not be much longer than the shortest routes over the network topology graph. In [65], Wu *et al.* proposed a simple distributed scheme which can creates CDS. This scheme was proven to be efficient but tends to create large CDSs. Therefore some other rules which aim to prune away redundant CDS nodes were also discussed. Wu's scheme was improved by Stojmenovic *et al.* [66]. In Stojmenovic's scheme, each node can locally determine whether it belongs to a cluster without exchanging its membership information with its neighbors. This results in a reduction of message complexity. By noticing that CDS clustering usually produce a large

number of clusters and these clusters are likely highly overlapping, Chen *el.al* [67] addressed the problem of finding a small Weakly-Connected Dominating Set (WCDS) in a given graph which can be used as cluster head in a network. Normally WCDS is smaller in size than CDS, thus it is better for further simplifying the network structure. Chen *el al.* proposed set of five approximation algorithms for determining WCDS and showed that they have the performance ratio of $O(ln\Delta)$, where Δ is the maximum degree of the network graph. Some other algorithms for determining WCDS with improved message complexity, computational complexity or localized degree were proposed in [68–70].

Since the nodes in WSNs and MANETs may move out of communication range and may run out of power, the network topology may change frequently. This in turn results in frequent cluster topology updates. Thus the control overheads for cluster maintenance should be effectively controlled to avoid overly consumption of network bandwidth and the energy of nodes. Being aware of this issue, some cluster schemes aimed at minimizing the maintenance cost. In [71], Chiang *et al.* proposed the Least Cluster Change (LCC) clustering scheme which is claimed to provide the stablest cluster structure for grouping mobile nodes and allocating radio channel codes. A non-overlapping clustering algorithm, which is referred as 3-Hop Between Adjacent Cluster Heads (3hBAC) was proposed by Yu *et al.* [72]. This scheme will form 1-hop non-overlapping clusters with three hops between neighboring cluster heads. In the cluster maintenance phase, Yu *et al.* combined 3hBAC with LCC algorithm to further reduce the reconfiguration of clusters and to extend average duration of a node being a cluster head or a cluster member. In [73], Kwon *et al.* proposed the passive clustering scheme, which does not use dedicated clustering control packets or signals. Passive clustering can form and maintain its cluster structure without explicitly exchanging the clustering control packets. Thus the control overhead caused by active clustering can be completely eliminated.

The nodes in a WSN or MANET are normally powered by batteries; hence the schemes and protocols should be designed to be energy-efficient in order to prolong the network

lifespan. In addition, the cluster head nodes bears extra work compared with ordinary nodes and it is likely to runs out of batteries earlier. By taking the energy issue into consideration, some of the works aimed at maximizing the lifespan of the network by minimizing the energy consumption or balancing the energy consumption among nodes in the network. In [74], Wu *et al.* proposed a cluster scheme which minimizes the energy consumption by decreasing the size of dominating set. This is based on the observation that nodes which are included in a dominating set consume remarkably more energy because these node will bear extra tasks. In [75], Yu *et al.* proposed a novel cluster head election scheme: each node estimates the number of active nodes in realtime and computes its optimal probability of becoming a cluster head so that the amount of energy spent in both intra- and inter-cluster communications can be minimized. The estimation of the number of active nodes is based on monitoring the received signal power from its neighboring nodes. The trade off between network throughput capacity and power consumption of clustering was studied in [76].

Some researches assume that there is an optimum number of member nodes that a cluster can handle. A over-sized cluster may put too much load on the cluster heads and reduce the system throughput. A under-sized cluster, on the other hand, will results in a large number of clusters which may also degrade the system performance. Thus some schemes took load-balance issue into consideration and tried to control the size of each cluster. In [77], Ohta *et al.* use the upper bound and lower bound to control the size of each cluster. In the scheme proposed by Ohta *et al.*, a over-sized(above upper bound) cluster will be divided into two smaller clusters and two under-sized(below lower bound) may merge into a larger cluster. In the scheme proposed by Amis *et al.* [78], the clustering scheme runs periodically in order to keep the number of mobile nodes in each cluster around a optimum value. A cluster head will degrade to an ordinary member node if the difference between the number of nodes that it currently serves and the optimum one exceeds some pre-defined value. In [79], Krishnan *el al.* propose two algorithms which can produce clusters of bounded size and low diameter by using a novel approach, in which nodes will allocate local "growth budgets"

to neighbors.

Although clustering problem has caught the attention of many researchers, most of the published clustering protocols do not consider the load balancing among clusters due to variable density of nodes and variable traffic load of each node in the system. Since a system that is not load-balanced will give rise of clusters with high-traffic-density and very low-traffic-density, the high-traffic-density cluster head will be overwhelmed with the processing and communication load and will be depleted of energy at a much faster rate than low-density cluster-heads. In [77–79], the clustering schemes which aim to produce clusters of restricted size were proposed. However such schemes will require all nodes to have uniform traffic load to ensure that the overall load is sufficiently balanced among all clusters in the network. This requirement may not be practical in many cases. For example, in the energy saving scenario, sensor nodes will be in the sleep mode while no targets are detected, the traffic load contributed by a sleeping sensor is zero or very close to zero. After a target has been detected, a portion of sensors will be awaken and monitor the target, the traffic loads contributed by these active sensors will be much larger than those contributed by sleeping sensors.

The procedure for cluster formation consists of two phases: gateway (cluster head) election and assignment of sensor nodes to gateways. We find most existing works only put emphasis on the first phase and the problem of assigning sensor nodes to gateways has not been sufficiently addressed.

In this chapter, we take the above-mentioned issues into consideration in the design of clustering scheme. We consider a network scenario where gateway nodes are chosen a priori and are fixed throughout the network lifetime. We then address the problem of cluster formation by assigning sensor nodes to gateways to with the aim of balancing the load among the gateways. The main objective of our work in this chapter is to cluster sensor nodes in a network efficiently around several high-energy gateway nodes. Clustering enable network scalability to large number of sensor nodes and extends the life of the network by

allowing the sensor nodes to conserve energy through communication with closer nodes and by balancing the load among the gateway nodes. Clusters are formed based on the load of the gateways. The network topology is assumed to be static, as in sensor networks or slowing changing.

3.4 Problem Formulation

In this section, a formal definition of the Load-Balanced Clustering Problem will be given. We adopt the following assumptions and notations in the problem formulation.

- we consider a network scenario where gateways are chosen a priori and the locations of sensor nodes are known
- the load (which is a function of processing load and communication load) contributed by each sensor node can be estimated
- the set of sensor nodes is denoted by T and $\mid T \mid = n$
- the set of gateways is denoted by C and $\mid C \mid = m$
- $n > m$, i.e. the number of sensor nodes is greater than the number of gateways
- d_i denotes the traffic load contributed by sensor t_i where $t_i \in T$ and $d_i \in Z^+$
- C_i denotes the set of gateways onto which sensor t_i may be assigned, where $t_i \in T$. We note that some constraints may be imposed such that a given sensor t_i can only be assigned to a member of a selected set of gateways C_i, where $C_i \subseteq C$. Examples of constraints that may restrict the assignment of sensors to gateways include that of ensuring that a given pair of sensor & gateway are within the communication range of each other. We refer to such constraints as the *assignment constraints*.

Formulation as a Mathematical Program

Prior to the problem formulation, the following variables are defined.

- x_{ij}: a binary variable, 1 if sensor t_i is assigned to gateway c_j, otherwise 0
- α : the maximum load that may be assigned to a gateway

The Integer Linear Programming (ILP) formulation of the Load-Balanced Clustering Problem is defined as follows:

Objective function:

$$Minimize \quad \alpha \tag{3.1}$$

Subject to

$$\sum_{c_j \in C_i} x_{ij} = 1, \qquad \forall t_i \in T \tag{3.2}$$

$$\sum_{t_i \in T} d_i \cdot x_{ij} \leq \alpha, \qquad \forall c_j \in C_i \tag{3.3}$$

The objective (3.1) is to minimize the overall maximum load of the gateways. Constraint (3.2) states that each sensor should be assigned to one (and only one) gateway. Constraint (3.3) imposes the condition that the total traffic load of all sensors assigned to a particular gateway should not exceed the maximum load permitted.

3.5 The Load-Balanced Clustering Problem with Uniform Traffic Load (LBCP-UTL)

In this section we consider a special case of LBCP whereby the traffic load from each sensor is the same, i.e. $d_i = \beta$ for some constant $\beta \; \forall i \in T$. Without loss of generality, lets assume that $\beta = 1$. We note that the problem of minimizing the maximum load of each gateway in this case is equivalent to that of minimizing the maximum number of sensors that are

assigned to each gateway. Let $l(j)$ denote the number of sensors that are assigned to gateway j, where $j \in C$. The Load-Balanced Clustering Problem with Uniform Traffic Load is that of finding as assignment $A : T \rightarrow C$ such that $A(i) \in C_i$ and l_{max} is minimized, where $l_{max} = \max_{j \in C} l(j)$ and $l(j) = |\{i \in T : A(i) = j \ \& j \in C_i\}|$.

We say that sensor i is *assigned* to gateway j if $A(i) = j$. An *optimal solution* for LBCP-UTL is an assignment A for which the resulting l_{max} is the least possible. In this section we propose an algorithm, called the *Load-Balanced Clustering Algorithm* (LBCA), that optimally solves LBCP-UTL in $O(mn^2)$.

Let the set of sensors to be assigned be denoted as $T = \{t_1, t_2, \ldots, t_n\}$ and the set of gateways available be denoted by $C = \{c_1, c_2, \ldots, c_m\}$. Starting with sensor t_1, LBCA will construct a BFS tree rooted at t_1. The set of gateways onto which sensor t_1 may be assigned, i.e. C_1, is next included into the tree. Since t_1 is the first sensor to be assigned, it is easy to see that each of the gateways in C_1 will have zero load at the moment. The algorithm will arbitrarily assign t_1 to one of these gateways. Next the algorithm proceeds to construct a breadth-first search tree rooted at t_2 and so on. In general a BFS tree rooted at t_i is constructed level-by-level with t_i in level 1 and the gateways onto which it may be assigned in level 2. The tree is next extended to the 3rd level by including the set of sensors that was previously assigned to the set of gateways in level 2 into the 3rd level of the tree. Observe that the tree alternates between sensors and gateways from one level of the tree to the next, with the sensors and gateways occupying the odd levels and the even levels, respectively. A queue is used to maintain the list of gateways and sensors that appears in the BFS tree. In particular, the set of gateways onto which sensor t_i may be assigned is first inserted in the queue Q. Next, an element v of Q is removed from the front of Q and the sensors that have been assigned to gateway v are inserted at the back of Q; if the element v that is removed from the front of Q is a sensor, then the corresponding gateways onto which sensor$v$ may be assigned, are inserted at the back of Q. This process continues until either one of the following two conditions is true: (i) a gateway with zero load is found or (ii) the set Q is

empty. Once the tree rooted at sensor t_i is constructed, the next step is to make adjustments to the previous assignment of sensors $\{t_1, t_2, \ldots, t_{i-1}\}$ to the set of gateways in C so that sensor t_i can be assigned to one of the gateway in C_i while at the same time ensuring that the maximum load of the set of gateways in C is minimized. This process is carried out as follows. We first identify a gateway, say $r_k \in C$, with the least load in the BFS tree rooted at t_i and let's assume that r_k is at level k of the BFS tree. The predecessor of r_k will be a sensor (in level $k-1$) that may be assigned to gateway r_k. Let this sensor be denoted by s_{k-1}. The predecessor of sensor s_{k-1} will be a gateway at level $k-2$ onto which sensor s_{k-1} was assigned in previous assignment. Let this gateway be denoted by r_{k-2}. This process continues until the root of the tree, namely t_i, is reached. At this stage we would have found a path P from t_i to the gateway r_k which is comprised of a set of edges which connects a sequence of vertices which alternate between sensors and gateways. In particular, let P be denoted by : $t_i \rightarrow r_2 \rightarrow s_3 \rightarrow \cdots \rightarrow s_{k-1} \rightarrow r_k$, where sensor s_3 was previously assigned to gateway r_2, sensor s_5 was assigned to gateway r_4 and in general sensor s_{2i+1} was assigned to gateway r_{2i}, where $i = 1, 2, \ldots, k/2 - 1$. Having identified the path P, the sensors are next reassigned to the gateways as follows:

(i) assign sensor t_i to gateway r_2,

(ii) next reassign sensor s_3 to gateway r_4 and

(iii) in general reassign sensor s_{2i-1} to gateway r_{2i}, where $i = 2, 3, \ldots, k/2$.

The pseudocode of the algorithm is shown in Table 3.1.

Lemma 3.1 The load of each gateway in the path $P = t_i \rightarrow r_2 \rightarrow s_3 \rightarrow \cdots \rightarrow s_{k-1} \rightarrow r_k$ will remain unchanged after the reassignment of the sensors in P to the set of gateways in P except for gateway r_k whose load will be increased by one.

Proof: After the reassignment procedure, sensor t_i will be a new sensor to be assigned to gateway r_2. However, sensor s_3 which was previously assigned to gateway r_2 will now

Table 3.1: Load-Balanced Clustering Algorithm

INPUT: A set of sensors $T = \{t_1, t_2, \ldots, t_n\}$, a set of gateways $C = \{c_1, c_2, \ldots, c_m\}$ and traffic load β for each sensor t_i .

OUTPUT: An assignment $A : T \rightarrow C$ such that $A(i) \in C_i$ and l_{max} is minimized, where $l_{max} = \max_{j \in C} l(j)$ and $l(j) = |\{i \in T : A(i) = j\}|$.

Begin

***Step 1*:**

  for $j = 1$ to m **do**

    set $l(j) = 0$;

  endfor

***Step 2*: /* construction of BFS tree */**

  for $j = 1$ to n **do**

    set $l_{min} = \infty$;

    $Q = \{t_j\}$;

***Step 3*:**

    while $(Q \neq \emptyset)$ and $(l_{min} > 0)$ **do**

      let v be the front element of Q;

      remove v from Q;

***Step 4*:**

      if v is a sensor **then**

        for each unmarked gateway w onto which v may be assigned **do**

          mark w;

          insert w to the end of Q;

          set $pred(w) = v$;

        endfor

***Step 5*:**

      else /* v **is a gateway** */

        if $l(v) < l_{min}$ **then** $l_{min} = l(v)$;

        for each sensor w that was assigned to gateway v **do**

          insert w to the end of Q;

          set $pred(w) = v$;

        endfor

    endwhile

***Step 6*: /* reassignment of sensors to gateways */**

    let v be a gateway with the least load;

    let $w = pred(v)$;

    assign sensor w to gateway v;

    increase load of gateway v by β;

    while $w \neq t_j$ **do**

      $v = pred(w)$;

      remove the previous assignment of sensor w to gateway v;

      let $w = pred(v)$;

      assign sensor w to gateway v;

    endwhile

  endfor

End

be reassigned to gateway r_4. Hence the load of gateway r_2 remains unchanged. Similarly, sensor s_5 that was previously assigned to r_4 will now be reassigned to r_6. Hence the load of gateway r_4 will also remain unchanged. By similar arguments, it is easy to see that the load of gateway r_{2i} , where $i < k/2$, will remain unchanged. Next, we see that since sensor s_{k-1} will be reassigned to gateway r_k, the load of gateway r_k will be increased by one following the reassignment. ■

Theorem 3.1 The Load-Balanced Clustering Algorithm (LBCA) produces an optimal solution for LBCP-UTL.

Proof: Let A be an assignment that is obtained using LBCA. Let A_j denote the assignment of (any) j sensors from T to the gateways in C using LBCA. Let the set of sensors in A_j be denoted by $T_j = \{t_1^j, t_2^j, \ldots, t_j^j\}$. Let c_i^j denote the gateway onto which sensor t_i^j is assigned in assignment A_j, where $1 \leq i \leq j$. Let $R_{t_i^j}$ denote the BFS tree rooted at t_i^j that is constructed using LBCA. Let P_j denote the path in $R_{t_i^j}$ that connects t_i^j to a least loaded gateway, say c, in $R_{t_i^j}$. We will prove by induction that the following two conditions, referred to as *optimality conditions* are satisfied for all values of j: (i) A_j is an optimal assignment for T_j and (ii) $l(c) \leq l(w)$, where w is a gateway with the least load in a given assignment.

It is clear that A_1 is an optimal assignment (as there were no other assignment prior to this and sensor t_1^1 is being assigned to a gateway, say c_1^1 , with zero load). The resultant assignment has a maximum load of one. In addition, we note that a gateway c with the least load in $R_{t_1^1}$ has load equal to (i) zero if $|C(t_1^1)| > 1$ or (ii) one if $|C(t_1^1)| = 1$. In either case, it is easy to see that for any given assignment with a least loaded gateway w, $l(w) \geq l(c)$.

Next we assume that both the above-mentioned optimality conditions are satisfied when $j = k - 1$. Let the maximum load of A_{k-1} be denoted by $l_{A_{max}}^{k-1}$, where $k \geq 2$. We will next argue that A_k is also an optimal assignment, i.e the resulting maximum load is the least possible. Let P_k denote the path in $R_{t_k^k}$ that connects t_k^k to a least loaded gateway, say c, in $R_{t_k^k}$. Let $l_{A_{k-1}}(c)$ denote the load of c which results from assignment A_{k-1}. After the

reassignment of the sensors in P_k to the set of gateways in P_k, the load of gateway c will be increased by 1 (Lemma 3.1), i.e. $l_{A_k(c)} = l_{A_{k-1}(c)} + 1$. The load of all other gateways remains the same (Lemma 3.1). Hence the resultant maximum load is $max[l_{A_k}(c), l^{k-1}_{A_{max}}]$.

We claim that there cannot exist another assignment for the set of sensors in T_k that will result in a lower maximum load. Suppose otherwise and let B_k be such an assignment for the sensors in T_k. Let $l^{k-1}_{B_{max}}$ denote the maximum load that results from the assignment of sensors from the set $T_k - \{t^k_k\}$ using assignment B_{k-1} . Suppose that sensor t^k_k is assigned to gateway w using assignment B_k. Then the maximum load of B_k is $l^k_{B_{max}} = max[l_{B_k}(w), l^{k-1}_{B_{max}}]$. By induction assumption, $l^{k-1}_{A_{max}} \leq l^{k-1}_{B_{max}}$. Hence, if $l^k_{B_{max}} < l^k_{A_{max}}$, then $l_{B_k}(w) < l_{A_k}(c)$. Since $l_{B_k}(w) = l_{B_{k-1}}(w) + 1$ and $l_{A_k}(c) = l_{A_{k-1}}(c) + 1$, $l_{B_{k-1}}(w) < l_{A_{k-1}}(c)$. But this contradicts the second optimality condition for A_{k-1}. Hence, A_k is an optimal assignment. By induction, A is therefore an optimal assignment. ■

Theorem 3.2 The time complexity of LBCA is $O(mn^2)$.

Proof: The initialization of the gateway load in step 1 can be done in $O(m)$. Each iteration of the for loop in step 2 deals with the construction of a BFS tree which is done within the while loop (step 3). Each sensor and gateway are inserted at most once into the queue Q. Hence there are at most $2(m+n)$ additions and removals of elements from Q. For each sensor i, $\mid C_i \mid \leq m$. Hence at most m gateways that will be inspected for each sensor that is removed from Q (in step 4). As there are at most n sensors that are inserted into Q, the total number of gateways that are inspected within the while loop in step 3 is $O(mn)$. Next we note that each removal of a gateway from Q results in the inspection of sensors that are assigned to it (step 5). Since each sensor is assigned to only one gateway, the total number of sensors that are inspected (due to the removal of the set of gateways from Q) within the while loop is $O(n)$. Hence each BFS tree can be constructed in $O(m+n+mn)$ and thus step 2 can be done in $O(n[m+n+mn])$. The reassignment of sensors to gateways in step 6 can be done in $O(m+n)$ (as there are at most $n+m$ elements in the path from $t_j$to v in the BFS tree rooted at t_j). Thus, the time complexity of the proposed algorithm is

$O(mn^2)$. ■

3.6 The Load-Balanced Clustering Problem (LBCP)

In this section, we consider the load-balanced clustering problem in which the traffic load from each sensor may differ from one another. We first analyze the computational complexity of the problem and prove that it is NP-hard. Having understood its computational complexity, we next propose an efficient approximation algorithm for the problem.

3.6.1 The intractability of LBCP

LBCP is related to the following machine scheduling problem.

Problem 3.1: Minimum Makespan Scheduling Problem on Identical Machines (MMSPIM)

We are given m machines and n jobs with respective processing times $p_1, p_2, \ldots, p_n \in Z^+$. The processing times are the same no matter on which machine a job is run and pre-emption is not allowed. Find an assignment of jobs to m identical machines such that the *makespan* (which is the latest completion time among all machines) is minimized.

Theorem 3.3 LBCP is NP-hard.

Proof: Consider a special case of LBCP whereby each sensor can be assigned to any gateways (i.e. no assignment constraints). It is easy to see that this special case of LBCP is identical to MMSPIM and LBCP is thus a generalization of MMSPIM. Since the MMSPIM is known to be NP-hard [22], LBCP is also NP-hard. ■

3.6.2 Some Observations about LBCP

In this section, we highlight some observations about LBCP.

Observation 1.

We first observe that LBCP is also related to another machine scheduling problem, namely the Minimum Makespan Scheduling Problem on Unrelated Machines (MMSPUM), which is defined as follows:

Problem 3.2: Minimum Makespan Scheduling Problem on Unrelated Machines (MMSPUM)

We are given a set J of n jobs and a set M of m machines. The processing time for a job $j \in J$ on machine $i \in M$ is $p_{ij} \in Z^+$ and pre-emption is not allowed. Find an assignment of jobs in J to the machines in M such that the *makespan* is minimized.

In particular, we note that each instance of LBCP can be transformed into an instance of the Minimum Makespan Scheduling Problem on Unrelated Machines (MMSPUM) whereby the sensors and the gateways of LBCP correspond to the jobs and machines of MMSPUM, respectively. For each sensor $t_i \in T$, let $p_{ij} = d_i \ \forall c_j \in C_i$ and let $p_{ij} = \infty \ \forall c_j \notin C_i$. Then it is easy to see that an optimal solution for LBCP corresponds to a schedule for MMSPUM with minimum makespan and vice versa. MMSPUM is also known to be NP-hard [22] and Lenstra *et al.* [80] gave a 2-approximation algorithm for the problem. This performance bound was further improved to $2 - \frac{1}{m}$ by Shchepin et al. [81] and this is currently the best-known approximation ratio that can be achieved in polynomial time.

Observation 2.

We next observe that each instance of LBCP can be represented using a bipartite graph as follows. Let $G = (T \cup C, E)$ denote a bipartite graph where E corresponds to a set of edges connecting the vertices in T to the vertices in C. An edge is said to exists between a pair of vertices (t_i, c_j) where $t_i \in T$ and $c_j \in C$ if $c_j \in C_i$. Let $q = \mid E \mid$ and let M be a maximum matching for G.

Lemma 3.2 The maximum number of gateways that may be used in any assignment of sensors in T to gateways in C is equal to $\mid M \mid$.

Proof: Let M be a maximum matching for the bipartite graph G. Let T_M and C_M

denote the set of matched vertices corresponding to sensors and gateways, respectively. It is easy to see that each sensor in T_M can be assigned to the corresponding gateway in C_M to which it is matched, thus utilizing $\mid M \mid$ gateways in this partial assignment. Next we argue that the sensors in $T - T_M$ can only be assigned to the gateways in C_M. This in turn implies that the maximum number of gateways that may be used in any assignment is equal to $\mid M \mid$.

The argument is as follows. Since M is a maximum matching, there does not exist any augmenting path in G with respect to M. Hence each path that begins with an unmatched vertex $t \in T - T_M$ must terminates at some matched vertex $t^* \in T_M$ and each of the vertices on this path are matched vertices. Hence each vertex $t \in T - T_M$ is only be adjacent to the vertices in C_M. This in turn implies that each vertex $t \in T - T_M$ can only be assigned to one of the vertices in C_M. Hence the maximum number of gateways that may be used in any assignment is equal to $\mid C_M \mid = \mid M \mid$. ■

Lemma 3.3 There exists an optimal assignment that uses exactly $\mid M \mid$ gateways.

Proof: Let X be an optimal assignment and $V_X \subseteq C$ denote the set of gateways utilized by X. Let $\mid V_X \mid = \delta$ and suppose that $\delta < |M|$. For each $v \in V_X$, let $U_v^X \subseteq T$ denote the set of sensors that are assigned to v (refer to Figure 3.2(a) and (b) for an illustration). We next construct a bipartite matching as follows. For each gateway $v \in V_X$, match v to a vertex, say u, where $u \in U_v^X$. Let the resultant matching be denoted by M' (refer to Figure 3.2(c) for an illustration). Clearly, $|M'| = \delta$. Since M' is not a maximum matching[1] in G, there must exists $|M|$ - δ augmenting paths[2] G with respect to M'. Each augmenting path in G will begin at some unmatched vertex $v \in C$ (a gateway with zero load) which is adjacent to a vertex u (corresponding to a sensor) which has been assigned to a gateway, say w, using X, i.e. $w \in V_x$. We consider the following two possibilities.

[1] A matching M on a graph G is a maximum matching if and only if there is no augmenting path in G with respect to M [82]

[2] An augmenting path with respect to M is one whose edges are alternately in M and not in M and the first and last vertices of the path are not incident to any edge of M.

Case (i): u is an unmatched vertex.

In this case, we will reassign u to v and removes it assignment to w. We note that there will be no increase in the overall maximum load following the reassignment. In addition, by matching u to v, the size of the bipartite matching will be increased by one.

Case (ii): u is matched to w.

Let the augmenting path that begins with v be denoted by $P_u = v - u - w - x$, where u, x and v, w denote sensors and gateways, respectively. In addition, we note that $u, x \in U_w^X$. Since we have matched u to w, x must be an unmatched vertex. Hence the gateway v may be assigned with a sensor as follows. Assign u to v and removes its assignment to w. Match u to v in G and removes its matching to w. We note that by reassigning u to v, there will not be any increase in the maximum load of the overall assignment. In addition, by matching x to w, the size of the matching (and the utilization of gateways) will be increased by one (refer to Figure 3.2(d) for an illustration). By repeating the above procedure for each augmenting path, we will establish a new optimal load-balanced assignment which utilizes exactly $\mid M \mid$ gateways. ■

3.6.3 An Approximation Algorithm

The algorithm proposed in [80] and [81] relies on solving a linear programming relaxation of the problem and uses the information obtained from the solution to allocate jobs to machines. In this section we present a new algorithm, called the Greedy Load-Balanced Clustering Algorithm (GLBCA), for LBCP that achieves a lower performance ratio than that of [81] without solving a linear program. In addition, our algorithm runs in $O(n[n+m+q])$ and is hence more efficient than the algorithm proposed in [81].

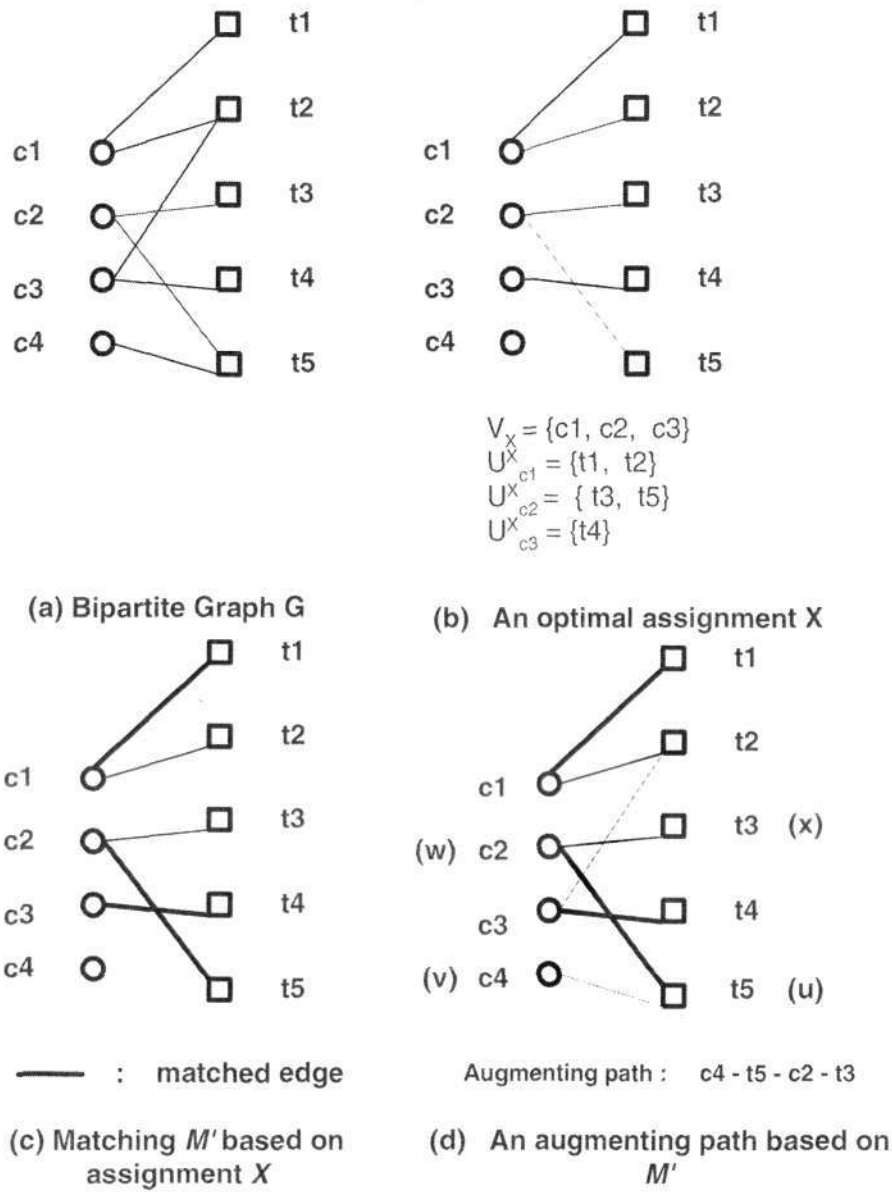


Figure 3.2: An Optimal Assignment using $|M|$ Gateways

3.6.4 The Greedy Load-Balanced Clustering Algorithm (GLBCA)

Our proposed algorithm adopts the approach of utilizing the maximum number of gateways possible in the assignment of sensors to gateways in order to distribute the traffic load among as many gateways as possible. Based on Lemma 3.2, we know that the maximum number

of gateways that may be used in any assignment is equal to $\mid M \mid$, where $\mid M \mid$ is the size of a maximum matching in the corresponding bipartite graph G. Hence our algorithm will attempt to find a maximum matching M in the graph G. We first sort the list of sensors in non-increasing order of traffic load. Let the resultant list be denoted by $T = \{t_1, t_2, \ldots, t_n\}$, where $d_1 \geq d_2 \geq \ldots \geq d_n$. Starting with the first sensor t_1 in the sorted list, we will attempt to match (or assign) t_1 to a gateway with zero load in the corresponding bipartite graph G. Next, the algorithm will proceed to match t_2 with another gateway with zero load in G. The algorithm will iterate in this manner where in each iteration, we will attempt to find an augmenting path P that connects a given sensor t_i to some unmatched gateway (with zero load) in G. If such a path is found , we will augment the edges in P which results in the assignment of t_i to some gateway in P and the reassignment of the other sensors to gateways in P. In addition the size of the resultant matching will be increased by one. If there does not exists any augmenting path that begins with t_i in G, then we will assign t_i to a least-loaded gateway in C_i.The algorithm terminates when all sensors have been assigned to some gateway. The pseudo code of the algorithm is given in Table 3.2.

Theorem 3.4 The time complexity of the proposed algorithm is $O(n[n + m + q])$.

Proof: The sorted list in step 1 can be done in $O(n \log n)$. The initialization of the load of gateways in step 2 can be done in $O(m)$. The while loop in step 3 will iterates n times. In step 3.1, each augmenting path can be found in $O(n + m + q)$ using breadth-first search. The augmentation of the edges in step 3.2 can be done in $O(n + m + q)$; the reassignment of sensors to gateways and computation of the new load in step 3.3 can be done in $O(n + m)$. In step 3.4, the assignment of a sensor to a least loaded gateway can be done in $O(m)$ and the computation of the resultant load can be done in $O(1)$. Hence step 3 can be completed in $O(n[n + m + q])$. Thus, the overall complexity of the algorithm is $O(n[n + m + q])$. ■

Table 3.2: Greedy Load-Balanced Clustering Algorithm

Begin

1. let T = list of sensors sorted in non-increasing order of traffic load;
2. **for** $j = 1$ to m **do**
set $l(j) = 0$;
endfor
set $i = 1$;
3. **while** $T \neq \emptyset$ do
3.1. find augmenting path P with t_i as one of its end vertex;
if P exists then
3.2. augment the edges in P;
3.3. **for** each matched edge (t_x, c_v) in P do
assign sensor t_x to gateway c_v;
let t_y be a sensor in P which was assigned to c_v prior to the augmentation of P;
$l(c_v) = l(c_v) + d_x - d_y$;
endfor
3.4. **else**
let c_j be gateway with the least load in C_i;
assign t_i to c_j;
$l(c_j) = l(c_j) + d_i$;
endif
$T = T - \{t_i\}$;
$i = i + 1$;
endwhile
End

3.6.5 Performance Ratio

Without loss of generality, we assume that the list of traffic loads $\{d_1, d_2, \ldots, d_n\}$ are all distinct. We will prove that our proposed algorithm is able to achieve a performance ratio of $\frac{3}{2}$ for LBCP.

Theorem 3.5 The Greedy Load-Balanced Clustering Algorithm (GLBCA) is a $\frac{3}{2}$-approximation algorithm for LBCP and this bound is tight.

Proof: Let W denote the sum of the traffic loads from all sensors, i.e. $W = \sum_{i=1}^{n} d_i$. Let OPT denote the maximum load of an optimal solution. Then it is clear that the sum of the traffic loads from all sensors is no more than $m \cdot OPT$. Hence $OPT \geq \frac{W}{m}$. In addition, it is easy to see that $OPT \geq d_i \forall i$.

Let I be an instance with the smallest number of sensors such that an assignment of I which is obtained using proposed algorithm has a maximum load $> OPT$. Let Φ denote this assignment. Let t_i be a sensor whose assignment to some gateway, say v^*, using Φ results in the overall maximum load of the assignment, i.e. $l(v^*) = \max l(v)\ \forall v \in C - \{v^*\}$, and $l(v^*) > OPT$. Suppose that $i \neq n$. Consider an instance I' which is equal to I without sensor t_n. Then I' is a smaller instance of I for which the proposed algorithm computes an assignment with maximum load $> OPT$. But this contradicts the choice of I. Hence we can assume that $i = n$. Note that this also implies that the load of each gateway prior to the assignment of sensor t_n to some gateway using Φ, is no more than OPT (otherwise we will again have a smaller problem instance with maximum load exceeding OPT).

Let A be an optimal assignment which uses the same number of gateways as Φ, i.e. $|M|$, (it follows from Lemma 3.3 that there exists such an optimal assignment). We first claim that $d_n \leq \frac{OPT}{2} - \epsilon$, where ϵ is a small positive constant. Suppose otherwise and assume that $d_n > \frac{OPT}{2} - \epsilon$. Since $d_i \geq d_n \forall i < n$, each gateway can be assigned with at most two sensors using A. Without loss of generality, we may assume that each gateway is assigned with two sensors (by adding dummy sensors with zero weight). We normalize the optimal assignment

A as follows:

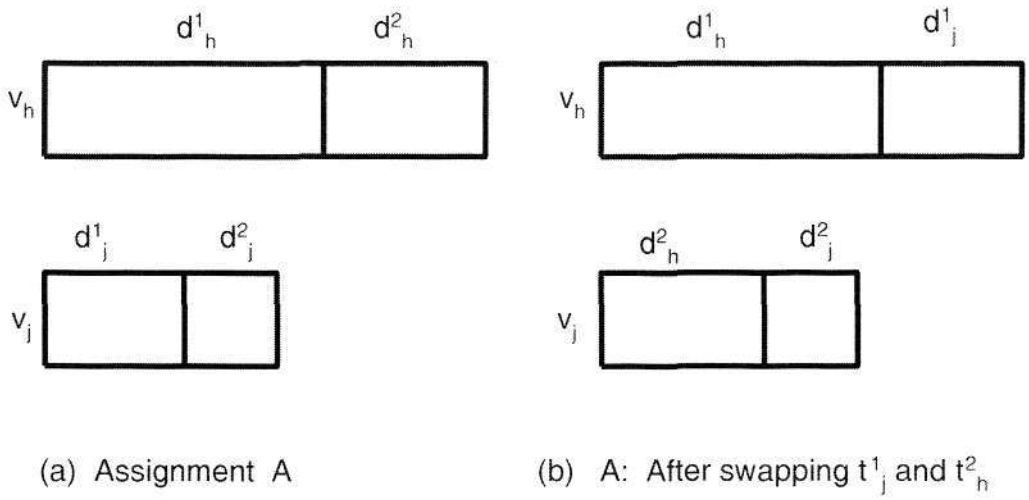
- for each gateway, place the sensor with the higher traffic load first
- sort the gateways so that the first sensors assigned are in descending order of traffic loads. Let the resultant set of gateways be denoted by $\{ v_1, v_2, \ldots, v_m \}$.

For each gateway v_q, let the first and second sensors assigned to v_q using A be denoted by t_q^1 and t_q^2, respectively. The corresponding traffic loads of t_q^1 and t_q^2 are denoted by d_q^1 and d_q^2, respectively. The assignment A may be further normalized as follows. Starting from $j = m$ *downto* 1, we compare the traffic load of t_j^1 with the traffic load of t_k^2 where $k < j$. Let t_h^2 be a sensor with highest traffic load among all sensors t_k^2, where $k < j$ which satisfies the following conditions, referred to as the *swapping conditions*:

- t_j^1 may be assigned to v_h
- t_h^2 may be assigned to v_j
- $d_h^2 > d_j^1$

We note that by interchanging the assignment of sensors t_h^2 and t_j^1, we will have sensors t_h^2 and t_j^2 assigned to v_j and sensors t_h^1 and t_j^1 assigned to v_h. Since $d_j^2 \leq d_h^1$, $d_h^2 + d_j^2 \leq d_h^2 + d_h^1 \leq OPT$. Similarly since $d_j^1 < d_h^2$, $d_h^1 + d_j^1 < d_h^2 + d_h^1 \leq OPT$. Hence, we can interchange the assignment of sensors t_h^2 and t_j^1 and yet keep an optimal assignment (refer to Figure 3.3 for an illustration). The resultant list of gateways (after considering all sensors t_j^1 for $j = m$ *downto* 1) is then sorted again so that the first sensors assigned are in descending order of traffic loads.

Following that, we will again check for the possibility of swapping the first sensor assigned to v_j with a second sensor assigned to another gateway which satisfy the above-mentioned swapping conditions for $j = m$ *downto* 1. If such a possibility exists, then the above-mentioned procedure is repeated. Otherwise, we next proceed the compare the traffic loads



(a) Assignment A (b) A: After swapping t^1_j and t^2_h

Figure 3.3: Further normalization of assignment A

of the second sensors assigned to the gateways. Starting from $j = m$ *downto* 1, we compare the traffic load of t^2_j with traffic load of t^2_k where $k < j$. Let t^2_h be a sensor with highest traffic load among all sensors t^2_k where $k < j$, which satisfies the following conditions:

- t^2_h may be assigned to v_j
- t^2_j may be assigned to v_h
- $d^2_h > d^2_j$

We note that by interchanging the assignment of sensors t^2_h and t^2_j, we will have sensors t^1_j and t^2_h assigned to gateway v_j and sensors t^1_h and t^2_j assigned to v_h. Since $d^2_j < d^2_h$, $d^1_h + d^2_j < d^1_h + d^2_h \leq OPT$. Similarly since $d^1_j \leq d^1_h$, $d^1_j + d^2_h \leq d^1_h + d^2_h \leq OPT$. Hence, we can interchange the assignment of sensors t^2_h and t^2_j and yet maintain an optimal assignment (refer to Figure 3.4 for an illustration). By iterating exchanges of this kind for $j = m$ *downto* 1, it is easy to see that our proposed algorithm gives an assignment that is equivalent to this. But this contradicts that the assumption that the maximum load of an assignment obtained by proposed algorithm, i.e. $\Phi > OPT$.

Hence $d_n \leq \frac{OPT}{2} - \epsilon$. As noted earlier, the load of each gateway prior to the assignment of t_n to some gateway using Φ, is no more than OPT. Hence, following the assignment of

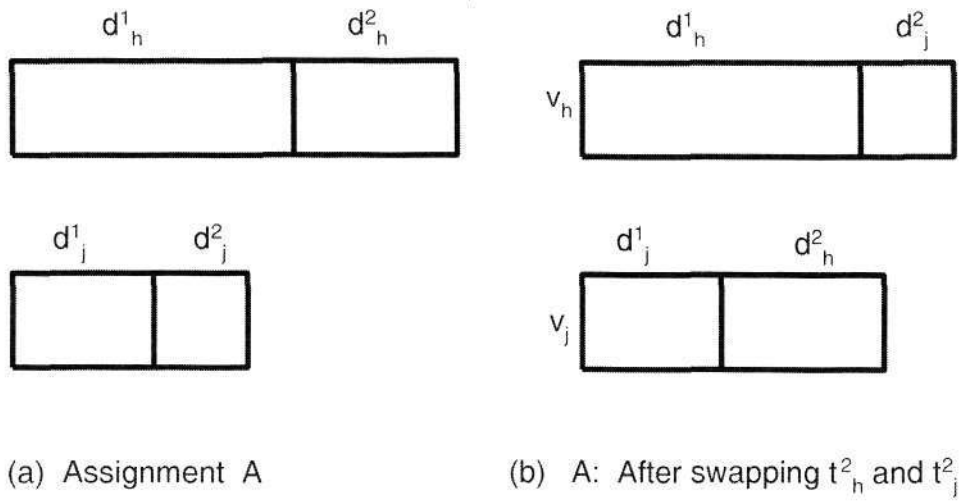


Figure 3.4: Swapping the assignment of the second sensors in A

t_n to some gateway by Φ, the overall maximum load $L \leq OPT + d_n \leq \frac{3}{2}OPT - \epsilon$.

We next show that the bound is tight using the following problem instance. Consider a problem instance whereby we have 2 gateways and 4 sensors with traffic loads of $5+\epsilon$, 5, $5-\epsilon$ and $5-\epsilon$, respectively. The assignment constraints of sensors to gateways are depicted in Figure 3.5. Using the proposed algorithm, sensor t_1 will be assigned to gateway c_1 while sensors t_2, t_3 and t_4 will be assigned to gateway c_2 giving an overall maximum load of $15-2\epsilon$. However an optimal assignment will assign t_1 and t_2 to c_1 and assign t_3 and t_4 to c_2 and the overall maximum load is $10+\epsilon$. The solution obtained by our algorithm is a factor 1.5 worse than the optimum.

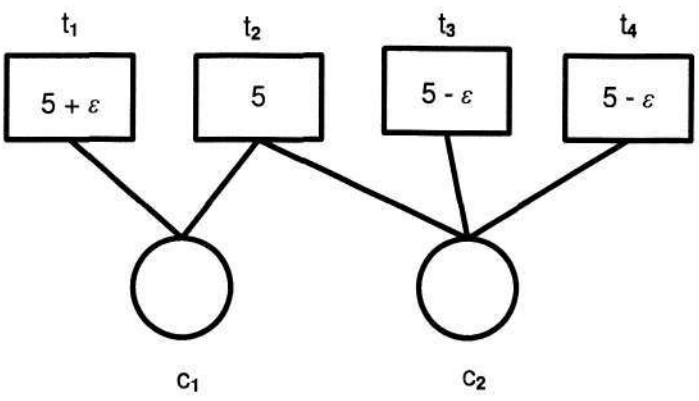


Figure 3.5: A Problem Instance: Performance Bound is Tight

■

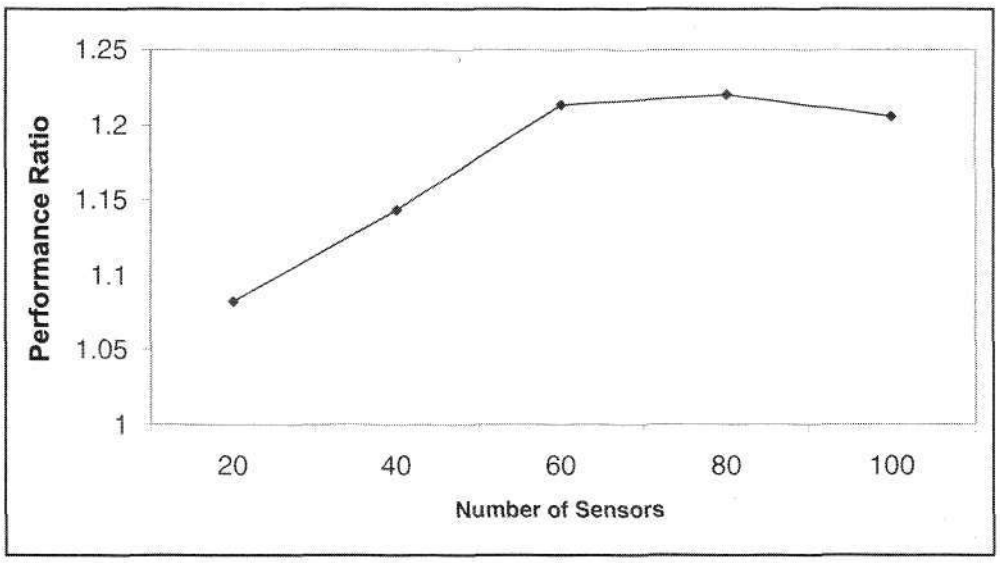


Figure 3.6: Performance ratio of GLBCA

3.7 Simulation Results

We study the performance of GLBCA by comparing its solutions with optimal solutions which are obtained by solving the ILP formulated program using CPLEX [83]. In our empirical studies, we consider the scenario of a sensor network whereby the gateways and sensors are generated randomly on a grid plane of 200×200, where each grid represents a 10×10meter square. The traffic load from each sensor is randomly selected from the range of 100 Kbps to 500 Kbps. We assume that a connection can be established between a gateway and a sensor if the distance between them is no larger than 550m.

Figure 3.6 shows the performance of GLBCA by varying the number of sensors (which ranges from 20 to 100) while fixing the number of gateways at 20. For each data point in Figure 3.6, 50 runs are taken and the average calculated. We observe that the solutions obtained using GLBCA differs from the optimal values by no more than 25%. In particular, we observe that the performance ratio of GLBCA ranges between 1.08 and 1.23 in all instances generated. This in turn implies that the average case performance of our proposed algorithm is much better than the worst-case performance ratio derived.

3.8 Summary

In this chapter, we address a problem that arise in the design of cluster-based wireless sensor networks. In particular, we consider the problem of assigning sensors to gateways in a wireless sensor network with the objective of distributing the traffic load among the gateways so as to ensure that no gateway is overloaded. We show that this problem is optimally solvable in polynomial time if all sensors have uniform traffic load. However, the problem turns out to be NP-hard if the sensors have differing traffic loads. We proposed an approximation algorithm for the NP-hard problem and prove that our proposed algorithm is able to guarantee a performance ratio of $\frac{3}{2}$. Empirical studies have shown that our proposed algorithm is able to perform even better on the average as compared to the worst-case performance ratio derived.

Chapter 4

Worst-Case and Best-Case k-coverage in Wireless Sensor Networks

In Chapter 3, we addressed the load-balancing issue in sensor networks, In this and next chapter, we will study the coverage issues of WSNs. Coverage is a fundamental problem in WSNs. Sensor coverage, which reflects how well a sensor network is monitored by sensors, is an important measure for the QoS that a sensor network can provide. In this chapter, we addressed the coverage problem from two different view points and refer to them as the worst-case and best-case coverage problems. Existing work on these two problems assumed that the coverage degree is one (i.e. the target area falls within the sensing range of at least one sensor). In this chapter, we address the k-coverage problem,where the coverage degree is a user-defined parameter k. This is a generalization of the earlier work where only k=1 is assumed. By combining geometric and algorithmic techniques, we establish optimal algorithms to solve the two variants of the k-coverage problem in polynomial time. Two important extensions of our study on the k-coverage problem were also proposed: a distributed algorithm for the problem and a solution for irregular coverage region. These two extensions help in applying the proposed algorithm under more practical scenarios.

4.1 Introduction

Sensor networks have been used for information collection,environmental monitoring and many other applications. A sensor network consists of a large number of low-cost sensor nodes which have sensing, storage, processing and communication capabilities. We assume that each sensor node knows its position through a low-power Global Positioning System (GPS) receiver. Since sensors may be randomly placed, one of the fundamental issues in a sensor network is the coverage problem. In general, coverage reflects how well an area is monitored or tracked by sensors and is thus one of the most important measures of the QoS that a sensor network can provide. In [84], Megerian *et al.* formulated the *1*-coverage problem by considering two extreme cases, namely worst-case coverage and best-case coverage. This formulation was adopted by some other existing works, such as [85–87]. Some existing work has also been done on the k-coverage problem problem [88–91] in which the target is covered by at least k sensors. We refer to this problem as one whose *coverage degree* is equal to k. The *k-coverage problem* is addressed for the following reasons:

1. Different applications require different degrees of sensing coverage. While some applications may only require that every location in a region be monitored by one sensor node, other applications require significantly higher degree of coverage. For example, distributed detection [92] requires every location to be monitored by multiple sensors, and distributed tracking and classification [93] requires even higher degrees of coverage.

2. The coverage requirements may vary depending on the number of faults that should be tolerated. A network with higher degree of coverage is more likely to maintain an acceptable coverage in the situation of higher nodes failure rates.

3. The coverage degree may also change after a network has been deployed. For example, a surveillance network may initially maintain a low degree of coverage for energy

conservation; after a certain event is detected, some of the sleeping sensors may be activated and the network would require higher degree of coverage for efficient distributed tracking.

Most of the existing work [84–87]in this area focus on the *1-coverage problem*, which means the target is covered by at least one sensor. In this work, we address the *k-coverage problem* from these two view points, namely worst-case k-coverage and best-case k-coverage. In the *worst-case k-coverage problem*, we are interested in finding a path connecting a point I and a point F which minimizes the maximum observability (detection probability); in the *best-case k-coverage problem*, we are interested in finding a path connecting a point I and a point F which maximizes the minimum observability. Although some existing work has been done on the k-coverage problem, to the best of our knowledge, these issues have not been sufficiently addressed by existing work.

The main contribution of this chapter are as follows:

- we first propose optimal, polynomial time algorithms for the worst-case and best-case of k-coverage problem.
- we next made two important extensions of our proposed algorithms:
  - In the first extension, the algorithm can be implemented in the case of irregular coverage region.
  - The second extension allows the algorithm to run in a distributed manner.

  With the introduction of these two extensions, our proposed algorithms can be applied under more practical scenarios.

The remainder of this chapter is organized as follows: In the next section, we review the related works. In Section 4.3, we give the formulation of the worst-case and best-case k-coverage problem and survey some techniques that are used in this study. The optimal

polynomial-time algorithms for both cases of the k-coverage problem are given and analyzed in Section 4.4. Section 4.5 gives two important extensions of our study proposed in Section 4.4. Some possible applications are discussed in Section 4.6. Section 4.7 gives the empirical results and the last section concludes the chapter.

4.2 Related Work

4.2.1 Optimal Covered Path

In [84], Megerian *et al.* used the term *agent* to refer to the phenomenon being detected by the sensors(for example, an enemy vehicle moving in the field). In order to evaluate the coverage of the sensor network, Megerian *et al.* formulated the coverage problem under two extreme cases, namely the worst-case coverage(maximum breach) problem and the best-case coverage(maximum support) problem. In the first problem, the objective is to find a path that stays as far away as possible from the sensors. This can be used to evaluate how well the sensors are placed and to determine the extent to which they can be breached. In the second problem, the objective is to find a path that stays as close as possible to the sensors. Both maximum breach and maximum support are metrics which can be used to evaluate the coverage of a sensor network and to guide the placement of additional sensors. Megerian *et al.* observed that an optimal solution for the maximum breach problem is a path lies along the edges of the Voronoi diagram [94, 95] and an optimal solution for the maximum support problem is a path lies along the edges of the Delaunay triangulation [94, 95]. Based on these facts, they proposed centralized optimal algorithms for both problems. Mehta *et al.* [87] improved on these algorithms and made them more computationally efficient. In [96], Adriaens *et al.* formulated the worst-case coverage problem in the case of each sensor has a directional "field of view"(coverage). A polynomial time algorithm for breach calculation was presented in [96] which solves the problem optimally.

Another work [86] of Megerian *et al.* studied a related problem of that was addressed

in [84]. In this work, Megerian *et al.* defined the term *exposure* to evaluate the coverage of a sensor network. Some properties of *exposure* are studied and an efficient and effective algorithm is proposed for exposure calculation in sensor networks, This algorithm can be used to specifically find minimal exposure paths.

Some works aimed at solving the problem formulated in [84] in a distributed manner. Li *et al.* [85] showed that the maximum support path can be constructed by using edges that belong to the relative neighborhood graph (RNG) of a sensor set. This is an improvement since the RNG is a subgraph of the Delaunay triangulation and can be constructed locally. In addition, a distributed algorithm based on RNG was proposed to solve the best-case coverage problem. On the other hand, Megerian *et al.* [84] have commented that a variation of the localized exposure algorithm presented in [97] can be used to solve the worse case coverage problem locally. Another localized algorithm with more practical assumptions was proposed by Huang *et al.* [98].

In [85] Li *et al.* attempted to address the problem of finding a shortest path of maximum support. The length of path computed by their algorithm is within a factor of 2.5 of the shortest path in the unit disk graph. In [87], Mehta *et al.* described an algorithm to find the geometric shortest path of maximum breach or maximum support.

4.2.2 k-coverage

By noticing that some applications require the coverage degree be more than one, Huang *et al.* [88] considered the k-coverage problem. In particular, Huang *et al.* studied the problem of determining if the area is sufficiently *k-covered*, in the sense that every point in the target area is covered by at least k sensors, where k is a predefined constant. In [88], this problem was formulated as a decision problem and a polynomial time algorithm which can be easily translated to distributed protocols was proposed. The extension of this problem to three-dimensional sensor networks was studied and solved in [89], by Huang *et al.*

The connected k-coverage problem was addressed in [90]. In [90], Zhou *et al.* considered

the problem of selecting a minimum set of sensors that are connected and also cover each point in a target region with at least k distinct sensors. By keeping only this set of sensors active to provide necessary coverage and connectivity, the energy can be conserved and the network is more fault-tolerant as compared to the *1*-covered network. Zhou *et al.* proposed a centralized greedy algorithm for this problem and proved that the algorithm is near-optimal (within a factor of $O(\lg n)$). A distributed algorithm has also been proposed in [90]. Xing *et al.* [91] explored the problem of energy conservation while maintaining both desired coverage degree and connectivity. An integrated study with coverage degree and connectivity was described and a flexible coverage configure protocol was proposed in [91].

Some studies focused on the relationship between the coverage degree k, the number of sensors n and the sensor coverage range r. Kumar *et al.* [99] considered the problem of determining the appropriate number of sensors that are enough to provide k-coverage of a region when sensors are allowed to sleep most of their lifetime. The critical conditions for the networks with both deterministic and random deployment were derived. Wan *et al.* [100] analyzed how the probability of the k-coverage changes with the sensing radius or the number of sensors while taking the boundary effect into account. An exact mathematical expression for the area that can be k-covered was formulated in [101] by Yen *et al.* and the results can be used to predict the coverage degree that a sensor network can provide.

Although the above mentioned works have signaled the importance and research interests of k-coverage, to the best of our knowledge, [87] is the only work which address the problem of finding the worst-case k-covered path. In [87], Mehta *et al.* suggested that the worst-case k-coverage problem can be addressed by adopting the k-th nearest Voronoi diagram. However, no details of the proposed algorithm were given. Based on our analysis, we found that the properties of k-th nearest Voronoi diagram are quite different from the ordinary Voronoi diagram. Thus there may not exist an optimal solution that lies along the edges of the k-th nearest Voronoi diagram.

4.3 Preliminaries

4.3.1 Problem Formulation

As in [84], we will formulate the *k-coverage problem* under two scenarios, namely worst-case coverage and best-case coverage. We assume that the wireless sensor nodes are given as a set S with n points which are distributed inside a continuous two-dimensional field Ω, where for each sensor $s_i \in S$, its location (x_i, y_i) is known. A pair of points I and F are given as initial point and final point, separately. We also assume that every sensor node has the same maximum transmission(communication) range, denoted by R. Hence, the set of wireless sensors S defines a unit disk graph and we assume that the graph is always connected.

Observation shows that the signal strength of a given sensor s_i diminishes as the distance increases. This phenomenon is referred to as path loss(or path attenuation). We say a target point poi_j is *covered* by s_i if the path loss from s_i to poi_j is no larger than a predefined threshold p. For the simplicity, we assume the path loss from a sensor s_i to target point poi_j is only determined by the Euclidean distance between s_i and poi_j. With this assumption, the region which is covered by s_i is a disk which center at s_i. The radius r of this disk is referred to as the *sensing radius* of s_i.

Definition 4.1: *k-th distance.* Given a positive integer k ($k \leq n$) and a point $poi_j = (x_j, y_j)$ in the field Ω, the *k-th distance* of poi_j, denoted by d_{jk}, is defined as the Euclidean distance from poi_j to the *k-th* nearest sensor of poi_j.

The reason why we are interested in k-th distance is that the k-th distance of a point poi_j serves as the critical value of the sensing radius in order to ensure that point poi_j is *k-covered*. We note that given a point poi_j with k-th distance d_{jk}, there are exactly k sensors within the distance d_{jk} of poi_j. It follows that as the k-th distance of a point increases, the sensing radius of each sensor will also need to be increased to ensure that this point is k-covered. Thus, the *k-coverage* of a point can be evaluated by the k-th distance of this point.

As an example illustrated in Figure 4.1, sensor s_a is the nearest sensor of target point poi_M, sensor s_b is the second nearest sensor of poi_M and sensor s_c is the third nearest sensor of poi_M. According to Definition 4.1, the distance between sensor s_a and point poi_M, denoted by $|Ms_a|$, is the first distance of poi_M. Similarly, $|Ms_b|$ is the second distance of poi_M and $|Ms_c|$ is the third distance of poi_M. We can note that, there are exactly three sensors, s_a, s_b and s_c within the distance $|Ms_c|$ of poi_M. This indicates that poi_M will falls in the sensing radius of exactly three sensors if and only if the sensing radius of each sensor is no less than $|Ms_c|$.

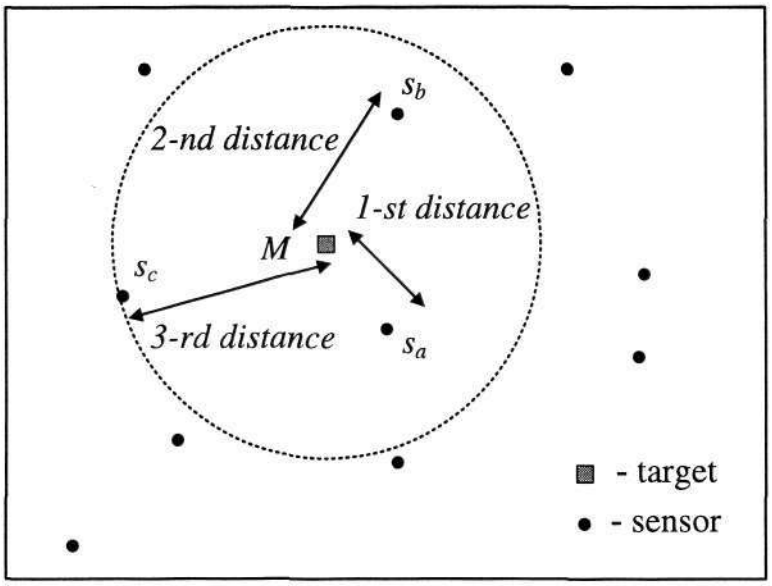


Figure 4.1: The definition of k-th distance

Definition 4.2: ***k-breach.*** Given a path P which is connecting initial point I and final point F, the *k-breach* of P, denoted by B_{Pk}, is defined as the minimum *k-th distance* experienced by an agent traversing along P.

Definition 4.3: ***k-support.*** Given a path P which is connecting initial point I and finial point F, the *k-support* of P, denoted by S_{Pk}, is defined as the maximal *k-th distance* experienced by an agent traversing along P.

The k-breach and k-support are two metrics that can be used to evaluate the *k-coverage of a path* from two viewpoints, namely worst-case coverage and best-case coverage.

Thus the *worst-case k-coverage problem*, or the *maximal k-breach path problem*, can be formally stated as follows:

Maximal k-breach Path Problem. Identify a path P_{Bk} in the field Ω connecting initial point I and final point F which maximizes the *k-breach* of P_{Bk}.

Similarly, the *best-case k-coverage problem*, or the *minimum k-support path problem* can be stated as:

Minimum k-support Path Problem. Identify a path P_{Sk} in the field Ω connecting initial point I and finial point F that minimizes the *k-support* of P_{Sk}.

In this Chapter, we evaluate how well a target field is k-covered by a set of sensors, which we referred to as the *k-coverage quality* of the network. The *k-coverage quality* of a network is evaluated from two viewpoints: the maximal k-breach and the minimum k-support.

The case when $k = 1$ for both the problems has been considered in [84]. We note that in [84], the best-case coverage problem is also referred to as the *maximum support path problem*. We find that this notion is misleading in some sense because the objective of the problem is to find a path which minimizes the support(defined as the maximum Euclidean distance from the path P to the nearest sensor). Thus in this article we refer to this problem as the "*minimum k-support path problem*".

4.3.2 Growing Disks

In [85], Li *et al.* introduced the *growing disks technique* which can be used to solve the *1*-coverage problem. Assume that every sensor node originally has a disk centered at it with radius 0 and every disk starts growing with the same speed. Let $D(S, r)$ be the region covered by at least one disk where each of these disks is centered at some sensors from S and with radius r. Let $\overline{D(S, r)}$ be the complementary region of $D(S, r)$ in the field Ω. Then the worst-case *1*-coverage problem seeks to determine the largest radius r such that there is a path, inside the region $\overline{D(S, r)}$, connecting points I and F. On the other hand, the best-case *1*-coverage problem aims to find the smallest radius r such that there is a path, inside the

region $D(S, r)$, connecting points I and F.

We extend the idea of growing disks in the study of the k-coverage problem. Let $D(S, k, r)$ denote the region covered by at least k disks. It is easy to see that the region $D(S, k, r)$ will decrease as k increases. In addition, $D(S, k, r) \subseteq D(S, r)$ $\forall k \geq 1$. Let $\overline{D(S, k, r)}$ be the complementary region of $D(S, k, r)$ in the field Ω. The following two theorems show that we can adapt the growing disks technique to solve the *k-coverage problem* for arbitrary k:

Theorem 4.1. $B_{Pk} > r$ if and only if $P \in \overline{D(S, k, r)}$.

Proof: "If": From the definition of $\overline{D(S, k, r)}$ we can note that a given point $poi_j = (x_j, y_j)$ in $\overline{D(S, k, r)}$ is covered by at most *k-1* disks. This means point poi_j is within the distance r of at most *k-1* sensors which in turn implies that the *k-th* distance of poi_j is larger than r. Thus for any path P in the region $\overline{D(S, k, r)}$, we have $B_{Pk} > r$.

"Only if": If there is a point $poi_j \in P$ that falls within the region $D(S, k, r)$, then poi_j will be covered by at least k disks. This means that poi_j is within the distance r of at least k sensors which in turn implies that $d_{jk} \leq r$. Hence we have $B_{Pk} \leq d_{jk} \leq r$.

■

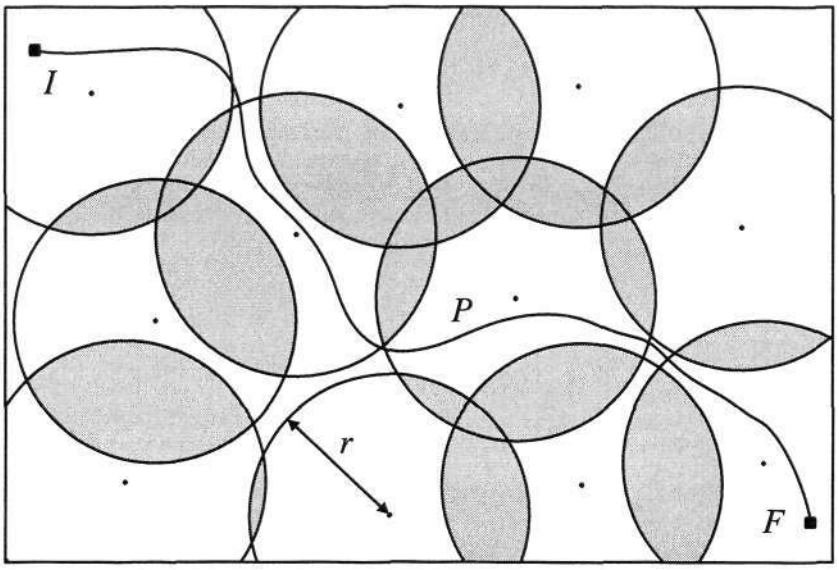


Figure 4.2: A path P with *2*-breach larger than r

Theorem 4.1 shows that we can obtain a path P with k-breach larger than r if P stays

outside the region $D(S,k,r)$. Figure 4.2 gives an example for the case of $k = 2$. The region $D(S,2,r)$ is shown in the shaded area.

Theorem 4.2. $S_{Pk} \leq r$ if and only if $P \in D(S,k,r)$.

Proof: "If": From the definition of $D(S,k,r)$, any point $poi_j = (x_j, y_j)$ in the region $D(S,k,r)$ is covered by at least k disks which in turn implies that $d_{jk} \leq r$. Thus we have $S_{Pk} \leq r$.

"Only if": If there is a point $poi_j \in P$ falls in the region $\overline{D(S,k,r)}$, then $d_{jk} > r$. Thus we will have $S_{Pk} \geq d_{jk} > r$.

■

As the radius r of each disk increases, the region $D(S,k,r)$ increases and $\overline{D(S,k,r)}$ decreases. It follows from Theorem 4.1 that the *worst-case k-coverage problem* is equivalent to that of finding the largest radius r such that there is a path P, inside the region $\overline{D(S,k,r)}$, connecting points I and F. Similarly, it follows from Theorem 4.2 that the *best-case k-coverage problem* is equivalent to that of finding the smallest radius r such that there is a path P, inside the region $D(S,k,r)$, connecting points I and F. These two problems can be solved by adopting binary search if we can solve the following two decision problems:

Problem 4.1: *Worst-case k-coverage Problem.* Given a field Ω, the sensor set S, the value of k and r, the initial point I and final point F, can we establish a path connecting I and F in the region $\overline{D(S,k,r)}$?

Problem 4.2: *Best-case k-coverage Problem.* Given a field Ω, the sensor set S, the value of k and r, the initial point I and final point F, can we establish a path connecting I and F in the region $D(S,k,r)$?

4.4 Proposed Solution

4.4.1 Segments and Borders

We adopt the notion of *disk diagram* in the study of k-coverage problem. Given a field Ω and the sensor set S, let C denote the set of n disks of radius r, where for each disk $dis_i \in C$, it is centered at $s_i \in S$. The combination of Ω and C, denoted by $\Omega \cup C$, is referred to as the *disk diagram*.

The perimeters of the disks in C will partition the field Ω into a set of *segments*, whereby the *borders* of each segment are comprised of (i) a set of arcs or (ii) a set of arcs and some line segments(which are referred to as boundary lines) which define the boundary of the field (see Figure 4.3 for an illustration). We note that each segment either corresponds to (a) some intersection region of a set of disks and the field or (b) a part of the field that is not covered by any disk.

Given: (i) the perimeters of two disks or (ii) a line segment which defines the boundary of Ω and the perimeter of a disk, they may intersect at (a) two points, which are referred to as *intersection points* or (b) one point, which is referred to as *tangential point*. Given two points, which may be intersection point(s) or tangential point(s), we say they are *adjacent* if (i) both points belong to the same perimeter and no other intersection or tangential point falls between them or (ii) both points belong to line segments which define the boundary of Ω and no other intersection or tangential point falls between them. It is easy to see that the end points of each border is a pair of adjacent intersection(or tangential) points.

As an example shown in Figure 4.3, dis_A and dis_B intersect at two points: poi_N and poi_Q, dis_A and dis_C intersect at one point: poi_M, thus poi_N and poi_Q are two intersection points and poi_M is a tangential point. Both poi_N and poi_Q belong to the perimeter of dis_A (or dis_B) and no other intersection or tangential point falls between them, thus poi_N and poi_Q are adjacent to each other and the two arcs fall between them are two borders.

Definition 4.4: Coverage Degree of Segment. Let seg_i be a segment which is obtained

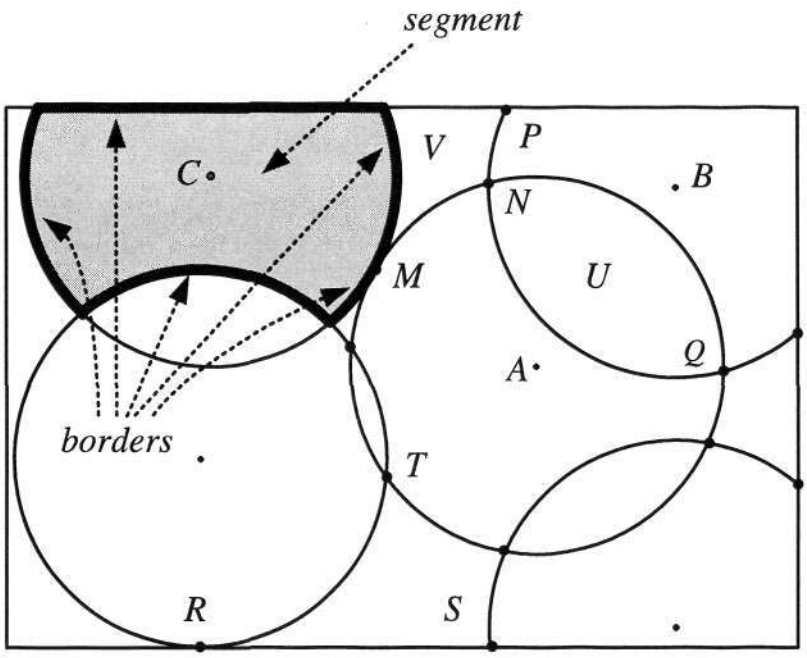


Figure 4.3: Segments and their borders

from the intersection of a set of w disks in the field Ω, where $0 \leq w \leq n$. We say that the coverage degree of segment seg_i, denoted by $deg(seg_i)$, is equal to w.

As shown in Figure 4.3, seg_U is the intersecting region of two disks: dis_A and dis_B, seg_V is not covered by any disk, thus $deg(seg_U) = 2$, $deg(seg_V) = 0$.

Since a segment is the smallest intersecting region of disks and the field, all points in a segment are covered by same set of disks.

Definition 4.5: Coverage Degree of Border. Given a border $bor_i \in seg_j$, the *coverage degree of* bor_i, denoted by $deg(bor_i)$, is defined as: $deg(bor_i) = deg(seg_j)$.

However, this definition will give a rise to the conflict when calculating the coverage degree of an arc since an arc is the border of two neighboring segments. In order to resolve this conflict, we make following definition:

Definition 4.6: Inner Arc and Outer Arc. Given an arc arc_i which lies on the perimeter of a disk dis_j, we can divide arc_i into two separate arcs, namely the *inner arc* and the *outer arc*. The inner arc will belong to the segment which is inside dis_j whereas the outer arc will

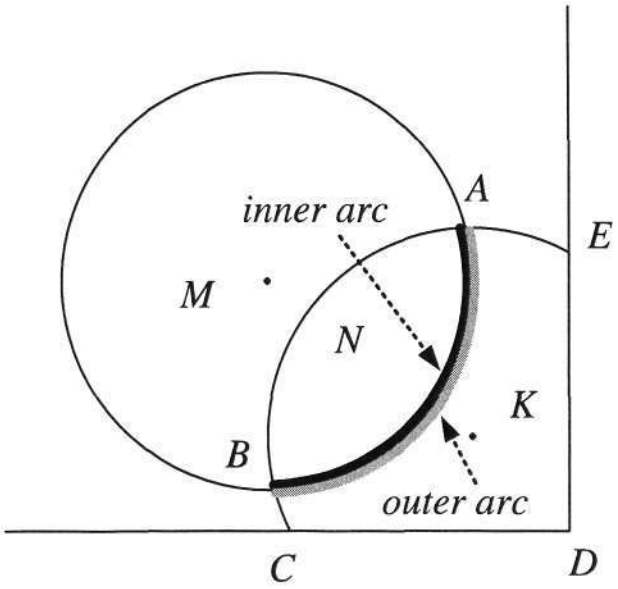


Figure 4.4: Inner arc and outer arc

belong to the segment which is outside dis_j.

As an example shown in Figure 4.4, an original arc $\widehat{AB}$ is divided into inner arc arc_{AB_inner} (shown in dark line) and outer arc arc_{AB_outer} (shown in light line), from Definition 4.4-4.6 we have:

$arc_{AB\_inner} \in seg_N$, $arc_{AB\_outer} \in seg_K$;

$deg(arc_{AB\_inner}) = deg(seg_N) = 2$;

$deg(arc_{AB\_outer}) = deg(seg_K) = 1$.

In general, for an arbitrary arc $\widehat{O}$:

$deg(arc_{\widehat{O}\_inner}) = deg(arc_{\widehat{O}\_outer}) + 1$.

Definition 4.7: Adjacent Borders. We say two borders are *adjacent* to each other if they share at least one common point.

Based on this definition, the inner arc and the outer arc of the same arc are adjacent to each other since they have infinite number of common points. In addition, it is easy to see that two borders are adjacent to each other if and only if they share a common intersection or tangential point. As an illustration, we refer to Figure 4.3, whereby borders $arc_{MN-inner}$ and $arc_{PN-outer}$ are adjacent borders which share a common intersection point poi_N; borders

$arc_{TR-inner}$ and SR are adjacent borders which share a common tangential point poi_R.

We note the borders of a segment can be seen as a series of adjacent borders and all these borders have same coverage degree(same to the coverage degree of the segment they belong to). For example, as shown in Figure 4.4, seg_K is defined by four borders: $arc_{AB-outer}$, $arc_{AE-inner}$, $ED\text{-}DC$, $arc_{BC-inner}$. From Definition 4.4-4.7 we have:

$arc_{AB-outer}$ is adjacent to $arc_{AE-inner}$, $arc_{AE-inner}$ is adjacent to $ED\text{-}DC$, $ED\text{-}DC$ is adjacent to $arc_{BC-inner}$, $arc_{BC-inner}$ is adjacent to $arc_{AB-outer}$;
$deg(arc_{AB-outer}) = deg(arc_{AE-inner}) = deg(ED\text{-}DC) = deg(arc_{BC-inner}) = deg(seg_K) = 1$.

Definition 4.8: Adjacent Segments. We say two segments are *adjacent* to each other if there exists a pair of borders bor_A and bor_B, where $bor_A \in seg_A$ and $bor_B \in seg_B$ such that bor_A and bor_B are adjacent to one another.

From Definition 4.8, it follows that an agent can traverse from seg_A to seg_B directly if and only if seg_A and seg_B are adjacent to each other.

We note that if a border is an arc, it can be defined by using four parameters, namely the center of disk, radius of disk, and the two angles that define the two extreme ends of the arc; on the other hand, if a border is a part of some boundary line(s), it can be defined by the boundary line(s) it belongs to and two extreme ends. Given the center of disk, the radius of disk and the boundary lines, the borders can be defined by computing the end points of all borders. We will illustrate how to compute these end points by the example shown in Figure 4.5:

Case i. The border is an arc. In the example shown in Figure 4.5, border $arc_{T-inner}$ and $arc_{T-outer}$ can be defined by $\angle FQJ$ and $\angle GQJ$. Given the center of two intersecting disks: $P = (x_P, y_P), Q = (x_Q, y_Q)$, $\angle FQJ$ and $\angle GQJ$ can be computed as:

$$\begin{aligned} &\angle FQJ = \angle\varphi + \angle\omega \\ &= arctan(\tfrac{y_P - y_Q}{x_P - x_Q}) + arccos(\tfrac{|EQ|}{|FQ|}) \\ &= arctan(\tfrac{y_P - y_Q}{x_P - x_Q}) + arccos(\tfrac{\frac{1}{2}\sqrt{(x_P - x_Q)^2 + (y_P - y_Q)^2}}{r}) \end{aligned}$$

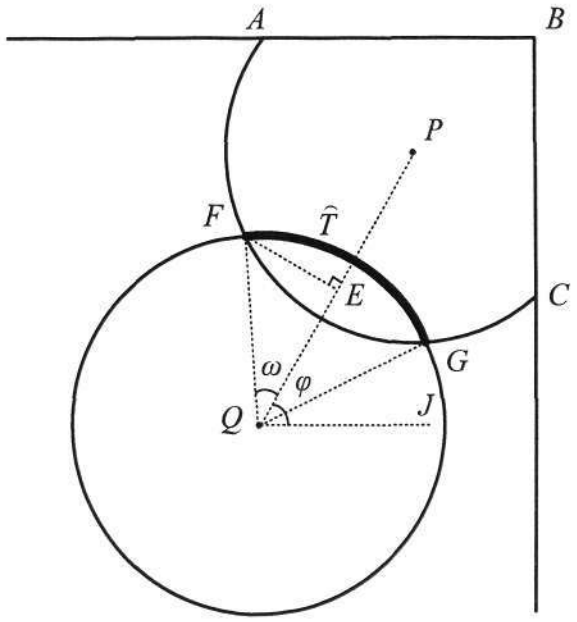


Figure 4.5: Calculating the end points of borders

$$\angle GQJ = \angle\varphi - \angle\omega$$
$$= arctan(\tfrac{y_P - y_Q}{x_P - x_Q}) - arccos(\tfrac{|EQ|}{|FQ|})$$
$$= arctan(\tfrac{y_P - y_Q}{x_P - x_Q}) - arccos(\tfrac{\frac{1}{2}\sqrt{(x_P - x_Q)^2 + (y_P - y_Q)^2}}{r})$$

Case ii. The border is a part of some boundary line(s). In this case, each end point of the border is a intersection or tangential point of disk perimeter and some boundary line(s). Given the center and the radius of a disk, its perimeter can be represented by the equation for a circle. Thus the intersection point or tangential point can be computed by combining the equation of the perimeter and the equation of the boundary line. As the example shown in Figure 4.5, for border AB-BC, one of its end points $poi_A = (x_A, y_A)$ can be computed by combining following equations:

$(x_A - x_P)^2 + (y_A - y_P)^2 = r^2$ (equation for the perimeter of $disk_P$)

$y_A = Y$ (equation of boundary line AB)

with the constraint: $x_A \leq X$

Similarly, another end point $C = (x_C, y_C)$ can be computed by combining following equations:

$(x_C - x_C)^2 + (y_C - y_C)^2 = r^2$ (equation for the perimeter of $disk_P$)

$x_C = X$ (equation of boundary line BC)

with the constraint: $y_C \leq Y$

The process of calculating all borders is described by the pseudo code in Table 4.1.

Lemma 4.1. The computational complexity of the border calculate procedure is $O(n^3)$.

Proof: There are n disks, each of which may intersect with or is tangential to the other $n-1$ disks and a set of line segments which define the boundary of target field. Hence each disk will has $O(n)$ intersection points and/or tangential points. Each border will be defined by a pair of neighboring intersection points or tangential points. To enumerate all pairs of neighboring intersection points or tangential points, these points should be sorted in clockwise or counterclockwise manner. This will cost $O(n \lg n)$ for each sensor and $O(n^2 \lg n)$ for all n sensors. $O(n)$ borders will be defined for each sensor and totally we have $O(n^2)$ borders. To calculate the coverage degree of each border, we have to look up all n sensors to determine the number of sensors which cover this border. This operation will costs $O(n)$ for each border an $O(n^3)$ for all n^2 borders. Hence the time complexity for the computation of all borders is $O(n^3)$. ■

4.4.2 Worst-Case k-coverage Problem

We next consider Problem 4.1. We note that the region $\overline{D(S,k,r)}$ may consists of many segments, each of which has coverage degree less than k. When an agent is traversing in the region $\overline{D(S,k,r)}$, it may travel from one segment to another if the two segments are adjacent and have coverage degree less than k. Let seg_I denote the segment where initial point I falls in and seg_F denote the segment where final point F falls in. Problem 4.1 asks if it exists a series of segments $seg_1, seg_2, ..., seg_m$, whereby the following conditions hold:

i). $seg_1 = seg_I, seg_m = seg_F$;

ii). seg_i and seg_{i+1} are adjacent to each other for $i = 1, 2, ..., m-1$;

Table 4.1: The calculation of all borders

Procedure: **border-calculate** /*This procedure calculate all borders and determine whether two borders are adjacent to each other.

input: sensor set $S = \{s_1, s_2, ..., s_n\}$, disk radius$r$.

output: border set $\{bor_1, bor_2, ..., bor_k\}$.

Begin
for $i = 1$ to $i = n$
    calculate the intersection or tangential points of $disk_i$ with all other disks or boundary lines
endfor
for each pair of adjacent intersection or tangential point which lie on the perimeter of one disk
    a border bor_j is **defined**
    split bor_j into a pair of inner arc and outer arc
endfor
for each pair of adjacent intersection or tangential point which lie on the boundary lines of the field
    a border bor_i is **defined**
endfor
/*Now calculate the coverage degree of each border
for each border bor_j
    switch border type
    case bor_j is a part of some boundary line(s)
        pick a point $poi_j \in bor_j$
        calculate the number of sensors within distance r of poi_j
    case bor_j is an outer arc
        pick a point $poi_i \in bor_j$
        calculate the number of sensors within distance r of poi_j, excluding the center of the disk poi_j belong to
    case bor_j is an inner arc
        pick a point $poi_i \in bor_j$
        calculate the number of sensors within distance r of poi_j, including the center of the disk poi_j belong to
    endswitch
endfor /*Now determine whether a pair of borders are adjacent
set all borders non-adjacent
for each intersection or tangential point poi_i
    if poi_i is an end point of bor_A and bor_B
        bor_A and bor_B are **set** to adjacent to each other
    endif
endfor
End

iii). $deg(seg_i) < k$ for $i = 1, 2, ..., m$.

From Definitions 4.7 and 4.8, we can transform Problem 4.1 into one of finding a series of adjacent borders which can be stated as follows:

Problem 4.3 Does there exist a series of borders bor_1, bor_2, ..., bor_r, whereby the following conditions hold?

i). $bor_1 \in seg_I, bor_r \in seg_F$;

ii). bor_i and bor_{i+1} are adjacent to each other for $i = 1, 2, ..., r-1$;

iii). $deg(bor_i) < k$ for $i = 1, 2, ..., r$.

Given the set of borders which defines all segments in the field Ω, Problem 4.1 can be solved by executing BFS. This search process can be described by the pseudo code shown in Table 4.2.

Figure 4.6 gives an example of the search process. As shown in Figure 4.6.I, this search process starts at AB-BC, which is one of the borders that belongs to seg_I(the segment where initial point I falls in). In Figure 4.6.II, the borders which are adjacent to AB-BC and have coverage degree less than k will be included in the search tree and be marked as explored(highlighted in bold). This process continues to explore unmarked borders, as illustrated in Figure 4.6.III to Figure 4.6.V. After this BFS search is complete (Figure 4.6.V), we will pick one of the borders which belongs to seg_F(the segment where final point F falls in) and check whether it is marked. If it is marked, that indicates the answer of Problem 4.3 is "true", otherwise the answer is "false".

The maximum value of r such that there is a path, inside the region $\overline{D(S, k, r)}$, connecting points I and F can be determined by binary search. Initially, we can set the upper bound and lower bound of search space by considering the dimension of the field and the density of sensors. In each iteration, r will be calculated as the average of upper bound and lower bound. If the answer for Problem 4.3 is "true", then the lower bound will be replaced by r;

Table 4.2: The BFS of borders

Procedure: connection /*This procedure determines if there is a path connecting I and F in the region $\overline{D(S,k,r)}$.
input: borders set $\{bor_1, bor_2, ..., bor_k\}$, $bor_I \in seg_I$, $bor_F \in seg_F$.
output: boolean variable *result*.

Begin
if $deg(bor_I) \geq k$ **or** $deg(bor_F) \geq k$
    $result = 0$
    return
endif
unmark all borders
mark bor_I /*start with bor_I
list $L = bor_I$
tree $T = bor_I$
while L nonempty
    choose some border bor_v from front of list L
    delete bor_v from list L
    for each border bor_w which is adjacent to bor_v
        if $deg(bor_w) < k$
            mark bor_w
            include bor_w in tree T as a child of bor_v
            add bor_w to end of list L
        endif
    endfor
endwhile
if bor_F is marked
    $result = 1$
    return
else
    $result = 0$
    return
endif
End

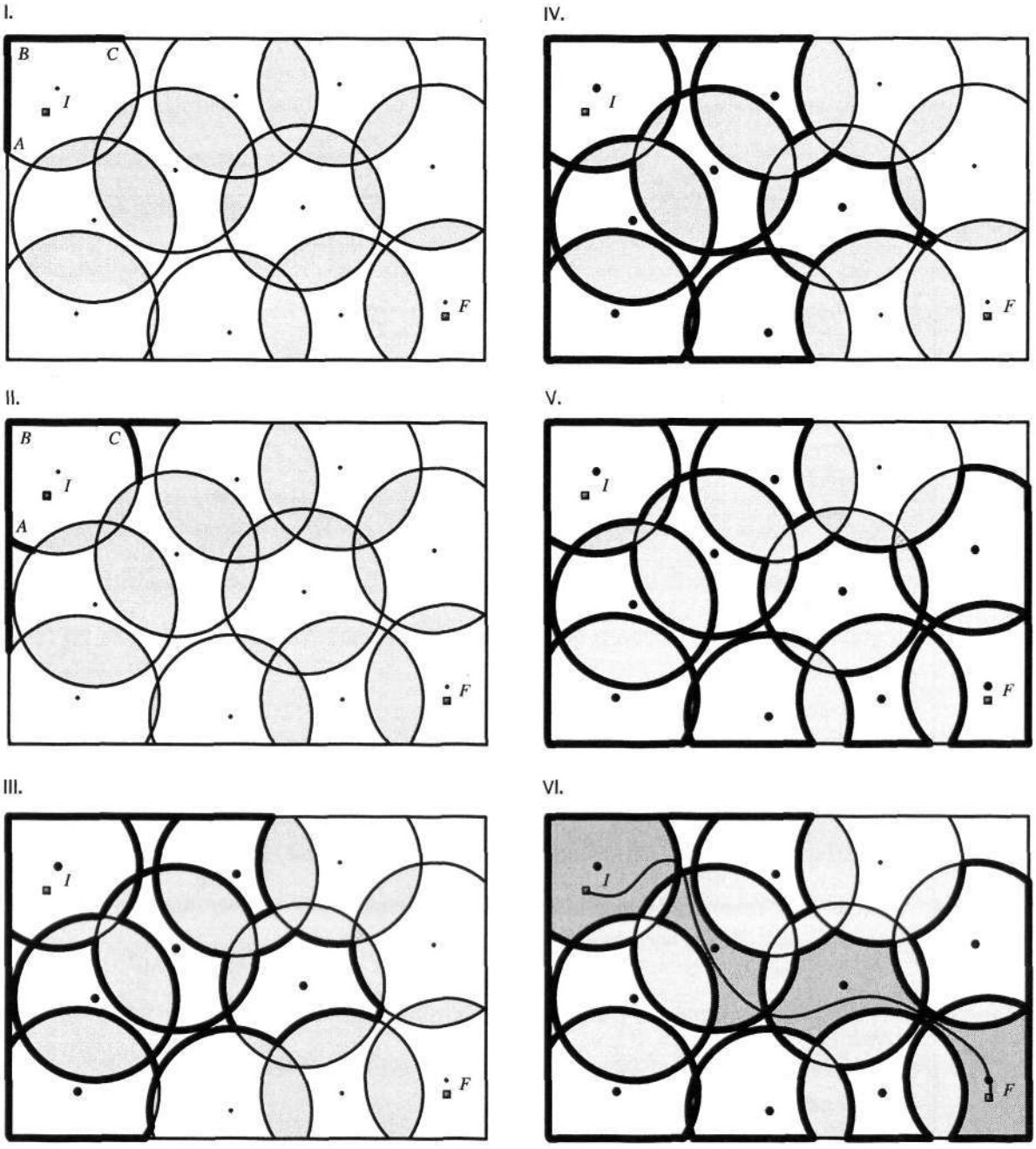


Figure 4.6: Exploration process of borders

otherwise the upper bound will be replaced by r. The process of the search can be described by the pseudo code shown in table 4.3.

This binary search will terminates when the maximum number of iterations is reached. The number of iterations can be set based on the requirements of accuracy. We can note

Table 4.3: The binary search procedure

Procedure: connection /*This procedure determines if there is a path connecting I and F in the region $\overline{D(S,k,r)}$.
Begin
set *upper bound* /* the upper bound of search space
set *lower bound* /* the lower bound of search space
set *iteration* /* the number of iterations
for $i = 1$ to *iteration*
    $r = (upperbound + lowerbound)/2$
    $\{bor_1, bor_2, ..., bor_k\}$=**border-calculate**$(S, r)$/*calculate all borders
    $result$ =**connection**$(\{bor_1, bor_2, ..., bor_k\}, bor_I, bor_F)$
    if $result = 1$
        $lowerbound = r$
    else
        $upperbound = r$
    endif
endfor
return r
End

that each iteration will reduce the search space by half. If the search space ($|upperbound - lowerbound|$) is 1000, 10 iterations would reduce the search space to below 1. Thus in most practical cases, $10 \sim 20$ iterations are enough to guarantee that the estimated error is negligible.

The maximal breach path can be obtained from the BFS tree constructed using the procedure *connection* if $result = 1$. In particular, the border bor_F which belongs to seg_F would has been included in the BFS tree. Hence the sequence of adjacent borders from the root(the border bor_I) to the leaf (bor_F) can also be obtained. This sequence of borders can be easily restored to the sequence of segments. For example, consider a sequence of borders which is obtained in the BFS tree: $bor_1, bor_2, ..., bor_r$, where $bor_1 \in seg_I$, $bor_r \in seg_F$. For each border bor_i, we will replace it by the segment seg_i, where $bor_i \in seg_i$. After all borders are replaced by segments, we will obtain a sequence of adjacent segments: $seg_1, seg_2, ..., seg_r$, where $seg_1 = seg_I$, $seg_r = seg_F$. All paths traversing along this sequence of segments will

have the *k-breach* larger than r(refer Figure 4.6.VI for an illustration).

4.4.3 Best-Case k-coverage Problem

Similar to the *worst-case k-coverage problem*, Problem 4.2 can be transformed into the following problem:

Problem 4.4 Does there exist a series of borders bor_1, $bor_2, ..., bor_r$, whereby the following conditions hold:

i). $bor_1 \in seg_I, bor_r \in seg_F$;

ii). bor_i and bor_{i+1} are adjacent to each other for $i = 1, 2, ..., r-1$;

iii). $deg(bor_i) \geq k$ for $i = 1, 2, ..., r$.

The solution for the *best-case k-coverage problem* is almost identical to that of the worst case. Using Breadth First Search, we will start at one of the borders which belongs to seg_I and try to explore the unmarked borders; if a border bor_i is adjacent to a marked border and $deg(bor_i) \geq k$, bor_i will be included in the set of explored borders and be marked. Similarly, the minimum value of r such that there is a path, inside the region $D(S, k, r)$, connecting points I and F can be determined by binary search: if a path connecting seg_I and seg_F can be established, we will reduce the disk radius r and vice versa.

4.4.4 Performance Analysis

Theorem 4.3 The time complexities of the proposed algorithms for the *best-case* and *worst-case k-coverage problem* are $O(n^3)$.

Proof: As proven in Lemma 4.1, the time complexity for the computation of all borders is $O(n^3)$. We assume that border bor_i keeps a list $list_i$ which stores all its adjacent borders. Since each border has two end points and each end point will be the common point of at most 8 borders, the number of adjacent borders in $list_i$ is always bounded from above by

some constant C_1. The complexity of BFS is $O(|V| + |E|)$, where $|V|$ is the number of vertices in the BFS tree and $|E|$ is the number of edges in the BFS tree. In the BFS tree of proposed algorithm, each vertex denotes a border. Since the number of borders is of order $O(n^2)$, the number of vertices in BFS tree is also of order $O(n^2)$. On the other hand, each edge in the BFS tree denotes the adjacency of one border to another. Since each border will has at most C_1 adjacent borders, $|E|$ is bounded from above by $C_1|V|$ or $O(n^2)$. For binary search, the time complexity is $O(log\frac{range}{\varepsilon})$, where *range* is the search space and ε is the error that can be tolerated. Based on our earlier discussion, it is easy to see that the value of $log\frac{range}{\varepsilon}$ is typically upper-bounded by some constant C_2. Thus the resultant complexity is $O(1)$. Hence the proposed algorithms for both the *best-case* and *worst-case k-coverage problems* have time complexities of $O(n^3)$. ■

4.5 Extensions of Proposed Solution

4.5.1 Irregular Coverage Region

Although most existing works assume the coverage region of each sensor is a disk, this assumption may not be valid in some practical situations. The coverage region of a sensor may be affected by obstructions in the field and results in an irregular coverage region. Further, some sensors are designed to detect the target in a certain direction by using directional antennas. Thus for such sensors, the shape of coverage region is a sector instead of a disk.

In Section 4.3.1 of this chapter, given a set of sensors S and a target poi_j, we evaluate how well this target can be k-covered by using k-*th* distance. However, such evaluation is not valid when the coverage region of each sensor is not a disk. For illustration, consider the example shown in Figure 4.7, in which the target is quite close to three sensors. Thus it has a short 3-rd distance. However, this target can not be covered by any sensor since it does not fall in the coverage region of any sensor. Thus we can conclude that when the

coverage region is not a disk, the *worst-case k-coverage problem* and *best-case k-coverage problem* would need to be redefined.

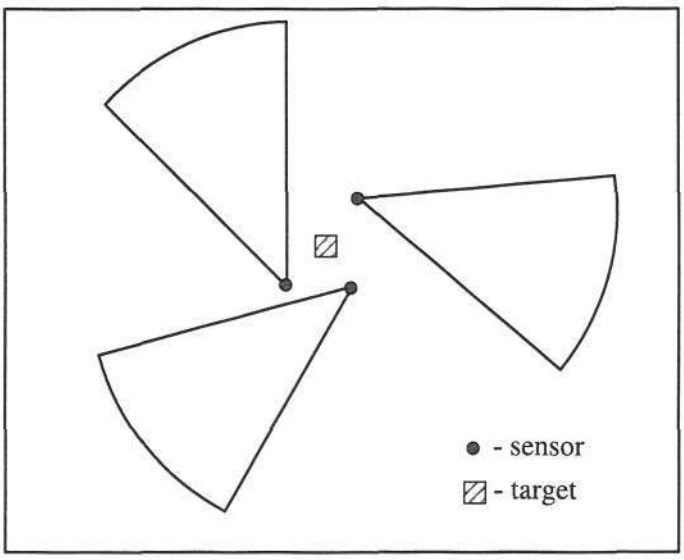


Figure 4.7: The shape of coverage region is a sector

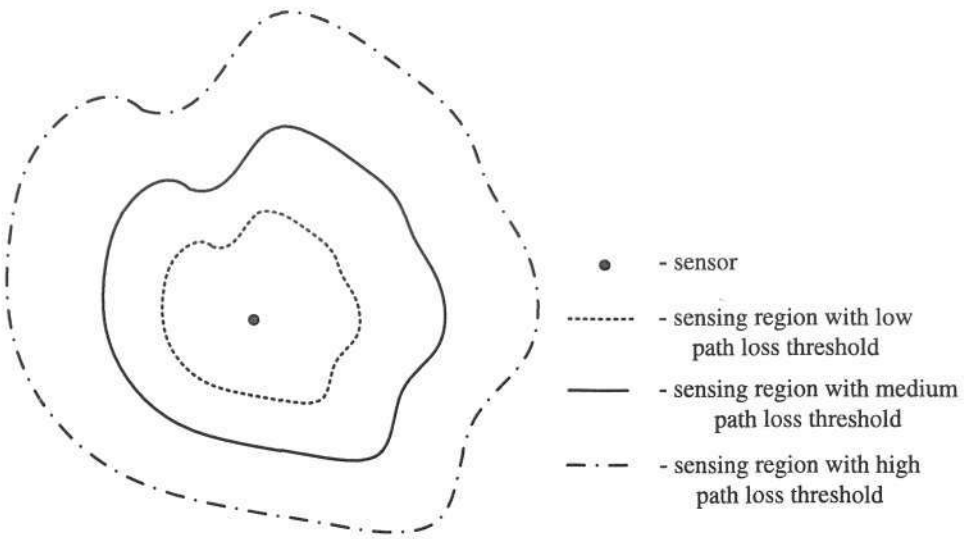


Figure 4.8: Sensing region increases as power increase

As described in Section 4.3.1, a point poi_i is covered by sensor s_i if the path loss from s_i to poi_i is no larger than a threshold p. Now let's consider the case that p is variable. As illustrated in Figure 4.8, the area which is covered by a given sensor s_i will increases as the

threshold p increases. We assume that all sensors have a unified path loss threshold p. Hence the region which is covered by s_i is a function of p, denoted by $d_i(p)$. Given a two dimensional field Ω, the sensor set $S = \{s_1, s_2, \ldots, s_n\}$ and the path loss threshold p, the region which is covered by at least k sensors can be denoted by $D(S, k, p)$ and the complementary region of $D(S, k, p)$ in the field Ω can be denoted by $\overline{D(S, k, p)}$. Since $d_i(p)$ will increase as p increases, $D(S, k, p)$ will also increase and $\overline{D(S, k, p)}$ will decrease as p increases.

Thus when the coverage region is not a disk, the *worst-case k-coverage problem* and *best-case k-coverage problem* can be defined as follows:

Worst-case k-coverage Problem. Given a field Ω, the sensor set S, the initial point I and final point F, identify a path P connecting I and F in the region $\overline{D(S, k, p)}$ while p is maximized.

Best-case k-coverage Problem. Given a field Ω, the sensor set S, the initial point I and final point F, identify a path P connecting I and F in the region $D(S, k, p)$ while p is minimized.

For a given value of p, the sensing radius of each sensor is irregular. Thus the area covered by each sensor is no longer a disk. Some existing works have proposed several schemes to detect the area covered by an arbitrary sensor when the sensing radius is irregular. For example, in [102], the area covered by a sensor is modeled as a set of polygons. This is a good approximation and the error tolerance can be well controlled. By applying such schemes, the area covered by an arbitrary sensor can be obtained. Thus the above mentioned two optimization problems can be solved by applying binary search if we can solve the following two decision problems:

Problem 4.5: ***Worst-Case k-coverage Problem.*** Given a field Ω, the sensor set S, the value of k and p, the initial point I and final point F, can we establish a path connecting I and F in the region $\overline{D(S, k, p)}$?

Problem 4.6: ***Best-Case k-coverage Problem.*** Given a field Ω, the sensor set S, the value of k and p, the initial point I and final point F, can we establish a path connecting I

and F in the region $D(S, k, p)$?

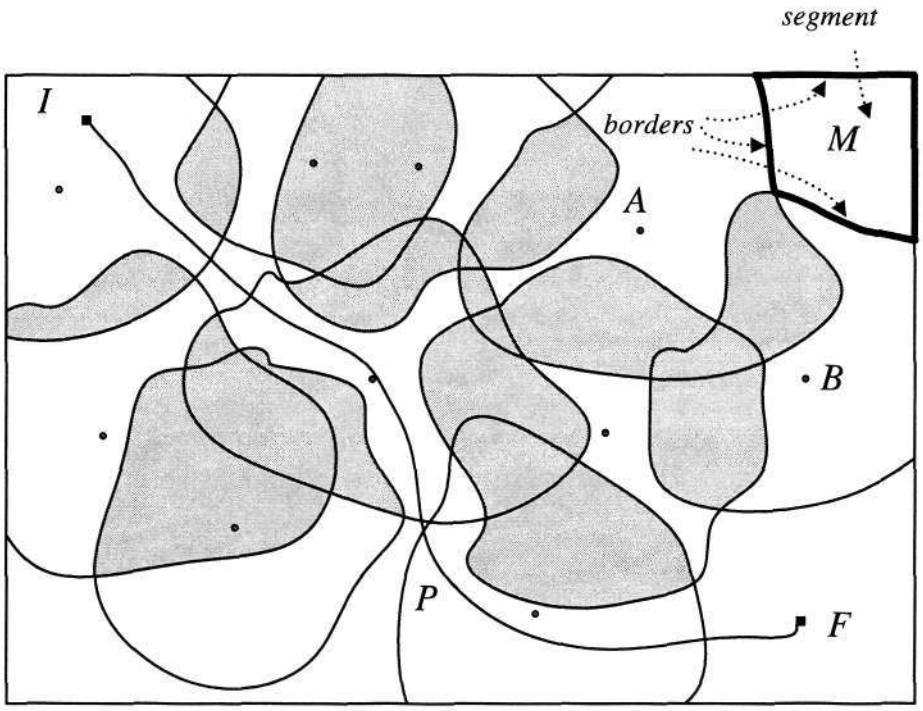


Figure 4.9: A path connecting I and F in the region $\overline{D(S, k, p)}$, where $k = 2$

Figure 4.9 gives an example of a path connecting I and F in the region $\overline{D(S, 2, p)}$. Similar to Section 4.4.1, given the value of p, the region which is covered by a sensor is a irregular $2 - D$ space within a *boundary*. The boundaries of all sensors in S will partition the field Ω into a set of *segments*, whereby each *border* of a segment is (i) a part of boundary of a region covered by a sensor or (ii) a part of boundary line(s) of the field. As an example shown in Figure 4.9, segment seg_M has three borders(labeled in bold): namely a part of boundary of the region covered by sensor A, a part of boundary of the region covered by sensor B and a part of boundary lines of the field. As shown in Figure 4.9, region $D(S, k, p)$ and $\overline{D(S, k, p)}$ are composed by many segments. Hence similar to the solution proposed in Section 4.4, Problem 4.5 and Problem 4.6 can be transformed into the problems of adjacent borders and solved by adopting BFS.

4.5.2 Distributed Algorithm

The algorithms proposed in Section 4.4 is based on a centralized approach, which assumes that a powerful server or cluster head is available to collect sensor locations and to compute the k-breach path or k-support path. However, in many sensor deployment environment such as disaster areas and battlefields, a centralized server may not be available. Thus a distributed algorithm is more desirable in sensor networks.

In this section, we assume there is no server or cluster head available. This is a more realistic assumption compared to Section 4.4. We will aim at solving Problem 4.1 and Problem 4.2 in a distributed manner.

Similar to Section 4.4, we adopt the *disk diagram*. Given the field Ω and the sensor set S, let C denotes the set of n disks of radius r, where for each disk $dis_i \in C$, it is centered at $s_i \in S$. We will illustrate how to solve Problem 4.1 and Problem 4.2 based on local knowledge.

Given a pair of disks: dis_A and dis_B, which are centered at sensor s_A and s_B, respectively, we note these two disks will intersect or be tangential to each other if the distance between s_A and s_B, denoted by d_{AB}, is smaller or equal to $2r$. The intersection point or tangential point can be calculated by either s_A or s_B if s_A and s_B can "*see*" each other, where "*see*" means they have the location information of each other.

Since we assume that the sensor network is always connected, we can guarantee that a sensor can be seen by any other sensors by broadcasting the location information of this sensor. This broadcasting will be done before the calculation of borders. As an alternative, some other positioning techniques reviewed in [103] can also be adopted to guarantee that each sensor can be seen by other sensors.

After the calculation of all borders, Problem 4.1 and Problem 4.2 are transformed into Problem 4.3 and Problem 4.4, respectively. Both these two problems can be solved by carrying out distributed BFS [104]. Starting at a border which is belong to seg_I, the

segment which initial point I falls in, the search process tries to explore unmarked borders which are adjacent to a marked border and have the coverage degree which is less than k(for *worst case k-coverage problem*) or larger or equal to k(for *best case k-coverage problem*).

Similar to Section 4.4 , the optimal k-breach and k-support can be calculated by binary search. The upper bound and lower bound of search space will be broadcasted in the sensor network in advance. After an iteration of BFS, the result will also be broadcasted in the network. Thus each sensor can determine the new value of r using binary search, calculate the borders and carry on the next iteration of BFS.

In summary, we have proposed a distributed solution for both *worst case k-coverage problem* and *best case k-coverage problem*. However, one shortcoming of this approach is the overhead of broadcasting the location information of each sensor. To reduce the broadcasting cost, we can adopt a alternative strategy described as follows:

- Instead of broadcasting the location information to all other sensors in the network, a given sensor s_i will only exchange the location information with the sensors which fall within the communication range of s_i.
- The rest sensors will be ignored by s_i in the calculation of borders.

By adopting the strategy described above, a pair of sensors will see each other only if the distance between them is no larger than R. In other words, the "eyesight" of each sensor is restricted to R. The performance of this strategy depends on the relationship between the communication range R and the radius of disk r:

Case i. $R \geq 2r$. Given a pair of disks dis_A and dis_B, which are centered at sensor s_A and s_B, respectively, we note these two disks will intersect or be tangential to each other if and only if the distance between s_A and s_B, denoted by d_{AB}, is smaller or equal to $2r$. Since $R \geq 2r$, we have: $R \geq 2r \geq d_{AB}$. This indicates that dis_A and dis_B will intersect or are tangential to each other only if s_A and s_B can see each other. Thus the intersection points

or tangential point produced by the intersection of dis_A and dis_B can be calculated by either s_A or s_B based on local knowledge.

Case ii. $R < 2r$. Given a pair of disks dis_A and dis_B, centered at sensor s_A and s_B, respectively, let d_{AB} denote the distance between s_A and s_B. Since $R < 2r$, there are two possibilities:

a). $d_{AB} \leq R < 2r$, it is same to the *Case i.*, since s_A and s_B can see each other, the intersection point or tangential point can be calculated locally;

b). $R < d_{AB} \leq 2r$, since the distance between s_A and s_B is larger than the maximum communication range, they can not see each other even if dis_A and dis_B are intersecting. Thus all such intersections will be ignored. Apparently this may affect the optimality of solution. The performance difference between this approximation algorithm and the centralized optimal algorithm will be studied experimentally in Section 4.7.

4.6 Applications

As discussed in Section 4.1, k-coverage is necessary in some applications. In this section we will discuss the applications of this study under several scenarios.

4.6.1 Coverage Evaluation

The algorithms we proposed can compute the *maximal k-breach path* and *minimum k-support path* in polynomial time. Thus it can be used to evaluate the coverage quality under different coverage degrees to cater different applications. For example, certain sensors can detect the azimuth angle of a target. When the target is detected by two such sensors simultaneously, given measurements of these two angles and coordinates of these two sensors, the coordinates of the target can be calculated by using a geometric technique which is referred to as *triangulation*. Although the target can be detected if it falls in the sensing radius of one sensor, the position of this target can be determined only if the target falls in the sensing

radius of two sensors. Given a set of sensors in a two-dimensional field, the initial point I and final point F, we can calculate the maximal 2-breach and the minimum 2-support by applying our proposed algorithm. Let $B(2)$ denote the maximal 2-breach and let $S(2)$ denote the minimum 2-support. Then when a target is traversing from I to F, at least at one point, this target will fall within $B(2)$ of at least two sensors. Thus if the sensing radius is larger or equal to $B(2)$, we can guarantee that this target can be located at some point when traversing from I to F. On the other hand, if the sensing radius is smaller than $S(2)$, then at a certain point, this target cannot be covered by at least two sensors and thus cannot be located.

4.6.2 Redundancy Analysis

Since sensors may run out of power or be damaged by the harsh environment, node failures may occur in the sensor network. Under such situations, it is important to provide some redundancy. It is apparent that the coverage degradation can be caused by node failures and such degradation will be reflected by the increase in values of k-breach and k-support. To estimate the coverage degradation in the worst case, we addressed the following problem:

Problem 4.7. Given a sensor network which is composed of n sensors, k and m are positive integers which satisfy $k + m \leq n$, what is the upper bound for both k-breach and k-support after m sensors failure?

Theorem 4.4. Given a sensor network with n sensors, let $B(k + m)$ denote the maximal $(k + m)$-breach and $S(k + m)$ denote the minimum $(k + m)$-support of this network. After removing m sensors in the network, let $S'(k)$ denote the maximal k-breach and let $B'(k)$ denote the minimum k-support. We have:

$$B'(k) \leq B(k + m); S'(k) \leq S(k + m)$$

Proof: The k-th distance of a point poi_j is defined as the distance from poi_j to the k-th nearest sensor. In the worst case, the m sensors which will be removed are the m

nearest sensors of poi_j. Thus the $(k+m)$-th nearest sensor will becomes the k-th nearest sensor of poi_j after the removing of m sensors. Then based on the definition of k-breach and k-support, the k-breach and k-support after removing of m sensors will be upper bounded by $(k+m)$-breach and $(k+m)$-support. ■

From Theorem 4.4, we can conclude that the maximal $k+m$-breach and minimum $k+m$-support of a field can be used as the worst-case guarantee for the maximal k-breach and minimum k-support of this field, respectively, after m sensor failures.

4.6.3 Energy Saving

In the energy saving scenario, most sensor nodes will be in the sleep mode while no targets are detected. Only a small portion of sensors will actively monitor the environment. By applying 1-coverage evaluation, we can determine the minimum number of sensors which is necessary to guarantee the effective detection of the target. After a target has been detected, it can be effectively tracked only if this target is covered by multiple ($k > 1$) sensors. Thus some sensors will be awakened to achieve higher coverage degree. By applying the proposed algorithms for the maximal k-breach and the minimum k-support problems, we can check whether a given set of active sensors is sufficient to achieve certain k-breach or k-support values and thus determine whether it is necessary to "wake up" more sensors.

4.7 Experimental Results

We have implemented the proposed algorithm by using Matlab 7.1 [105]. The target field is set to be a $600m \times 400m$ rectangular region, a set of n sensors is randomly deployed in the field with uniform distribution. The number of sensors n is gradually increased from 10 to 80 in steps of 10. For each value of n, we calculated the maximal *k-breach* and the minimum *k-support* for $k = 1, 2, 3, 4, 5$. Each data point is obtained by taking the average of 100 repetitions.

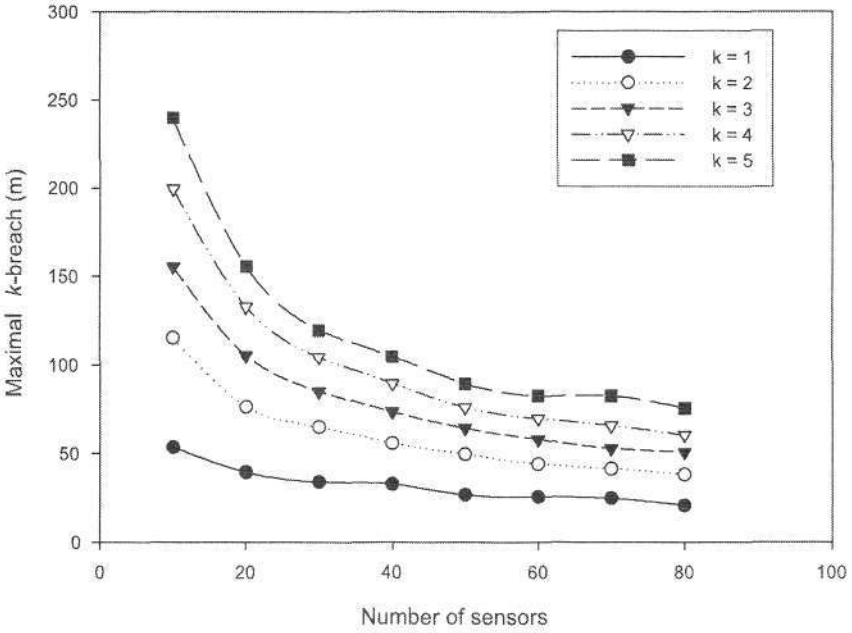


Figure 4.10: Maximal k-breach

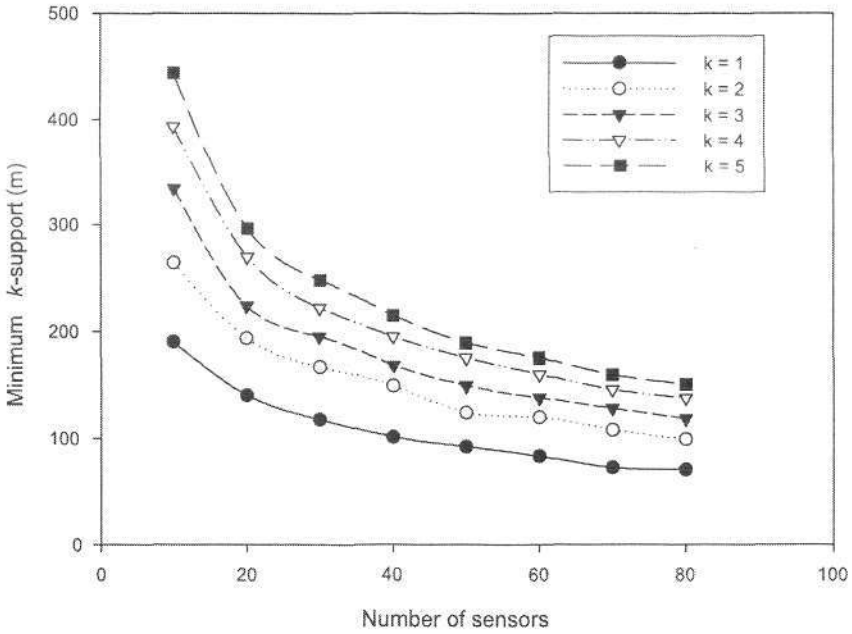


Figure 4.11: Minimum k-support

The results in Figure 4.10 and Figure 4.11 clearly show that both maximal *k-breach* and minimum *k-support* decreases when the number of sensors increases. Given the number of sensors and required coverage degree, these results can be used to estimate the coverage quality. We observe that deploying more sensors is an important measure to improve the quality of coverage. However, we note that as the number of sensor node increases, the percentage of improvement in coverage quality decreases. Hence the simulation results can be used to derive a trade-off between the number of sensors to be deployed and the desired improvement in coverage quality.

These results also show that when a larger k value is applied, both the maximal *k-breach* and the minimum *k-support* increases dramatically. This implies that when a larger k is defined, there are two measures to guarantee the quality of k-coverage: (i) increase the sensing radius; (ii) deploy more sensors to achieve a certain k-breach or k-support value. Both of them will increase the cost and the cost can be estimated by applying the algorithms and results proposed.

By comparing Figure 4.10 and Figure 4.11, we can see the minimum k-support is much larger than the maximal k-breach for same number of sensors. This is consistent with the definition of k-breach and k-support which inherently implies that the minimum k-support is no less than the maximal k-breach.

Figure 4.12 and Figure 4.13 compare the performance of centralized algorithm proposed in Section 4.4 and the distributed algorithm proposed in Section 4.5.2(using the approximate strategy). The communication range is set to $R = 150m$. We can see as discussed in Section 4.5.2, the optimal results can also be obtained by applying distributed algorithm if the k-breach and k-support is no larger than $R/2$. Further, when k-breach and k-support exceed $R/2$, we can note that the distributed algorithm differs from the optimal solutions by no more than 5%. For an instance, from Figure 4.12 we can see that when 20 sensors are placed in Ω, the 3-breach calculated by centralized algorithm is 109.6, which is larger than $R/2$. As a comparison, the 3-breach calculated by the approximate distributed algorithm is 114.0,

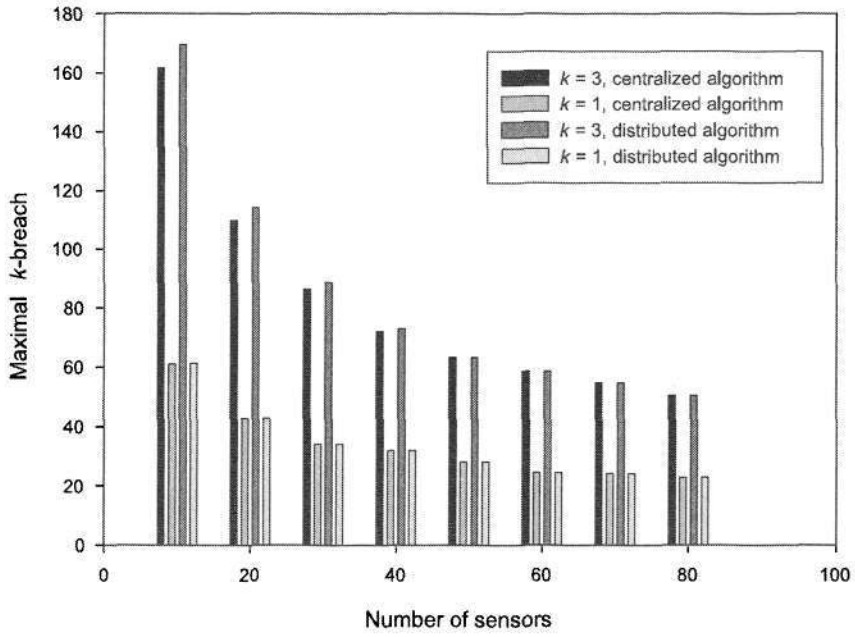


Figure 4.12: Comparison of maximal k-breach

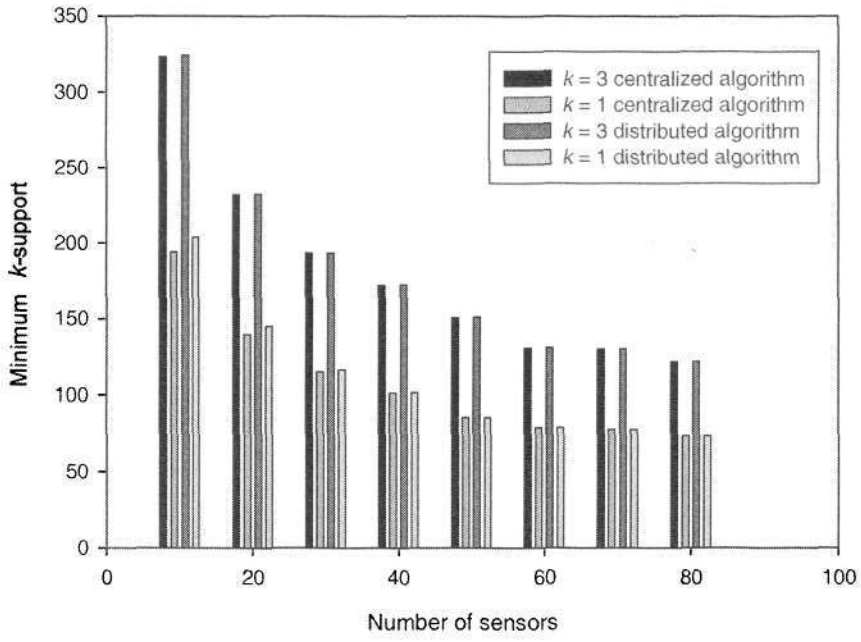


Figure 4.13: Comparison of minimum k-support

which differs from the optimal results by 4.9%. Thus our experimental studies show that the proposed distributed algorithm is a good approximation algorithm for calculating the k-breach and k-support. Next we can note that the k-breach and k-support calculated by distributed algorithm are no less than the centralized algorithm, this phenomenon is expected since the distributed algorithm may ignore some sensors in the calculation, which in turn results in larger k-breach and k-support value.

4.8 Summary

In this chapter, we presented the formulation of *worst-case* and *best-case k-coverage problem*. This is a generalization of some earlier work on the *1*-coverage problem. By adopting the basic idea of growing disks and proposing a series of definitions and theorems, we transformed the *k-coverage problem* into one of finding a sequence of adjacent borders and we proposed optimal polynomial time algorithms to solve both variants of the problem. Two important extensions of the study on the problem were also addressed. Firstly, we addressed the problem by removing the assumption that the coverage region of each sensor is a unified disk and proposed an approach to solve the problems under such scenario. Secondly, we discuss how our proposed algorithms can be adapted to solve the problems in a distributed manner. Both the extensions address issues that arise in most practical scenarios.

Chapter 5

A Generalized Study of The Movement-Assisted Sensor Deployment Problem

As mentioned in Chapter 4, coverage is a fundamental problem in sensor networks. In Chapter 4, we assume all sensors are fixed and evaluate the sensor coverage by using the worst-case and the best-case k-coverage. In this chapter, we addressed another aspect of coverage problem in WSNs: deploy a set of sensors in a target field to achieve a good coverage. We assume the sensors in a given WSN are mobile. By noticing that mobility of sensors can be used to enhance the coverage of a target field, some existing works have proposed several movement-assisted sensor deployment schemes. These works assume that the target field to be a 2-dimensional space. In this chapter, we study a generalized case of this problem whereby the target field can be a space which ranges from 1-dimensional to 3-dimensional. Two variations of the movement-assisted sensor deployment problem with different optimization objectives are formulated. We identify a set of basic attributes which can be used as guidelines for designing movement-assisted sensor deployment schemes. Based on these attributes, we propose efficient algorithms for both variants of the movement-assisted sensor deployment problem.

5.1 Introduction

Wireless sensor networks(WSNs), although were originally developed for military applications, are now widely used in many civilian application areas, including environment and habitat monitoring, home automation, and traffic control [10, 11]. A WSN is composed of a set of spatially distributed autonomous nodes which uses sensors to cooperatively collect data concerning environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations [10, 12].

Coverage problem is one of the fundamental issues in a sensor network. Given a set of sensors with uniform communication range and sensing range, the coverage, communication cost, routing issues are all greatly affected by the positioning of sensors. Thus, sensors must be deployed appropriately to meet the coverage requirements. However, in many typical applications, a random placement maybe adopted for the following reasons:

- Sensors may be deployed at remote harsh fields, where manual deployment is not practical. Hence a possible solution is to scatter the sensors by dropping or throwing. However, it is impossible to control the actual landing positions accurately by using such techniques.

- Since sensors may be deployed at a hostile region, there may be a lack of knowledge about the target field prior to deployment. Thus it is impossible to determine the optimal positions of each sensors before deployment.

Apparently the random placement of sensors may lead to ineffective coverage, especially if some of the sensors are overly clustered or are placed out of the target field. One of the promising solutions is to make use of the mobility of sensors. In particular, after an initial random placement, the sensors will move toward selected locations to enhance the coverage. This self-deploying technique is referred to as movement-assisted sensor deployment [106].

Existing movement-assisted sensor deployment schemes usually consider the problem of deploying a set of sensors in a 2-*dimensional* (2-D) field. By noticing that the sensor

deployment schemes for 1-D, 2-D and 3-D cases share some common attributes, we study the general problem of movement-assisted sensor deployment in the context of achieving good coverage. In this generalized formulation, the target field may be a 1-D, 2-D or 3-D space. We identify a set of basic attributes for movement-assisted sensor deployment schemes. Based on that, we proposed efficient algorithms for general movement-assisted sensor deployment problem.

The rest part of this chapter will be organized as follows: In the next section, we review some existing work on sensor deployment problem. In Section 5.3, we present the problem assumptions and formulate two variations of movement-assisted sensor deployment problem. An in-depth problem analysis is given in Section 5.4. In Section 5.5 and Section 5.6, we will propose efficient algorithms for the two problems formulated in Section 5.3. Several optimization issues are discussed in Section 5.7. Section 5.8 gives the experimental results and the last section concludes this chapter.

5.2 Related Work

In general, the sensor deployment problem is to determine positions and/or movements of nodes to achieve maximum coverage and to form a uniformly distributed wireless network. Sometimes the time and energy consumption will also be considered as performance metrics. One of the approaches is to deploy the sensors in an incremental way, and new sensors will be deployed into the field based on the knowledge collected by prior sensors. This approach is referred to as incremental sensor deployment [107]. However, this deployment usually consumes a long time, since sensors have to be deployed one by one. Further, it requires the environment to be static; thus it is unable to adapt to dynamic environment.

By noticing that the mobility of sensors can be used to enhance the coverage of a target field, several movement-assisted sensor deployment schemes have been proposed to enhance coverage. In [108], Wang *et al.* considered the problem of deploying a mixture of mobile and

static sensors to provide the required uniform distribution of sensors. A bidding protocol, which is a greedy approximation algorithm, was proposed to deploy mobile sensors to achieve the desired distribution. In [109], Zou *et al.* proposed a centralized and virtual force based sensor deployment algorithm(VFA). Zou *et al.* assume that there is a powerful cluster head, which will collect the position information of all sensors, calculate the virtual force and direct the sensor movements. Three distributed algorithms, VEC, VOR and Minimax were proposed by Wang *et al.* in [110]. VEC is based on virtual force, while VOR and Minimax are based on Voronoi diagrams [94, 95]. In [111], Wu *et al.* introduced SMART, a hybrid of centralized and distributed approach. The sensors were partitioned into an $n \times n$ mesh of grids. There is a leader for each grid. The leader will direct the sensor exchange based on column and row scan to achieve a balanced state. In [112], Wang *et al.* addressed the problem of repairing the coverage hole by using mobile sensors while keeping the rest part of the network unaffected. Another movement-assisted sensor deployment scheme which is based on density control was proposed in [113], by Chang *et al.*

Since the energy is a scarce resource for each node in a WSN, some studies aim at deploying the sensors in an energy-efficient manner. In [114], Chellappan *et al.* guaranteed the energy consumption of each node by restricting the maximum moving distance of each sensor. Chellappan *et al.* adopted the methodology of transferring the nonlinear variance/movement minimization problem into a linear optimization problem and proposed a set of algorithms for the problem addressed. In [115], Heo *et al.* considered the problem of deriving a topology to maximize the system lifetime by utilizing mobility of sensor nodes. A energy model which characterize the entire energy consumption in sensor movement was proposed by Wang *et al.* [116]. Based on the model, Wang *et al.* proposed an optimal velocity schedule for minimizing energy consumption when the road condition is uniform; and a near optimal velocity schedule for variable road conditions.

Although many sensor deployment schemes have been proposed in existing work, these schemes usually only consider the problem of deploying a set of sensors in a 2-*dimensional*

(2-D) field. Further, they assume that the field in which the sensors are initially placed is identical to the target field. We note that these assumptions may not be valid in certain scenarios:

- In some practical applications, the target field may not be a 2-D plane. For example, in underwater sensor networks, the target field may be a 3-D space [117, 118]. On the other hand, in some applications we are interested about "path coverage" [87, 119], which is the coverage of a $1-D$ target field.

- As mentioned before, sensors may be deployed at a region where there may be a lack of knowledge about the target field prior to deployment. Further, sensors may be initially scattered at a non-accurately controlled way, such as dropping and throwing. These may cause a gap between the target field and the field in which the sensors are initially placed. In some cases, the target field and the field in which the sensors are initially placed are even not equal in dimension. For example, in "path coverage", the target field is 1-D while the sensors may be initially placed in a 2-D field.

Based on above-mentioned observations, in the study of this chapter, we address a more general problem: the initial deployment field and target field can be a 1-D, 2-D or 3-D space. Further, the initial deployment field and the target field are not required to be identical to each other. We will show that the sensor deployment schemes for 1-D, 2-D and 3-D cases share some common attributes, which will be identified and an efficient algorithms will be proposed based on these attributes.

5.3 Generalized Movement-Assisted Sensor Deployment Problem

In this section, we will describe the underlying assumptions and give the problem formulation.

5.3.1 Preliminaries

We assume that the wireless sensor nodes are given as a set S with n sensor nodes which are initially distributed inside a continuous field Ω_s. We assume that Ω_s is a 3-D space. The case of Ω_s being a 2-D or 1-D space will be considered as a special case of 3-D space. For each sensor $s_i \in S$, its location $l_i = (x_i, y_i, z_i)$ is known. The target field is a continuous space denoted by Ω_t, which may be a 1-D, 2-D or 3-D space. Each sensor is assumed to have the same sensing range, denoted by r. Given a point P at (x, y, z), we denote the Euclidean distance between P and sensor s_i as $d(P, s_i)$, i.e. $d(P, s_i) = |\overrightarrow{l_P - l_i}| = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}$.

We adopt the binary sensing model [120, 121] for its simplicity. In this model, the coverage of a point $P = (x, y, z)$ by sensor s_i, denoted by $c(P, s_i)$, can be described as following equation:

$$c(P, s_i) = \begin{cases} 1\,(P \text{ is covered by } s_i) & \text{if } d(P, s_i) \leq r \\ 0\,(P \text{ is not covered by } s_i) & \text{otherwise} \end{cases} \tag{5.1}$$

where r is the sensing radius of sensor s_i.

Given two sensors $s_i \in S$ and $s_j \in S$, let R denote the maximum communication range of each sensor and let $d(s_i, s_j)$ denote the distance between s_i and s_j. We say s_i and s_j can *see* and *hear* each other if $d_(s_i, s_j) \leq R$. Here "*see*" means they know the distance and azimuth angle of each other(this can be done by location information exchange) and "*hear*" means they can communicate to each other directly. We assume the communication range R is always larger than $2r$. Thus a pair of sensors will able to see and hear each other if they have overlapped coverage space. Further, we assume each sensor is aware about the existence and location of the target field boundary.

We assume the sensor network is organized in a distributed way, which means all calculations are based on local knowledge. This is a more practical assumption as compared with the centralized model used in some existing works such as [109] because there is no powerful cluster head or server available in many environments. Further, distributed approached will

not introduce the "single point failure" problem and is thus more reliable.

5.3.2 Problem Formulation

In this section, we will formulate movement-assisted sensor deployment problem by using two different optimization objectives.

Maximizing Coverage Ratio

Given a sensor s_i located at (x_i, y_i, z_i), let $C(s_i)$ denote the region covered by s_i. It follows from (5.1) that $C(s_i)$ is a *sphere* centered at $l_i = (x_i, y_i, z_i)$, with radius r.

However, since sensors are initially distributed in Ω_s, which may not be equal to the target field Ω_t, it is possible that $C(s_i) \nsubseteq \Omega_t$. In an extreme case, $C(s_i)$ and Ω_t may have no intersection, i.e $C(s_i) \cap \Omega_t = \emptyset$. Thus we make the following definition to describe the "effectiveness" of the sensor coverage.

Definition 5.1: Effective Coverage. Given a sensor s_i with coverage $C(s_i)$ and a target field Ω_t, the effective coverage of s_i in Ω_t, denoted by $C_e(s_i, \Omega_t)$, is defined as the intersection of $C(s_i)$ and Ω_t: $C_e(s_i, \Omega_t) = C(s_i) \cap \Omega_t$.

The region which belongs to Ω_t and is covered by S, denoted by $C_e(S, \Omega_t)$, can be defined as: $C_e(S, \Omega_t) = \bigcup_{i=1}^{n} C_e(s_i, \Omega_t)$. When the binary sensor model is adopted, we can evaluate the coverage of Ω_t by S by using $ratio_e$, which is the ratio between $C_e(S, \Omega_t)$ and Ω_t, i.e $ratio_e = \frac{C_e(S,\Omega_t)}{\Omega_t}$. This ratio can be affected by the sensing radius r and the sensor locations. Since we assume the sensing radius r is known and fixed, the $ratio_e$ is thus determined by the sensor locations.

The Maximizing Coverage Ratio Problem (MCRP) can be formulated as:

Given: the field in which sensors are initially placed Ω_s, the target field to be covered Ω_t, the sensor set $S = \{s_1, s_2, \ldots, s_n\}$, the sensing radius r, the initial locations of sensors $L_s = \{l_1, l_2, \ldots, l_n\}$.

Determine: the target location of all sensors $L_t = \{l'_1, l'_2, \ldots, l'_n\}$ to maximize the coverage ratio $ratio_e = \frac{C_e(S,\Omega_t)}{\Omega_t}$.

Minimizing Maximal First Distance

Next we will introduce a variation of movement-assisted sensor deployment problem which has a different optimization objective. Given a point $P \in \Omega_t$ and a set of sensors S, the *first distance* of P, denoted by $D_{1st}(P, S)$, is defined as the Euclidean distance between P and the nearest sensor of P in S. It follows from equation (5.1) that the coverage of P is determined by its first distance. We say P is *better covered* when $D_{1st}(P, S)$ is smaller and vice versa.

Given the target field Ω_t and a set of sensors S, the coverage of Ω_t by S can be evaluated from the viewpoint of the worst case. In other words, the coverage of Ω_t is determined by the worst covered point in Ω_t, defined as follows:

Definition 5.2: Worst Covered Point. Given a set of sensors S and the the target field Ω_t, the worst-covered point, denoted by P_w, is defined as the point in Ω_t whose first distance is the maximal, i.e $D_{1st}(P_w, S) = max(D_{1st}(P_i, S)), \forall P_i \in \Omega_t$.

Thus the Minimizing Maximal First Distance Problem (MFDP) can be formulated as follows:

Given: the field in which sensors are initially placed Ω_s, the target field to be covered Ω_t, the sensor set $S = \{s_1, s_2, \ldots, s_n\}$, the initial locations of sensors $L_s = \{l_1, l_2, \ldots, l_n\}$.

Determine: the target location of all sensors $L_t = \{l'_1, l'_2, \ldots, l'_n\}$ to minimize the first distance of the worst covered point $D_{1st}(P_w, S)$.

5.4 Problem Analysis

Before we propose the solution strategies for MCRP and MFDP, we will made a comparison between these two problems and identify a set of basic attributes which could be used in

designing movement-assisted sensor deployment schemes.

5.4.1 Attributes for MCRP Scheme

In MCRP, the objective of moving sensors to their target locations is to enhance the coverage ratio of Ω_t. To obtain a good coverage ratio with limited number of sensors, the sensor movement scheme should have the following attributes:

1. Since sensor s_i may be initially placed out of the target field, i.e $(x_i, y_i, z_i) \notin \Omega_t$, s_i should move towards the target field to achieve a better coverage. Thus the scheme should move sensors which are outside of Ω_t to the target field.

2. For a sensor s_i which is inside the target field, i.e $(x_i, y_i, z_i) \in \Omega_t$, if s_i is too close to the boundary of Ω_t, some of the coverage may fall outside of Ω_t and thus be wasted. To maximize the coverage of s_i, s_i should not be placed too close to the boundary of target field.

3. If several sensors are placed densely in a small region, which are referred to as clusters, the overlapping of the sensor coverage will cause redundant coverage and reduce the total effective coverage. In such cases, the clustered sensors should move away from the clusters to reduce the overlapping of coverage.

4. The uncovered region in Ω_t is referred to as a *potential field* [122, 123] or *coverage hole* [110, 112]. Moving sensors to the potential field will possibly result in an increase of effective sensor coverage. Thus a movement-assisted sensor deployment scheme should be able to detect the potential field and drive sensors towards the potential field.

5.4.2 Attributes for MFDP Scheme

The scheme for MFDP aims at minimizing the first distance of the worst covered point in Ω_t. To achieve this objective, the sensor movement scheme should have the following attributes:

1. Similar to the MCRP scheme, since sensor s_i may be initially placed out of the target field, an effective MFDP scheme should move sensors which are outside of Ω_t to the target field.
2. The key issue of minimizing the first distance is to detect the worst covered area or at least detect some possible worst covered areas. When such areas are detected, some sensors should be moved towards them to reduce the first distance of the points in these areas.

The four attributes summarized in Section 5.4.1 and two attributes summarized in Section 5.4.2 can be used as guidelines for devising efficient algorithms for MCRP and MFDP. Although we note that some of the existing works have taken some of the above mentioned attributes into consideration, to the best of our knowledge, this is the first work that comprehensively identifies all the above-mentioned attributes that should be taken into consideration in designing solution strategies for MCRP and MFDP.

5.5 Proposed Algorithm for MCRP

5.5.1 Movement Methodology

Two variations of movement-assisted sensor deployment problem are formulated in Section 5.3.2, both of which aim at determining the target location of sensors L_t to enhance the coverage of Ω_t. However, due to the lack of global knowledge, it is impractical to determine L_t directly. Thus we adopt the same approach used in [109, 110]: the sensor will move iteratively (instead of directly) move to the target location. This protocol can be described

as:

while (*terminating condition* $\neq$ true)

$move = \textbf{calculate}(L, \Omega_t)$

$L = L + move$

return L

where L is the set of locations of all sensors, i.e $L = \{l_1, l_2, \ldots, l_n\}$; *move* is a set of n vectors to denote the moving of all sensors, i.e $move = \{\overrightarrow{m_1}, \overrightarrow{m_2}, \ldots, \overrightarrow{m_n}\} = \{(m_1^x, m_1^y, m_1^z), (m_2^x, m_2^y, m_2^z), \ldots, (m_n^x, m_n^y, m_n^z)\}$.

5.5.2 Virtual Force

As described in Section 5.5.1, sensors will move iteratively based on the vector set *move* which they have calculated. In this section, we will focus on the problem of determining the vector set *move* based on the location of sensors L and the target field Ω_t.

We adopt the idea of virtual force proposed in [109]. Any factor which will cause the movement of sensors will be considered as a "force". These forces can be either positive(attractive) or negative(repulsive). The movement of a sensor s_i in each iteration will be calculated as a vector summation of all forces executed on s_i.

Our solution is based on the four attributes given in Section 5.4.1. For a given sensor s_i, the target field, boundaries of target field, sensors which are close to s_i and potential field will behave as "sources of force" of s_i. The definition of these four forces will be described in the following parts.

5.5.3 Attractive Force of Target Field

When a sensor s_i is placed outside of Ω_t, Ω_t will execute the attractive force on s_i, denoted by $\overrightarrow{F_A(s_i)}$, which is defined as:

$$\overrightarrow{F_A(s_i)} = \begin{cases} \overrightarrow{l_A - l_i} & \text{if } l_i \notin \Omega_t \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

where A is a point in Ω_t which is nearest to s_i. $l_A = (x_A, y_A, z_A)$ is used to denote the location of A. $\overrightarrow{l_A - l_i}$ is a vector which points from (x_i, y_i, z_i) to (x_A, y_A, z_A).

In each iteration, each sensor will compare its location and the map of Ω_t which has been pre-stored in sensors. If s_i finds itself to be out of the target field, it will choose the nearest point in Ω_t and move towards it.

5.5.4 Repulsive Force of Boundary

When a sensor s_i is placed in Ω_t and is close to a boundary of Ω_t, this boundary will execute the repulsive force $\overrightarrow{F_B(s_i)}$ on s_i.

When Ω_t is a 3-D space, a boundary of Ω_t will be some 2-D surface. However, since Ω_t may also be 1-D or 2-D, a boundary of Ω_t may be of different dimensions. To resolve this problem and unify the formulation of $\overrightarrow{F_B(s_i)}$, we made the following definition:

Definition 5.3: Extended Boundary. Given a limited and continuous 1-D or 2-D space Ω, an extended boundary of Ω will be defined as a 2-D space which is perpendicular to Ω and contains a boundary of Ω.

Figure 5.1 gives two examples for the definition of extended boundary. As shown in Figure 5.1 a), when Ω is a part of straight line, the boundaries of Ω will be two points: A and B. Thus according to Definition 5.3, the extended boundaries of Ω will be the two planes which are perpendicular to Ω and include A and B, separately. As another example shown in Figure 5.1 b), when Ω is a plane, the boundaries of Ω will be some line segments. Thus according to Definition 5.3, the extended boundaries of Ω will be some planes which are perpendicular to Ω and include the corresponding line segments.

By replacing the boundaries of Ω_t by its extended boundaries in the case where Ω_t is

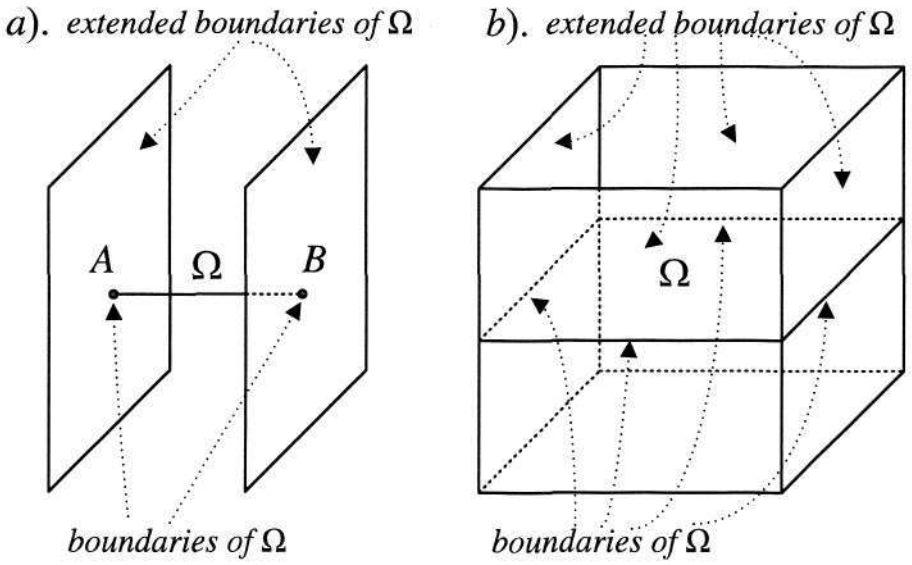


Figure 5.1: Examples of extended boundaries

a 1-D or 2-D space, we can unify all boundaries into 2-D spaces. Given a sensor s_i, let B denote a point in an extended boundary of Ω_t which is nearest to s_i. $\overrightarrow{F_B(s_i)}$ can be defined as:

$$\overrightarrow{F_B(s_i)} = \begin{cases} \frac{\overrightarrow{l_i - l_B}}{d(B,s_i)} \cdot (r - d(B,s_i)) & \text{if } l_i \in \Omega_t \text{ AND } d(B,s_i) < r \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

We note that when $l_i \in \Omega_t$, there may be m boundaries ($m > 1$) or extended boundaries whose distance to s_i is smaller than r. In such cases, $\overrightarrow{F_B(s_i)}$ will be calculated as the vector summation of all boundaries:

$$\overrightarrow{F_B(s_i)} = \sum_{j=1}^{j=m} \overrightarrow{F_B(s_i)_j} \tag{5.4}$$

To apply $\overrightarrow{F_B(s_i)}$, each sensor will compare its location and the map of Ω_t stored in it. If s_i finds itself is too close to some boundaries or extended boundaries, it will "bounce back" by the resultant force from these boundaries.

5.5.5 Repulsive Force between Sensors

Given a sensor s_i, the repulsive force executed by s_j on s_i is defined as:

$$\overrightarrow{F_C(s_j, s_i)} = \begin{cases} \frac{\overrightarrow{l_i - l_j}}{d(s_j, s_i)} \cdot (2r - d(s_j, s_i)) & \text{if } d(s_j, s_i) < 2r \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

The total repulsive force of other sensors executed on s_i $\overrightarrow{F_C(s_i)}$ can be calculated as the vector summation as follows:

$$\overrightarrow{F_C(s_i)} = \sum_{j=1}^{j=n} \overrightarrow{F_C(s_j, s_i)} \tag{5.6}$$

In Section 5.3.1, we have made the assumption that two sensors can see and hear each other if their coverage region are overlapping. In each iteration, each sensor will exchange location information with the sensors which fall in the communication range of it. If a pair of sensors find they are closer than $2r$ to each other, they will execute the repulsive force on each other and this tends to result in a larger distance between these two sensors.

5.5.6 Attractive Force of Potential Field

The most important design challenge for moving sensor to a potential field or repairing a coverage hole is to detect the potential field or coverage hole effectively and accurately. To guarantee that the movement of a sensor to an uncovered area will increase the local coverage, we will move a sensor to an uncovered area only if this area is *adjacent* to the sensor, whereby the notion of adjacency is defined as follows:

Definition 5.4: Adjacent Potential Field. Given a sensor s_i and an uncovered area Ω_A in Ω_t, let P_i denote a point in Ω_A which is nearest to s_i. We say Ω_A is an adjacent potential field of s_i if $d(s_i, P_i) = r$.

Given a sensor s_i with effective coverage $C_e(s_i, \Omega_t)$, let $B(s_i, \Omega_t)$ denote the boundary of $C_e(s_i, \Omega_t)$. We will show that all potential fields or coverage holes which are adjacent to s_i

can be detected in a distributed way by checking whether all points in $B(s_i, \Omega_t)$ are covered by some sensors other than s_i.

Case i: Ω_t is a 1-D field.

To simplify the discussion, we will only consider the case that Ω_t is a straight line. Then $C_e(s_i, \Omega_t)$ will be a line segment. We can check whether there exists an adjacent potential field of s_i by following lemma:

Lemma 5.1: Let T_A and T_B denote the two end points of $C_e(s_i, \Omega_t)$, then T_A (or T_B) will belongs to some adjacent potential field of s_i if and only if T_A (or T_B) is not covered by any sensors beside s_i.

Proof: "if": Since T_A (or T_B) is the end point of $C_e(s_i, \Omega_t)$, it must belongs to $B(s_i, \Omega_t)$, thus $d(T_A, s_i) = r$ and T_A (or T_B) must belongs to some area which is not covered by s_i. Then based on the Definition 5.4, T_A (or T_B) must belong to some adjacent potential field of s_i.

"only if": if T_A (or T_B) is covered by some sensors beside s_i, then T_A (or T_B) must belongs to some field which is covered. Thus this field is not a potential field.

■

Next we will focus on determining the magnitude of attractive force on a sensor from its adjacent potential fields. Since the potential fields are some line segments, for each potential field, the magnitude of its attractive force will be defined to be proportional to its length. We note that there are two possible adjacent potential fields of s_i, denoted by Ω_A and Ω_B. Without loss of generality, let T_A and T'_A denote the two end points of Ω_A, as shown in Figure 5.2. The length of this potential field can be calculated as the distance between T_A and T'_A: $d(T_A, T'_A)$.

However, since T'_A belongs to $B(s_j, \Omega_t)$ while s_j may be too far to see for s_i, the location of T'_A may not be determined accurately by s_i, based on its local knowledge. To resolve this problem, we adopt the following approximation: a). When $d(T'_A, s_i) \leq R - r$, we have: $d(s_j, s_i) \leq d(T'_A, s_i) + d(T'_A, s_j) \leq R - r + r = R$, this implies that s_j can be seen by s_i

and $d(T_A, T'_A)$ can be calculated accurately; b). When $d(T'_A, s_i) > R - r$, let T_C denote a point in Ω_t which satisfy $d(T_C, s_i) = R - r$, as illustrated in Figure 5.2. We will assume $d(T_A, T'_A) = d(T_C, T_A)$.

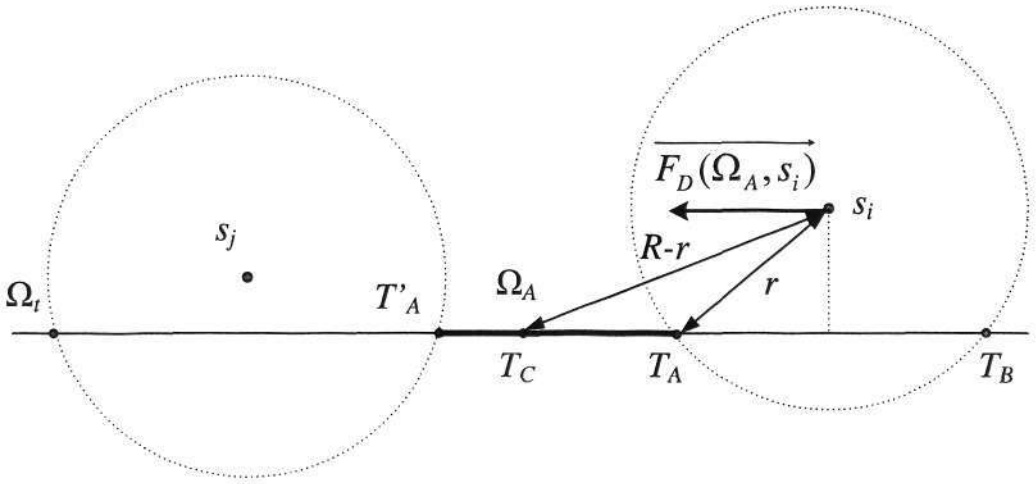


Figure 5.2: Attractive force of a 1-D potential field

Then the attractive force of an adjacent potential field Ω_A can be defined as:

$$\overrightarrow{F_D(\Omega_A, s_i)} = \begin{cases} min\{\overrightarrow{l_{T_{A'}} - l_{T_A}}, \overrightarrow{l_{T_C} - l_{T_A}}\} & \text{if } C_e(s_i, \Omega_t) \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

The total attractive force can be calculated as the vector total:

$$\overrightarrow{F_D(s_i)} = \overrightarrow{F_D(\Omega_A, s_i)} + \overrightarrow{F_D(\Omega_B, s_i)} \tag{5.8}$$

Case ii: Ω_t is a 2-D field.

For the sake of simplicity, we will only consider the case that Ω_t is a plane. Thus $C_e(s_i, \Omega_t)$ will be a circle if $C_e(s_i, \Omega_t) \neq \emptyset$ and $B(s_i, \Omega_t)$ will be the perimeter of this circle.

It is can be deduced from the result proven by Huang *et al.* [88] that an adjacent potential field of s_i, denoted by Ω_j exists if and only if part of $B(s_i, \Omega_t)$ is not covered by any sensors beside s_i. Further, Huang *et al.* have proposed a distributed algorithm to check whether $B(s_i, \Omega_t)$ is covered by some sensor beside s_i.

We adopt the algorithm proposed in [88] to detect the adjacent potential fields for a given sensor s_i. As shown in Figure 5.3, let ARC_j denote the arc which is a part of $B(s_i, \Omega_t)$. If ARC_j is not covered by any sensors beside s_i, there must be an adjacent potential field Ω_j which includes ARC_j. Let θ denote the central angle of this arc and let P_j denote the mid point of this arc. The attractive force of Ω_j executed on s_i is defined as:

$$\overrightarrow{F_D(\Omega_j, s_i)} = \begin{cases} \overrightarrow{l_{P_j} - l_i} \cdot \sin(\frac{\theta}{2}) & \text{if } C_e(s_i, \Omega_t) \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

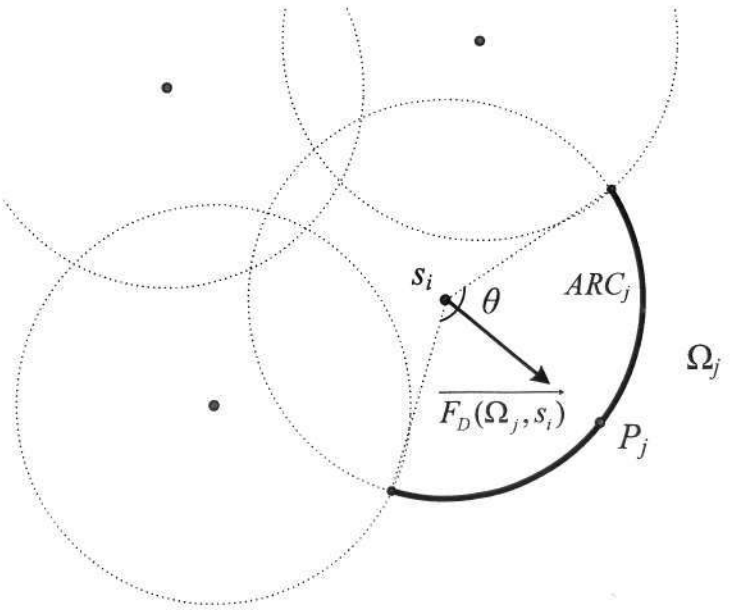


Figure 5.3: Attractive force of a 2-D potential field

Sensor s_i may have m ($m \geq 1$) adjacent potential fields. Thus the total attraction force can be calculated as the vector total:

$$\overrightarrow{F_D(s_i)} = \sum_{j=1}^{m} \overrightarrow{F_D(\Omega_j, s_i)} \tag{5.10}$$

Case iii: Ω_t is a 3-D field.

The 3-D case can be considered as an extension of 2-D case. When Ω_t is a 3-D field, $C_e(s_i, \Omega_t)$ will be a sphere and $B(s_i, \Omega_t)$ will be a spherical surface. In [89], Huang *et al.* proved that

an adjacent potential field of s_i, denoted by Ω_u exists if and only if part of the $B(s_i, \Omega_t)$ is not covered by any sensors beside s_i. Huang *et al.* further proved that Ω_u exists if and only if part of the circle $Cir(i,j)$, defined as the intersection circle of $B(s_i, \Omega_t)$ and $B(s_j, \Omega_t)$, is not covered by any sensors beside s_i and s_j.

Huang *et al.* [89] proposed a distributed algorithm to check whether a circle $Cir(i,j)$ is covered by some sensors beside s_i and s_j. We adopt this algorithm to detect the adjacent potential fields for a given sensor s_i. As shown in Figure 5.4, given an arc $ARC_j^i q$ which is a part of the $Cir(i,j)$, there exists an adjacent potential field $\Omega_j q$ which includes $ARC_j^i q$ if $ARC_j^i q$ is not covered by any sensors beside s_i and s_j. Let O_{ij} denote the center point of $Cir(i,j)$, let φ denote the central angle of $ARC_j^i q$ and let P_q denote the mid point of this arc, the attractive force of $\Omega_j q$ executed on s_i is defined as:

$$\overrightarrow{F_D(\Omega_j q, s_i)} = \begin{cases} \overrightarrow{l_{P_q} - l_{O_{ij}}} \cdot \sin(\frac{\varphi}{2}) & \text{if } C_c(s_i, \Omega_t) \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{5.11}$$

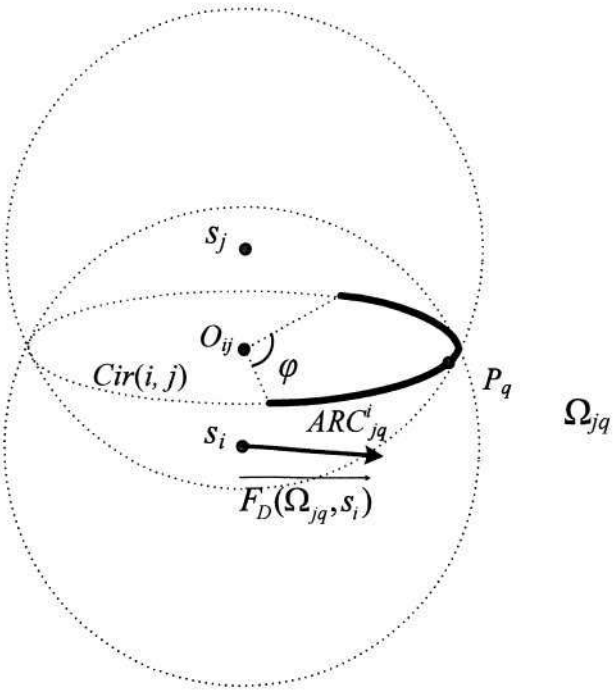


Figure 5.4: Attractive force of a 3-D potential field

Next we note that there may be k ($k \geq 1$) arcs in circle $Cir(i,j)$ which are not covered by any sensors beside s_i and s_j. In addition, s_i may intersect with m ($m \geq 1$) sensors. Thus the total attractive force executed by all potential fields can be calculated as a vector total:

$$\overrightarrow{F_D(s_i)} = \sum_{j=1}^{m} \sum_{q=1}^{k} \overrightarrow{F_D(\Omega_j q, s_i)} \tag{5.12}$$

5.5.7 Movement of Sensors

The total virtual force executed on a given sensor s_i can be calculated by a weighted combination of the four types of forces and the movement(reaction of forces) of a given sensor can be calculated as:

$$\overrightarrow{m_i} = C_A \cdot \overrightarrow{F_A(s_i)} + C_B \cdot \overrightarrow{F_B(s_i)} + C_C \cdot \overrightarrow{F_C(s_i)} + C_B \cdot \overrightarrow{F_D(s_i)} \tag{5.13}$$

where C_A, C_B, C_C and C_D is a set of constant factors used to control the degree of influence of each force. We will discuss the setting of these factors in Section 5.7.

5.6 Proposed Algorithm for MFDP

In this section, we will analyze the relationship between MCRP and MFDP and show that MFDP can be solved based on the solution of MCRP.

5.6.1 Relationship between MCRP and MFDP

Consider the following problem:

Problem 5.1: Minimum Sensing Radius Problem.

Given: the field in which sensors are initially placed Ω_s, the target field to be covered Ω_t, the sensor set $S = \{s_1, s_2, \ldots, s_n\}$, the initial locations of sensors $L_s = \{l_1, l_2, \ldots, l_n\}$.

Objective: determine the target location of all sensors $L_t = \{l'_1, l'_2, \ldots, l'_n\}$ to minimize the sensing radius of each sensor r, such that Ω_t is fully covered by S, i.e $C_e(S, \Omega_t) = \Omega_t$.

Lemma 5.2. MFDP is equivalent to Problem 5.1.

Proof: From equation (5.1), if the target field Ω_t is fully covered, then any point $P \in \Omega_t$ will have the first distance smaller or equal to the sensing radius r. On the other hand, in MFDP, the first distance of any point $P \in \Omega_t$ will be smaller than or equal to $D_{1st}(P_w, S)$. Thus the problem of minimizing the the sensing radius so that the target field can still be fully covered is equivalent to the problem of minimizing the first distance of the worst covered point. ■

Both MCRP and MFDP are optimization problems. To further explore the relationship between them, we can write the corresponding decision problem of these two problems as follows:

Problem 5.2: Decision problem of MCRP.

Given: the field in which sensors are initially placed Ω_s, the target field to be covered Ω_t, the sensor set $S = \{s_1, s_2, \ldots, s_n\}$, the sensing radius r, the initial locations of sensors $L_s = \{l_1, l_2, \ldots, l_n\}$, a predefined value ρ which satisfies: $0 < \rho \leq 1$. Can we determine the target location of all sensors $L_t = \{l'_1, l'_2, \ldots, l'_n\}$ so that $ratio_e \geq \rho$?

Problem 5.3: Decision problem of MFDP.

Given: the field in which sensors are initially placed Ω_s, the target field to be covered Ω_t, the sensor set $S = \{s_1, s_2, \ldots, s_n\}$, the sensing radius r, the initial locations of sensors $L_s = \{l_1, l_2, \ldots, l_n\}$. Can we determine the target location of all sensors $L_t = \{l'_1, l'_2, \ldots, l'_n\}$ so that Ω_t is fully covered by S?

By comparing Problem 5.2 and Problem 5.3, we can see that the decision problem of MFDP is a special case (where $ratio_e = 1$) of the decision problem of MCRP. Thus the solution of problem 5.2 will lead to the solution for Problem 5.3. Since we have proposed the solution for MCRP in Section 5.5, the decision version of MCRP, i.e Problem 5.2 can also be solved. Thus we can solve the decision version of MFDP, i.e Problem 5.3.

5.6.2 Proposed Algorithm for MFDP

As concluded in Section 5.6.1, the decision version of MFDP can be solved. In this section, we will show that the optimization version of MFDP can be solved using binary search. To apply the binary search algorithm, we introduce the idea of *virtual sensing radius*, which may not be equal to the sensing radius of sensors. In each iteration of binary search, the sensors will use the virtual sensing radius to replace the "real" sensing radius and calculate the virtual force. Initially, we can set the upper bound and lower bound of virtual sensing radius by considering the dimension of the field and the density of sensors. In each iteration, the coverage ratio $ratio_e$ will be calculated after applying the algorithm for MCRP. If the target field can be fully covered, i.e $ratio_e = 1$, the virtual sensing radius will be decreased and vice versa. This binary search will be terminated when the maximum number of iterations(denoted by num) is reached. The number of iterations can be set based the requirements of accuracy. We can note that each iteration will reduce the search space by half. Thus in practical cases, 10 iterations are enough to guarantee that the error is sufficiently negligible. The binary search process can be described using the pseudo code listed in Table 5.1.

This proposed algorithm addresses the two attributes described in Section 5.4.2:

- Since in each iteration of binary search, the algorithm proposed in Section 5.5 is applied, sensors which are initially placed outside of Ω_t will be moved into Ω_t;
- This algorithm detects the worst covered area in Ω_t by using the "bound and search" methodology. When a virtual sensing radius r is applied, the target field will be divided into two parts: $C_e(S, \Omega_t)$ and $\Omega_t - C_e(S, \Omega_t)$. It is easy to see that any point in $C_e(S, \Omega_t)$ will has its first distance smaller than or equal to r and any point in $\Omega_t - C_e(S, \Omega_t)$ will has the first distance larger than r. Thus we say $\Omega_t - C_e(S, \Omega_t)$ is the coverage hole and the worst covered area must be located in somewhere inside $\Omega_t - C_e(S, \Omega_t)$. By applying the algorithm proposed in Section 5.5, sensors will try to repair the coverage hole. If the coverage hole can be totally repaired, a smaller virtual sensing radius will

Table 5.1: Proposed algorithm for MFDP

```
Begin
set upper bound /* the upper bound of search space
set lower bound /* the lower bound of search space
set num /* the number of iterations
for i = 1 to num
   r = (upperbound + lowerbound)/2
    solve MCRP /* maximize the coverage ratio with algorithm proposed in Section 5.5
   if ratio_e < 1 /* Ω_t can not be fully covered
        lowerbound = r /* the virtual sensing radius will be increased
   else Ω_t can be fully covered
        upperbound = r /* the virtual sensing radius will be decreased
    endif
    i = i + 1
endfor
End
```

be applied and vice versa.

5.7 Optimizations

In this section, we will discuss several optimization issues for enhancing the performance of the proposed algorithms.

5.7.1 Factor Settings

As implied in (5.13), the reaction(movement) of sensors is controlled by a set of factors. The sensors are more "sensitive" to a particular force if the corresponding factor is set to be larger and vice versa. These factors are referred as "sensitivity factors". When the factors are set to be larger, the increase of coverage will be faster in the earlier stage. However, in the later stage, the coverage hole may be small and the sensors may move further than necessary due to the larger factors. In such instance, some new coverage holes may be created, this in turn reduce the coverage ratio. On the other hand, smaller factor settings will has a slower

increase of coverage in the earlier stage but it allows fine adjustment of sensor locations at later stage. A proper setting should balance these two issues and this balance can be determined through experimental studies.

5.7.2 Terminating Condition

As described in Section 5.5.1, the movement-assisted sensor deployment process will terminate when the termination condition is met. The termination condition can be defined as a maximum number of iterations. This number, denoted by Max_iter, can be used to control the tradeoff between deployment time and solution quality. The combination of large Max_iter and small sensitivity factors will results in a finer adjustment of sensor locations, which lead to a better coverage ratio. However, this will increase deployment time and consumes more energy.

In some practical applications, the coverage requirement can be met before reaching the maximum number of iterations. In such cases, the deployment process can be terminated before the maximum number of iterations is reached to reduce the deployment time and conserve energy.

5.8 Case Study and Simulation Results

We use Matlab 7.1 [105] to simulate the performance of proposed algorithms for MCRP and MFDP. The field which sensors are initially placed in and the target field are assumed to be continuous. In the calculation of coverage ratios for 2-D and 3-D cases, a grid-based model is adopted as an approximation of the continuous space due to the lack of optimal methods to calculate the space to be covered by sensors. In the grid-based model, the target field is divide into grids(for 2-D case) or cubes(for 3-D case) and we assume the coverage of any points in the same grid or cube are identical. We assume a grid/cube if covered if and only if the center point of this grid/cube is covered. A large number of grids or cubes(from

Table 5.2: Parameter settings of scenarios

Scenario	A	B	C
Ω_s(m)	400×600	100×100	$80 \times 80 \times 80$
Ω_t(m)	400	100×100	$50 \times 50 \times 50$
n	40	140	125
r(m)	5	6	8.6
R(m)	20	20	20
C_A, C_B, C_C, C_D	1,0.1,0.1,0.1	N/A,0.1,0.1,0.2	0.5,0.2,0.2,0.1

4000-10000)is used to ensure that a high degree of accuracy can be achieved.

5.8.1 Performance of algorithm for MCRP

The performance of algorithm proposed for MCRP is analyzed from two aspects: coverage quality and average total moving distance. The coverage quality is measured by the coverage ratio $ratio_c$, which is defined in Section 5.3.2. Average total moving distance is calculated by summing up the moving distance from the first iteration to the current iteration and then taking the average of all n sensors. We assume the sensors are initially uniformly distributed in Ω_s. The deployment process will be terminated when the maximum number of iterations Max_iter is reached, where Max_iter is set to be 30. We adopt three typical scenarios and the parameter settings of each scenarios can be found in Table 5.2.

In scenario A, Ω_s is a $400m \times 600m$ rectangular field. The target field is a $400m$ straight line placed in the center of Ω_s. As we can see in Figure 5.5, the coverage ratio of Ω_t increased dramatically in the first iteration, from 0.02 to 0.70. This is caused by the strong attractive force of Ω_t. The sensors are initially uniformly distributed in Ω_s; after the first iteration, all of them are "attracted" to Ω_t, or to the locations which are very close to Ω_t. This results in a dramatic increase of coverage ratio and a large moving distance. From the second iteration, since the sensors are already very close to Ω, the attractive force of Ω_t is no longer dominating. Thus the sensors will slightly adjust their locations along Ω_t and results in a gradual increase of coverage ratio and moving distance.

Scenario B is the case which is usually assumed by existing works, such as [109, 110].

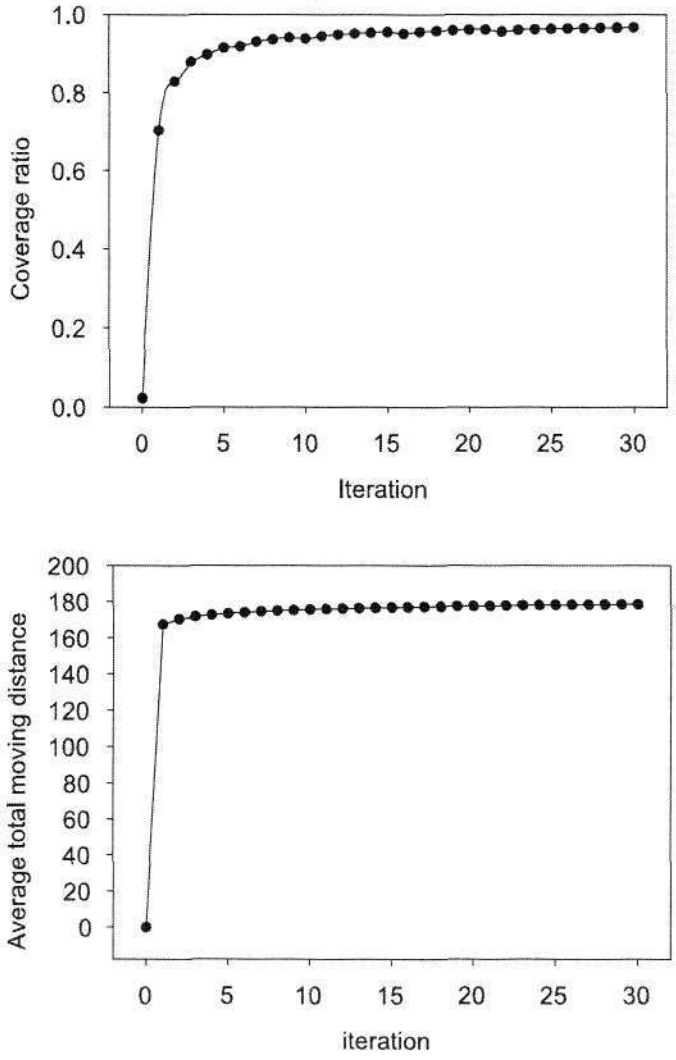


Figure 5.5: Coverage ratio and average total moving distance of scenario A

In this case, both Ω_s and Ω_t are set to be the same rectangular plane. Since all sensors are initially placed in Ω_t, the attractive force of Ω_t is not applicable. As we can see in Figure 5.6, the first move alone will enhance the coverage ratio from 0.78 to 0.90, while the average total moving distance is $2.77m$. The following iterations will further increase the coverage ratio at the cost of gradual increase of moving distance.

In scenario C, the Ω_t is a $50m \times 50m \times 50m$ cube and Ω_s is a larger$(80m \times 80m \times 80m)$ cube which include Ω_t. As shown in Figure 5.7, although the increasing speed of coverage ratio is slightly slower than scenario A and scenario B, we can still achieve a good coverage

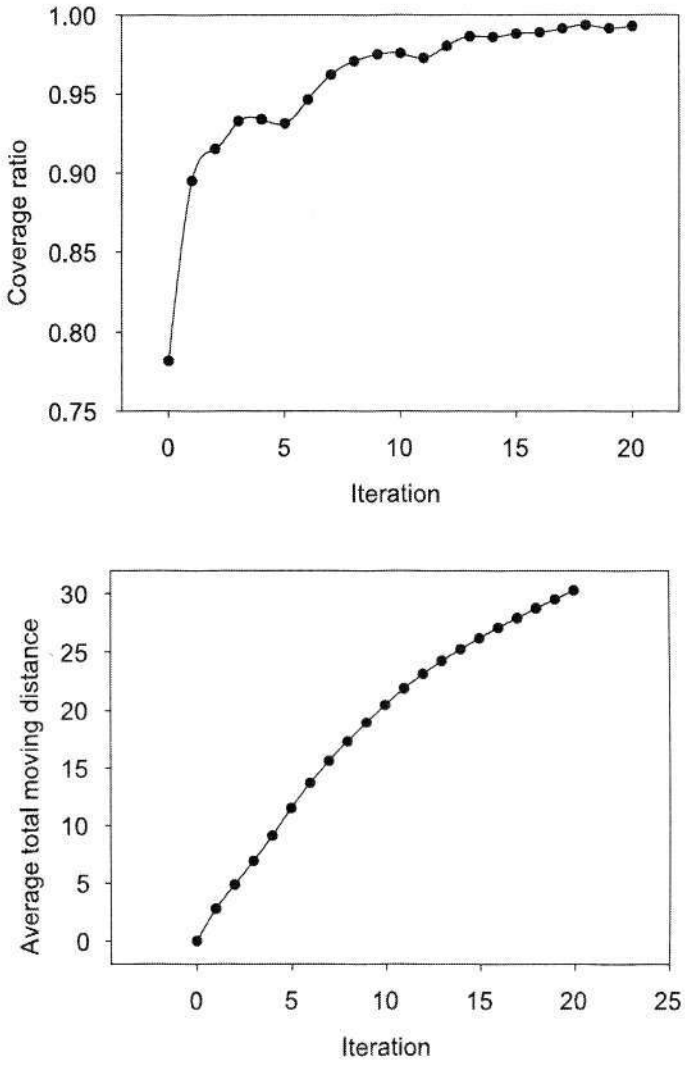


Figure 5.6: Coverage ratio and average moving distance of scenario B

quality while the moving distance is kept to a minimum.

From Figure 5.5-Figure 5.7, we can conclude our algorithm performs very well in 1-D, 2-D and 3-D cases.

5.8.2 Performance of algorithm for MFDP

The performance of algorithm proposed for MFDP is evaluated by the first distance of the worst covered point in the target field, denoted by $D_{1st}(P_w, S)$. We assume the sensors

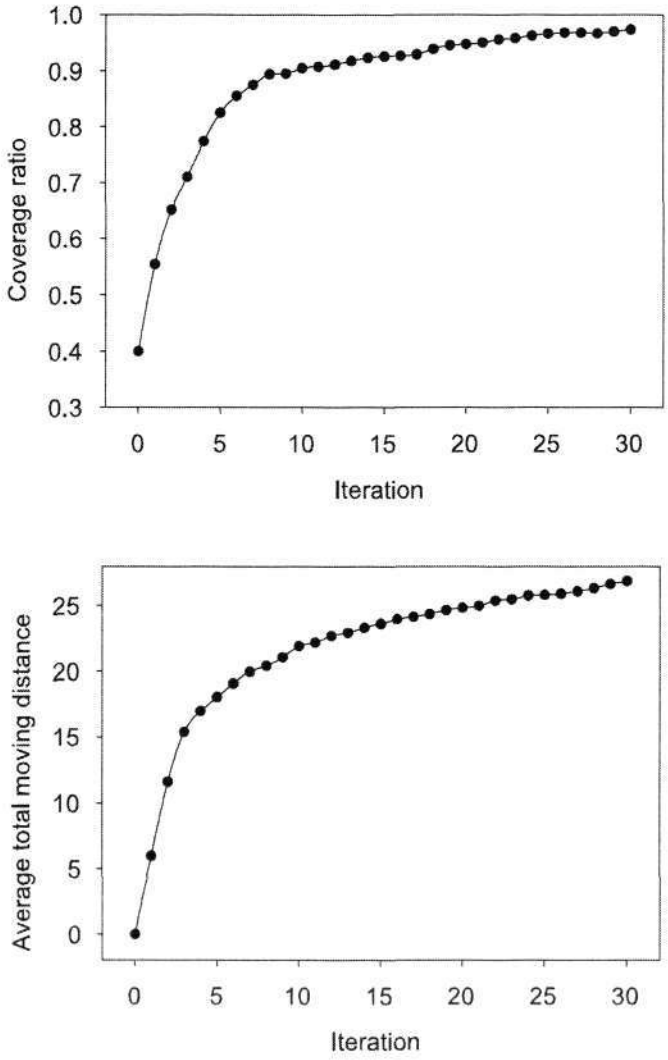


Figure 5.7: Coverage ratio and average moving distance of scenario C

are initially uniformly distributed in Ω_s. The binary search process will be executed for 10 iterations. Three scenarios described in Section 5.8.1 were adopted. The parameter settings of each scenarios and the $D_{1st}(P_w, S)$ achieved after 10 iterations of binary search are listed in Table 5.3. For comparison, the optimal values, denoted by $D'_{1st}(P_w, S)$ are also listed in Table 5.3. These optimal values are estimated using analytical methods.

By comparing the $D_{1st}(P_w, S)$ achieved by proposed algorithm and the optimal values, we can conclude that the proposed algorithm for MFDP is able to achieve close to optimal

Table 5.3: Parameter settings and simulation results

Scenario	A'	B'	C'
Ω_s(m)	400 × 600	100 × 100	80 × 80 × 80
Ω_t(m)	400	100 × 100	50 × 50 × 50
n	40	100	125
R(m)	40	40	40
C_A, C_B, C_C, C_D	1,0.1,0.1,0.1	N/A,0.1,0.1,0.2	0.5,0.2,0.2,0.1
$D_{1st}(P_w, S)$(m)	5.13	7.08	8.95
$D'_{1st}(P_w, S)$(m)	5.00	7.07	8.67

results, especially for 2-D case.

Figure 5.8 shows the convergence process of scenario B'. This process begins with a large virtual sensing radius (11.0m). At the beginning, the random placement of sensor results in a large $D_{1st}(P_w, S)$. This distance is reduced dramatically in the first few iterations, with coverage ratio $ratio_e$ increasing at the same time. When $ratio_e$ reaches 1 at 9th iteration, a smaller virtual sensing radius value (6.5m) is applied at 10th iteration. Since the new virtual sensing radius value is small, the target field cannot be fully covered when terminating condition is met(Max_iter is set to be 20). Thus a larger virtual sensing radius value (8.75m) is applied at 30th iteration. This results in a notable decrease in the first distance and a dramatic increase in coverage ratio. At 32th iteration, the coverage ratio $ratio_e$ reaches 1 and a smaller virtual sensing radius (7.0625m) is applied at 33th iteration. This binary search process continues until the search space is small enough to guarantee the desired level of accuracy.

We can also note that in this process, except for the first few iterations, noticeable reduction of $D_{1st}(P_w, S)$ usually take place one or two iterations after a new virtual sensing radius is applied. As described in Section 5.6.2, the proposed algorithm detect the worst covered area by using "bound and search" method. Since the worst covered area must be located in $\Omega_t - C_e(S, \Omega_t)$, a proper virtual sensing radius helps in detecting the worst covered area and leads to an efficient restoration of coverage for this area. This in turn results in a reduction of $D_{1st}(P_w, S)$. On the other hand, an improper virtual sensing radius may

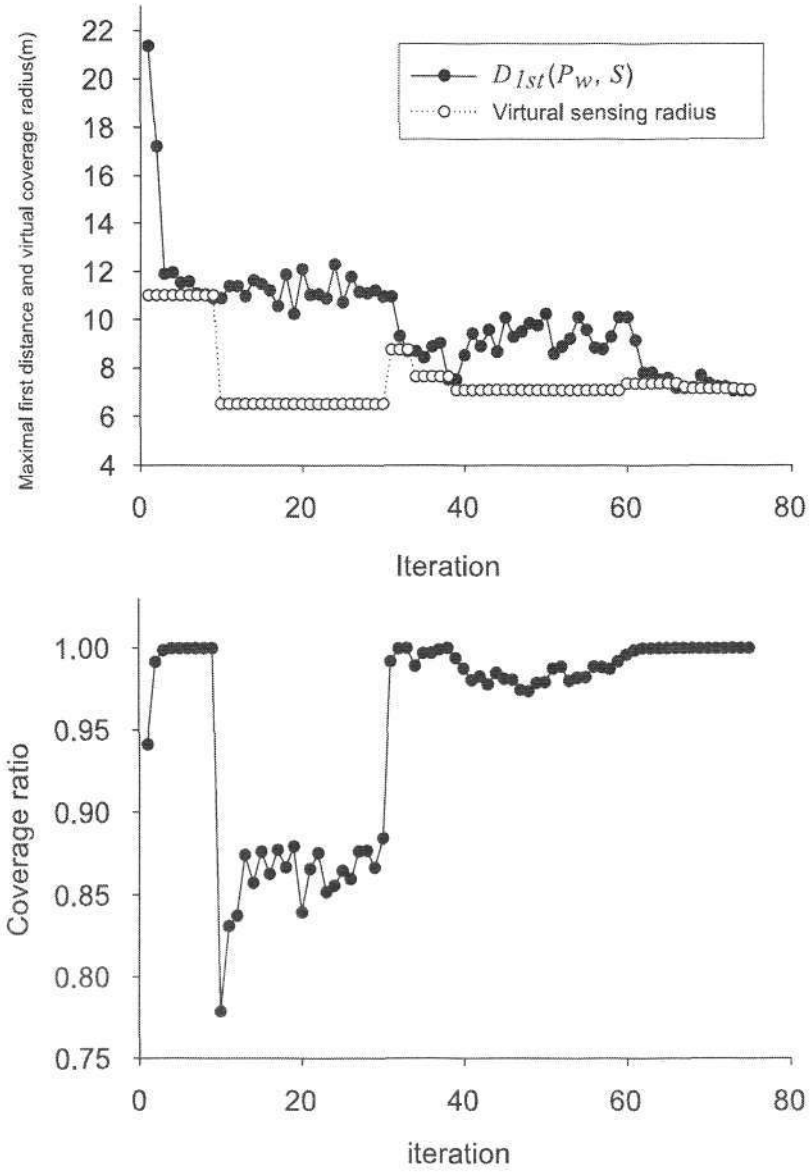


Figure 5.8: Convergence process of scenario B'

results in a larger $D_{1st}(P_w, S)$. As we can see in Figure 5.8, the value of $D_{1st}(P_w, S)$ has a notable increase from 38th iteration to 40th iteration, this increase is caused by the new sensing radius applied at 38th iteration. Thus the simulation results indicate that a proper virtual sensing radius plays an important role in searching for the minimal $D_{1st}(P_w, S)$. In the proposed algorithm for MFDP, we vary the virtual radius by using binary search, which

is an efficient way to search for the minimal $D_{1st}(P_w, S)$.

5.9 Summary

Movement-assisted sensor deployment is a promising method for enhancing the coverage quality of sensor networks. In this chapter, we formulated a generalized case of the movement-assisted sensor deployment problem which allows the target field to be a 1-D, 2-D or 3-D space. Two variations of the problem which have different optimization objectives were studied. We proposed a unified framework which captures some fundamental attributes that a movement-assisted sensor deployment scheme should have to enhance the coverage of the target field. Based on this framework, we proposed the efficient algorithms based on virtual force for both variations of the problem. Experimental study shows our algorithms work efficiently in enhancing the sensor coverage.

Chapter 6

Conclusions and Further Work

We conclude this thesis by summarizing our contributions, and recommending some directions for further research.

6.1 Conclusions

In this thesis, we have addressed some of the optimization problems which arise in communication networks. Two distinct optimization approaches, namely graph-theoretic approach and geometric approach were adopted. We addressed the wavelength converter placement problem in WDM networks and the clustering problem in WSNs by using graph-theoretic approach. Another two problems, the k-coverage problem and movement-assisted sensor deployment problem in WSNs were addressed by using geometric approach. We found that both approaches, although they are classic mathematic techniques, are very effective in solving the problems introduced by new structures and new applications in communication networks.

By noticing that wavelengths and wavelength converters are two most important resources in WDM networks, we addressed the problem of placing a minimum number of wavelength converters to guarantee that the number of wavelengths used will not exceed a given bound

αL, in Chapter 2. Unlike existing works, in our schemes, αL is a parameter, which can be defined by the network designers according to the overall availability of wavelength resources. Thus our schemes is more flexible compared with existing schemes which only provide fixed upper bounds for the wavelength usage. In addition, we took the traffic demand into consideration and we found this can reduce the redundant placement of wavelength converters. We first proved that this problem is NP-hard. Next we aim at designing effective approximate algorithms for this problem. We adopted the methodology of decomposing a given network modeled using a graph into some sub-networks which have special topologies. We found that the wavelength assignment problem in a sub-network with star topology can be transformed into the edge coloring problem by using a scheme proposed by us. Thus some existing results which have been proposed in the literatures can be applied to obtain a bound for the number of wavelengths required to support all lightpaths in a sub-network with star topology. We next proved that the number of wavelengths required by a network with tree topology, denoted by G_{tree}, will be bounded from above by αL if and only if the number of wavelengths required by each star sub-network of G_{tree}, is bounded from above by αL. Based on these observations and analysis, we proposed a two-step algorithm which can place a minimal set of wavelength converters in a WDM with arbitrary topology and the total wavelength usage is bounded. Experimental results showed that the algorithm proposed by us can achieve a flexible trade-off between the number of wavelengths required and the number of wavelength converters placed.

In WSNs, clustering can effectively enhance the system scalability and many efficient clustering schemes have been proposed in literatures. However, we found most of these schemes did not take the load-balancing issue into consideration. In addition, most existing studies put their attention on electing cluster heads, while the problem of assigning sensors to cluster heads has not been sufficiently addressed. By taking the above-mentioned two issues into consideration, in Chapter 3 we address the problem of assigning sensor nodes to cluster heads (gateways) to form clusters with the objective of minimizing the maximum

load of each gateway in the given network. We refer to this problem as LBCP. We first investigated a special case of LBCP whereby the traffic load from each sensor is the same. We found that the traffic load of a gateway can be reassigned to another gateway which has less traffic load by using a BFS tree, where this BFS tree can be constructed by the scheme we proposed. This reassignment procedure leads to an optimal polynomial time ($O(mn^2)$) algorithm for the special case of LBCP. We next examined the computational intractability of general case of LBCP and proved that LBCP is a NP-hard problem. This indicates that it is highly unlikely to propose any polynomial time optimal solution for this problem. We noted that each instance of LBCP can be modeled using a bipartite graph. Based on this observation, we proposed an greedy algorithm which always includes a maximum matching of this bipartite graph. This in turn ensures that the maximum possible number of gateways are utilized to distribute the traffic load. We proved that the algorithm proposed by us has a performance bound of $\frac{3}{2}$ and this bound is proved to be tight. Empirical studies shown that our proposed algorithm is able to perform even better on the average as compared to the worst-case performance ratio derived.

Since WSNs are designed for monitoring physical or environmental conditions, the extent of its coverage is an essential issue in any WSNs. In Chapter 4 and Chapter 5, we studied the coverage issue of WSN in two distinct aspects by using geometric techniques. In Chapter 4, we addressed the coverage problem from two different view points and refer to them as the worst-case and best-case coverage problems. Existing work on these two problems assumed that the coverage degree is one (i.e. the target area falls within the sensing range of at least one sensor). We found this is inefficient since some applications require the coverage degree be more than one. Thus we address the k-coverage problem, where the coverage degree is a user-defined parameter k. In other words, the problem addressed by us can be considered as a generalization of the existing work where only k=1 is assumed. We first extended the idea of growing disks and made it still be valid in the case of k-coverage. We then proposed a series of transformations, whose correctness are guaranteed by a set of definitions and

theorems proposed. With these transformations, we could reformulate both cases of the *k-coverage problem* into the problem of finding a sequence of adjacent borders. Then we proposed optimal polynomial time algorithms to solve both variants of the problem. Next we addressed two important extensions of the study on the k-coverage problem: the case that the coverage region of each sensor is not a disk and the distributed version of the algorithm. With these two extensions, the algorithms proposed by us can be applied in more practical scenarios.

In Chapter 4, we assume that the location of each sensor in WSN is fixed and evaluate the sensor coverage from two different point of view, namely worst-case and best-case coverage. In Chapter 5, we addressed another aspect of coverage problem: how to enhance the sensor coverage by relocating each sensors. This is referred to as movement-assisted sensor deployment problem. Existing works on this problem usually assume sensors were placed in a 2-*dimensional* (2-D) field. Further, they assume that the field in which the sensors are initially placed is identical to the target field. We found these assumptions may not valid in some practical scenarios. Thus we addressed the movement-assisted sensor deployment problem in a more general way. The initial deployment field and target field can be a 1-D, 2-D or 3-D space. Further, the initial deployment field and the target field are not required to be identical to each other. Two variations of the problem which have different optimization objectives, namely Maximizing Coverage Ratio Problem (MCRP) and Minimizing maximal First Distance Problem (MFDP) were formulated. For both variations, we first identified a set of basic attributes which can be used as guidelines for designing the movement-assisted sensor deployment schemes. Then based on the attributes we identified for MCRP, we combined the idea of "virtual force" with geometric techniques and proposed an efficient algorithm for MCRP. Next we analyzed the relationship between MCRP and MFDP and proved that the solution of MCRP can lead to an solution for MFDP, this solution satisfies the attributes we identified for MFDP. The efficiency of our algorithms in enhancing the sensor coverage has been shown in experimental studies.

6.2 Recommendations for Further Research

This research has explored the issue of solving some optimization problems that arise in WDM networks and WSNs by using graph-theoretic and geometric approaches. Our work showed the effectiveness of these two classic mathematical techniques in solving optimization problems in communication networks. Since ad hoc networks and optical networks are lately introduced types of communication networks, many problems are still open and the research in ad hoc networks and optical networks will continue to receive a lot of attention in the foreseeable future. In this section, we identify potential areas for the future work.

Firstly, our research works can be further extended, and some of the potential extensions are listed as follows:

1. Topology evaluation and design in WDM networks. In Chapter 2, we addressed the wavelength converter placement problem: given the topology and traffic demand, determine the minimum set of converter nodes so that the wavelength usage is bounded from above by αL. It is shown in observations and existing work that the topology of a network can affect the number of wavelengths required. Thus one of the problems faced by the network designers is: What is the optimal topology for a WDM network to support all traffic demand and to satisfy the connectivity requirements? The notion "optimal" here can be defined as minimizing the number of wavelength converters or minimizing the number of wavelengths required. We note similar problems have been addressed in some literatures, such as in [34, 42–44]. However, the results obtained in these literatures are incomplete. For example, [34, 43, 44] only consider some special topologies, [42] do not consider the use of wavelength converters. Thus systemic studies on the relationship between network topologies and performance and the design of optimal topologies for WDM networks are open research issues.

2. Multi-hop clustering in sensor networks. In Chapter 3, we assume there is only one hop between a sensor and its cluster head (gateway). This model is referred to as single-hop

clustering. Single-hop clustered networks just interconnect nodes that are within the same transmission range. This limitation can be overcome by adopt multi-hop clustering model. In multi-hop clustered sensor networks, nearby nodes can communicate directly by exploiting a single-hop wireless technology such as Bluetooth, 802.11, etc, while sensors which are not directly connected to gateways communicate with gateways by forwarding their traffic via a sequence of intermediate sensor nodes. We note that most of the existing clustering algorithms can not be applied in multi-hop sensor networks. Thus the clustering problem in multi-hop clustered sensor networks is a potential problem that need to be addressed.

3. Joint study of the load-balanced clustering problem. As introduced in Chapter 3, the clustering problem in sensor networks is composed of two phases: the cluster head(gateway) election and sensor assignment. In Chapter 3 we only focus on the second phase because that we found the first phase has been well studied by many literatures while the second phase has not been sufficiently addressed. However, we could also note that since the cluster head elected in the first phase will be considered as the design input in the second phase, the first phase will also affect the performance of load-balancing clustering. Thus it is interesting to carry on the research on the relationship between two phases. A joint solution of both phases is promising to achieve a better result than the solution which consider these two phases separately.

4. Redundancy evaluation and optimization in WSNs. In Section 4.6.2, we briefly introduced the application of proposed algorithms in evaluating the redundancy of coverage. Since the node failure rate in WSNs is considerably higher than in wired and centralized networks, "redundancy" of certain extent is desirable in WSNs. Here "redundancy" may include redundant coverage and redundant connectivity. Since a WSN should cope with node failures, a series of problems such as "How to evaluate the redundancy of a given WSN?", "What is the proper degree of redundancy?", "How to design and

implement a WSN to achieve a certain degree of redundancy?" arise naturally in the design and maintenance of WSNs. The solution of these problems may lead to great impact on the design of WSNs and thus are promising research directions.

5. Variable sensitivity factors. In the algorithms for movement-assisted sensor deployment problem proposed in Chapter 5, we use a set of factors, which are referred to as "sensitivity factors", to control the reaction(movement) of sensors. As implied in Section 5.7.1, the setting of these factors can affect the performance of proposed algorithms significantly. Larger factor setting will result in a faster increase of coverage ratio in the earlier stage and smaller factor setting allows adjustment of sensor locations in the later stage. One possible scheme to combine the advantages of both settings is to use variable sensitivity factors. In this scheme, the factors will be gradually decreased. In the earlier stage, larger factors can achieve a faster increase of coverage, and in the later stage, the smaller factors will allow a finer adjustment of sensor locations. By adopting such scheme, the performance of proposed algorithms may be further improved.

Furthermore, some other techniques can be adopted in solving the optimization problems in WSNs and MANETs. We note that a WSN is composed of a large number of sensor nodes, and they may be equipped with mobility. Thus a WSN can typically be modeled by particles in multidimensional space that have a position and a velocity. Interestingly, this is a perfect match to the essential features of an optimization technique which is referred to as Particle Swarm Optimization (PSO) [124, 125]. Thus some of the optimization issues in WSNs may be solved by using PSO methods. On the other hand, we note the nodes in a WSN or MANET have both cooperative(in routing, sensing, relaying messages) and competitive(in competition for resources) features. Thus the game theory [126, 127], which is a branch of applied mathematics, may help in solving some optimization issues arise in WSNs and MANETs.

Author's Publications

- Can Fang and Chor Ping Low, "On the Generalized Movement-Assisted Sensor Deployment Problem", *Submitted to IEEE Transaction on Vehicular Technology.*
- Can Fang and Chor Ping Low, "The k-coverage problem in wireless sensor networks", *Submitted to Elsevier Computer Networks.*
- Can Fang and Chor Ping Low, "A unified framework for movement-assisted sensor deployment", in *Proceedings of IEEE WCNC 2008*, pp. 2057-2062, March/April. 2008.
- Chor Ping Low, Can Fang, Jim Mee Ng and Yew Hock Ang "Efficient Load-Balanced Clustering Algorithms for Wireless Sensor Networks", *Elsevier Computer Communications*, vol. 2, issue 4, pp. 750-759, March. 2008.
- Can Fang and Chor Ping Low, "A Flexible Wavelength Converter Placement Scheme for Guaranteed Wavelength Usage", *Journal of Communications* (*JCM*), vol. 2, issue 1, pp. 34-43, 2007.
- Can Fang and Chor Ping Low, "Redundant Coverage in Wireless Sensor Networks", in *Proceedings of IEEE ICC 2007*, pp. 3535 - 3540 , June. 2007.
- Chor Ping Low and Can Fang, "Load-Balanced Clustering Algorithms for Wireless Sensor Networks", in *Proceedings of IEEE ICC 2007*, pp. 3485-3490, June. 2007.
- Can Fang and Chor Ping Low, "Optimal Wavelength Converter Placement with Guaranteed Wavelength Usage", in *Proceedings of IFIP Networking 2006*, pp. 1050-1061,

May. 2006.

Bibliography

[1] *http://www.internetworldstats.com/stats.htm.*

[2] P. Green, *Fiber-Optic Networks.* Prentice-Hall, Cambridge, MA, 1992.

[3] ——, "Optical networking update," *IEEE journal on Selected Areas on Communication*, vol. 14, pp. 764–779, June 1996.

[4] C. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems.* Prentice Hall, 2001.

[5] C. R. Murthy, *Ad Hoc wireless networks : architectures and protocols.* Prentice Hall, 2004.

[6] K. Tonuz, *Ad hoc wireless networks : a communication-theoretic perspective.* Wiley, 2006.

[7] S. Kazem, *Wireless sensor networks : technology, protocols, and applications.* Wiley-Interscience, 2007.

[8] X. Y. Li, *Wireless ad hoc and sensor networks : theory and applications.* Cambridge University Press, 2008.

[9] D. M. C. Carlos, *Ad Hoc and sensor networks : theory and applications.* World Scientific Publishing Co, 2006.

[10] K. Romer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54–61, Dec 2004.

[11] S. Hadim and N.Mohamed, "Middleware: middleware challenges and approaches for wireless sensor networks," *IEEE Distributed Systems Online*, vol. 7, no. 3, March 2006.

[12] T. Haenselmann, "An FDL'ed textbook on sensor networks," *www.informatik.uni-mannheim.de*, 2006.

[13] A. Hagit and J. Welch, *Distributed Computing: Fundamentals, Simulations, and Advanced Topics.* Wiley-Interscience, New York, 2004.

[14] T. Small and Z. Haas, "Quality of service and capacity in constrained intermittent-connectivity networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 6, no. 7, pp. 803–814, July 2007.

[15] L. Liu, P. Parag, and J.-F. Chamberland, "Quality of service and capacity in constrained intermittent-connectivity networks," *IEEE Transactions on Information Theory (TIT)*, vol. 53, no. 10, pp. 3833–3842, Oct. 2007.

[16] J. Elmirghani and H. Mouftah, "All-optical wavelength conversion technologies and applications in DWDM networks," *IEEE Communications Magazine*, vol. 38, no. 3, pp. 86–92, 2000.

[17] "Active media robotics, the high performance all-terrain robot," *http://www.activrobots.com/ROBOTS/*.

[18] J. Huang and M. Chen, "On the effect of group mobility to data replication in ad hoc networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 5, no. 5, pp. 492–507, May 2006.

[19] C. Shen, C. Srisathapornphat, R. Liu, Z. Huang, C. Jaikaeo, and E. Lloyd, "CLTC: a cluster-based topology control for ad hoc networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 3, no. 1, pp. 18–32, Jan-Feb 2004.

[20] K. Venugopal, M. S. Kumar, and P. S. Kumar, "A heuristic for placement of limited range wavelength converters in all-optical networks," *Computer Networks*, vol. 35, no. 2-3, pp. 143–163, Feb 2001.

[21] S. Thiagarajan and A. K. Somani, "Optimal wavelength converter placement in arbitrary topology wavelength-routed networks," *Computer Communications*, vol. 26, no. 9, pp. 975–985, 2003.

[22] M. Garay and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman company, New York, 1979.

[23] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization : algorithms and complexity.* Englewood Cliffs, N.J,Prentice Hall, 1982.

[24] G. Ausiello and M. Lucertini, *Analysis and design of algorithms in combinatorial optimization.* Wien, New York, Springer-Verlag, 1981.

[25] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data structures and algorithms.* Reading, Mass: Addison-Wesley, 1983.

[26] N. Biggs, E. Lloyd, and R. Wilson, *Graph Theory, 1736-1936.* Oxford University Press, 1986.

[27] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[28] R. Bellman, "On a routing problem," *Quarterly of Applied Mathematics*, vol. 16, pp. 87–90, 1958.

[29] L. Ford and D. Fulkerson, *Flows in Networks.* Princeton University Press, 1962.

[30] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, *The Traveling Salesman Problem: A Computational Study.* Princeton University Press, 2006.

[31] J. Kleinberg and A. Kumar, "Wavelength conversion in optical networks," *Journal of Algorithms*, vol. 1, no. 38, pp. 25–50, Mar. 2000.

[32] X. Jia, D. Du, X. Hu, H. Huang, and D. Li, "On the optimal placement of wavelength converters in WDM networks," *Computer Communications*, vol. 26, pp. 986–995, 2003.

[33] ——, "Optimal placement of wavelength converters for guaranteed wavelength assignment in WDM networks," *IEICE Transactions on Communication*, vol. E85-B, pp. 1731–1739, Sep 2002.

[34] G. Wilfong and P. Winkler, "Ring routing and wavelength translation," in *Proceedings of The 9th ACM-SIAM Symposium on Discrete Algorithm (SODA)*, 1998, pp. 333–341.

[35] M. Attygalle, Y. Wen, and A. Nirmalathas, "Cascaded operation of all-optical wavelength converters based on nonlinear optical loop mirror," *Lasers and Electro-Optics Society*, vol. 2, pp. 463–464, Nov 2002.

[36] Y. Yang and J. Wang, "Designing wdm optical interconnects with full connectivity by using limited wavelength conversion," *IEEE Transactions on Computers (ToC)*, vol. 53, no. 12, pp. 1547–1556, Dec 2004.

[37] H. Ngo, P. Dazhen, and Y. Yang, "Optical switching networks with minimum number of limited-range wavelength converters," *IEEE Transactions on Networking (ToN)*, vol. 15, no. 4, pp. 969–979, Aug 2007.

[38] Chlamtac, A. Ganz, and G. Karmi, "Lightpath communications: an approach to high bandwidth optical WAN's," *IEEE Transactions on Communications*, vol. 40, pp. 1171–1182, July 1992.

[39] X. Chu, B. Li, and I. Chlamtac, "Wavelength converter placement under different rwa algorithms in wavelength-routed all-optical networks," *IEEE Transactions on Communications*, vol. 51, no. 4, pp. 607–617, Apr 2003.

[40] B. Li, X. Chu, and K. Sohraby, "Routing and wavelength assignment vs. wavelength converter placement in all-optical networks," *IEEE Communication Magzine*, vol. 41, no. 8, pp. S22–S28, Aug 2003.

[41] X. Chu and B. Li, "Dynamic routing and wavelength sssignment in the presence of wavelength conversion for all-optical networks," *IEEE/ACM Transactions on Networking (ToN)*, vol. 13, no. 3, pp. 704–715, 2005.

[42] S. Baroni and P. Bayvel, "Wavelength requirements in arbitrary connected wavelength-routed optical networks," *Journal of Lightwave Technology*, vol. 15, no. 2, Feb 1997.

[43] T. Erlebach, K. Jansen, C. Kaklamanis, and P. Persiano, "Optimal wavelength routing in directed fiber trees," *Theoretical Computer Science*, vol. 221, pp. 119–137, 1999.

[44] P. Raghavan and E. Upfal, "Efficient routing in all-optical networks," in *Proceedings of The 26th Annual ACM Symposium on Theory of Computing (STOC)*, 1994, pp. 134–143.

[45] D. Hochbaum, *Approximation Algorithms for NP-hard Problems.* PWS Publishing Company, Boston, MA, 1997.

[46] P. Wan, O. Frieder, and L. Liu, "Optimal placement of wavelength converters in trees, tree-connected rings, and tree of rings," *Computer Communications*, vol. 26, no. 7, pp. 718–722, May 2003.

[47] T. Erlebarh and S. Stefanakos, "On shortest-path all-optical networks without wavelength conversion requirements," in *Proceedings of The 20th International Symposium on Theoretical Aspectes of Computer Science*, 2003, pp. 133–144.

[48] R. Ramaswami and K. Sivarajan, "Routing and wavelength assignment in all-optical networks," *IEEE/ACM Transactions on Networking (ToN)*, vol. 3, pp. 489–500, Oct 1995.

[49] K. Lee and V. Li, "A wavelength-convertible optical network," *Journal on Lightwave Technology*, vol. 11, pp. 962–970, May/June 1993.

[50] M. Kubale, "Graph colorings," 2004.

[51] D. König, "Über graphen und ihre anwendung auf determinantentheorie und mengenlehre," *Math. Ann*, vol. 77, pp. 453–465, 1916.

[52] C. E. Shannon, "A theorem on coloring the lines of a network," *Journal. Math. Phys*, vol. 28, pp. 148–151, 1949.

[53] V. G. Vizing, "On an estimate of the chromatic class of a p-graph," *Metody Diskret. Analiz*, vol. 3, pp. 9–17, 1964.

[54] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath communications: an approach to high bandwidth optical WAN's," *IEEE Transactions on Communications*, vol. 40, pp. 1171–1182, July 1992.

[55] V. Bafna, P. Berman, and T. Fujito, "A 2-approximate algorithm for the undirected feedback set problem," *SIAM Discrete Math*, vol. 12, no. 3, pp. 289–297, 1999.

[56] S. Jardosh and P. Ranjan, "A survey: Topology control for wireless sensor networks," in *Proceedings of International Conference on Signal Processing, Communications and Networking (ICSCN '08)*, Jan. 2008, pp. 422–427.

[57] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 3, no. 4, pp. 366–379, Oct-Dec 2004.

[58] C. Lin and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 15, no. 7, pp. 1265–1275, Sep 1997.

[59] T. C. Hou and T. J. Tsai, "A access-based clustering protocol for multihop wireless ad hoc networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 19, no. 7, pp. 1201–1210, 2001.

[60] M. Pearlman and Z. Haas, "Determining the optimal configuration for the zone routing protocol," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 17, no. 8, pp. 1395–1414, Aug 1999.

[61] U. Kozat, G. Kondylis, B. Ryu, and M. Marina, "Virtual dynamic backbone for mobile ad hoc networks," in *Proceedings of IEEE Internatoinal Conference on Communications (ICC 2001)*, vol. 1, 2001, pp. 250–255.

[62] A. McDonald and T. Znati, "A mobility-based framework for adaptive clustering in wireless ad hoc networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 17, no. 8, pp. 1466–1487, Aug 1999.

[63] W. Chen, J. Nitin, and S. Singh, "ANMP: ad hoc network management protocol," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 17, no. 8, pp. 1506–1531, Aug 1999.

[64] A. Iwata, C.-C. Chiang, G. yu. Pei, M. Gerla, and T.-W. Chen, "Scalable routing strategies for ad hoc wireless networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 17, no. 8, pp. 1369–1379, Aug 1999.

[65] J. Wu and H. Li, "On calculating connected dominating sets for efficient routing in ad hoc wireless networks," in *Proceedings of The 3rd ACM International Workshop on Discrete Algorithms and Methods for Mobile Computing*, Aug 1999, pp. 7–17.

[66] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 1, pp. 14–25, 2002.

[67] Y. P. Chen and A. L. Liestman, "Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, 2002, pp. 165–172.

[68] B. Han and W. Jia, "Clustering wireless ad hoc networks with weakly connected dominating set," *J. Parallel Distrib. Comput.*, vol. 67, no. 6, pp. 727–737, 2007.

[69] K. M. Alzoubi, P.-J. Wan, and O. Frieder, "Weakly-connected dominating sets and sparse spanners in wireless ad hoc networks," in *Proceedings of The 23rd International Conference on Distributed Computing Systems (ICDCS 03)*, 2003, p. 96.

[70] I. Stojmenovic, M. Seddigh, and J. Zunic, "A zonal algorithm for clustering ad hoc networks," *International Journal of Foundations of Computer Science*, vol. 14, no. 2, pp. 305–322, Apr 2003.

[71] C. Chiang, H. Wu, W. Liu, and M. Gerla, "Routing in clustered multihop, mobile wireless networks," in *Proceedings of IEEE Singapore International Conference on Networks*, 1997, pp. 197–211.

[72] J. Yu and P. Chong, "3hBAC (3-hop between adjacent clusterheads): a novel non-overlapping clustering algorithm for mobile ad hoc networks," in *Proceedings of IEEE Pacific Rim Conference on Communications, Computers and signal Processing (PACRIM 2003)*, vol. 1, Aug 2003, pp. 318–321.

[73] T. J. Kwon, M. Gerla, V. Varma, M. Barton, and T. Hsing, "Efficient flooding with passive clustering-an overhead-free selective forward mechanism for ad hoc/sensor networks," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1210–1220, Aug 2003.

[74] J. Wu, M. Gao, and I. Stojmenovic, "On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks," in *Proceedings of International Conference on Parrale Processing*, 2001, pp. 346–354.

[75] M. Yu, K. Leung, and A. Malvankar, "A dynamic clustering and energy efficient routing technique for sensor networks," *IEEE Transactions on Wireless Communications*, vol. 6, no. 8, pp. 3069–3079, Aug 2007.

[76] Z. Quan, A. Subramanian, and A. Sayed, "REACA: An efficient protocol architecture for large scale sensor networks," *IEEE Transactions on Wireless Communications*, vol. 6, no. 8, pp. 2924–2933, Aug 2007.

[77] T. Ohta, S. Inoue, and Y. Kakuda, "An adaptive multihop clustering scheme for highly mobile ad hoc networks," in *Proceedings of The 6th International Symposium on Autonomous Decentralized Systems*, 2003, pp. 293–300.

[78] A. Amis and R. Prakash, "Load-balancing clusters in wireless ad hoc networks," in *Proceedings of The 3rd IEEE Symposium on Application-Specific Systems and Software Engineering Technology*, 2000, pp. 25–32.

[79] R. Krishnana and D. Starobinski, "Efficient clustering algorithms for self-organizing wireless sensor networks," *Ad Hoc Networks*, vol. 4, no. 1, pp. 36–59, 2006.

[80] J. Lenstra, D. Shmoys, and E. Tardos, "Approximation algorithms for scheduling unrelated parallel machines," *Math Programming*, vol. 46, pp. 259–271, 1990.

[81] E. Shchepin and N. Vakhania, "Task distributions on multiprocessor systems," *Lecture Notes in Computer Science*, vol. 1872, pp. 112–125, 2000.

[82] C. Berge, "Two theorems in graph theory," *Proc. Nat. Acad. Sci.*, vol. 43, pp. 842–844, 1957.

[83] *http://www.ilog.com/products/cplex/.*

[84] S. Megerian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Worst and best-case coverage in sensor networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 1, pp. 84–92, Jan/Feb 2005.

[85] X. Li, P. Wan, and O. Frieder, "Coverage in wireless ad-hoc sensor networks," *IEEE Transactions on Computers*, vol. 52, no. 6, pp. 753–763, June 2003.

[86] S. Megerian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless sensor networks: Theory and practical solutions," *ACM Journal of Wireless Networks*, vol. 8, no. 5, pp. 443–454, September 2002.

[87] D. Mehta, M. Lopez, and L. Lin, "Optimal coverage paths in ad-hoc sensor networks," in *Proceedings of IEEE Internatoinal Conference on Communications (ICC 03)*, 2003, pp. 507–511.

[88] C. Huang and Y. Tseng, "The coverage problem in a wireless sensor network," *ACM Mobile Networks and Applications*, vol. 10, no. 4, pp. 519–528, Augest 2005.

[89] C. Huang, Y. C. Tseng, and L. Lo, "The coverage problem in three-dimensional wireless sensor networks," *Journal of Interconnection Networks*, vol. 8, no. 3, pp. 209–227, September 2007.

[90] Z. Zhou, S. Das, and H. Gupta, "Connected k-coverage problem in sensor networks," in *Proceedings of ICCCN 04*, 2004, pp. 373–378.

[91] G. Xing, X. Wang, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration for energy conservation in sensor networks," *ACM Transactions on Sensor Networks*, vol. 1, pp. 36–72, Aug 2005.

[92] P. Vashney, "Distributed detection and date fusion," Spinger-Verlag, New York, 1996.

[93] D. Li, K. Wong, Y. Hu, and A. Sayeed, "Detection, classification and tracking of targets in distributed sensor networks," *IEEE Signal Proceecing Magazine*, vol. 19, Mar 2002.

[94] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, "Computational geometry: Algorithms and applications," Spinger, New York, 1997.

[95] J. O'Rourke, "Computational geometry in C," Cambridge University Press, New York, 1998.

[96] J. Adriaens, S. Megerian, and M. Potkonjak, "Optimal worst-case coverage of directional field-of-view sensor networks," in *Proceedings of The 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks (SECON '06)*, vol. 1, Sep 2006, pp. 336–345.

[97] S. Meguerdichian, S. Slijepcevic, V. Karayan, and M. Potkonjak, "Localized algorithms in wireless ad hoc networks: Location discovery and sensor exposure," in *Proceedings of MobiHoc 01*, Oct 2001, pp. 106–116.

[98] L. Huang, H. Xu, Y. Wang, J. Wu, and H. Li, "Coverage and exposure paths in wireless sensor networks," *Journal of Computer Science and Technology*, vol. 21, no. 4, pp. 490–495, July 2006.

[99] S. Kumar, T. Lai, and J. Barlogh, "On k-coverage in a mostly sleeping sensor network," in *Proceedings of MobiCom 04*, 2004, pp. 144–158.

[100] P. Wan and C. Yi, "Coverage by randomly deployed wireless sensor networks," *IEEE Transactions on Information Theory*, vol. 52, pp. 2658–2669, June 2006.

[101] L. Yen, C. Yu, and Y. Cheng, "Expected k-coverage in wireless sensor networks," *Ad Hoc Networks*, vol. 4, pp. 636–650, Sep 2006.

[102] A. Boukerche and X. Fei, "Coverage-preserving scheme for wireless sensor network with irregular sensing range," *Ad Hoc Networks*, vol. 5, pp. 1303–1316, 2007.

[103] G. Mao, B. Fidan, and B. D. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, pp. 2529–2553, July 2007.

[104] B. Awerbuch and R. Gallager, "A new distributed algorithm to find breadth first search trees," *IEEE Transactions on Information Theory*, vol. 33, pp. 315–322, May 1987.

[105] *http://www.mathworks.com/.*

[106] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, Augest 1986.

[107] A. Howard, M. Mataric, and G. Sukhatme, "An incremental deployment algorithm for mobile robot teams," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and System*, vol. 3, 2002, pp. 2849–2854.

[108] G. Wang, G. Cao, and T. L. Porta, "A bidding protocol for deploying mobile sensors," in *Proceedings of The 11th IEEE International Conference on Network Protocols (ICNP)*, 2003, pp. 315–324.

[109] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization in distributed sensor networks," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, pp. 61–91, 2004.

[110] G. Wang, G. Cao, and T. L. Porta, "Movement-assisted sensor deployment," *IEEE Transactions on Mobile Computing*, vol. 5, no. 6, pp. 640–652, June 2006.

[111] S. Yang, M. Li, and J. Wu, "Scan-based movement-assisted sensor deployment methods in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, pp. 1108–1121, Aug. 2007.

[112] G. Wang, G. Cao, T. L. Porta, and W. Zhang, "Sensor relocation in mobile sensor networks," in *Proceedings of INFOCOM*, vol. 4, March 2005, pp. 2302–2312.

[113] R. Chang and S. Wang, "Self-deployment by density control in sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 3, pp. 1745–1755, May 2008.

[114] S. Chellappan, W. Gu, X. Bai, D. Xuan, B. Ma, and K. Zhang, "Deploying wireless sensor networks under limited mobility constraints," *IEEE Transactions on Mobile Computing*, vol. 6, no. 10, pp. 1142–1157, Oct 2007.

[115] N. Heo and P. Varshney, "Energy-efficient deployment of intelligent mobile sensor networks," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 35, no. 1, pp. 78–92, Jan 2005.

[116] G. Wang, M. Irwin, P. Berman, H. Fu, and T. L. Porta, "Optimizing sensor movement planning for energy efficiency," in *Proceedings of The 2005 International Symposium on Low Power Electronics and Design*, Aug 2005, pp. 215–220.

[117] J. Heidemann, Y. Wei, J. Wills, A. Syed, and L. Yuan, "Research challenges and applications for underwater sensor networking," in *Proceedings of IEEE Wireless Communications and Networking Conference*, vol. 1, 2006, pp. 228–235.

[118] E. Cayirci, H. Tezcan, Y. Dogan, and V. Coskun, "Wireless sensor networks for underwater survelliance systems," *Ad Hoc Networks*, vol. 4, pp. 431–446, 2006.

[119] S. S. Ram, D. Manjunath, S. K. Iyer, and D. Yogeshwaran, "On the path coverage properties of random sensor networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 5, pp. 446–458, May 2007.

[120] K. Chakrabarty, S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Transactions on Computers*, vol. 51, pp. 1448–1453, Dec 2002.

[121] ——, "Coding theory framework for target location in distributed sensor networks," in *Proceedings of International Symposium on Information Technology: Coding and Computing*, April 2001, pp. 130–134.

[122] J. Lee, D. Dharne, and S. Jayasuriya, "Potential field based hierarchical structure for mobile sensor network deployment," in *Proceedings of American Control Conference*, July 2007, pp. 5946–5951.

[123] D. Popa, H. Stephanou, C. Helm, and A. Sanderson, "Robotic deployment of sensor networks using potential fields," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, 2004, pp. 642–647.

[124] J. Kennedy and R. Eberhart, *Swarm Intelligence*. Morgan Kaufmann, 2001.

[125] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 58–73, Augest 2002.

[126] P. Dutta, *Strategies and games: theory and practice.* MIT Press, 1999.

[127] M. Osborne, *An introduction to game theory.* Oxford University Press, 2004.