ALGORITHMS FOR WIRELESS SENSOR NETWORKS

by

Min Kyung An

APPROVED BY SUPERVISORY COMMITTEE:

D. T. Huynh, Chair

Farokh B. Bastani

Weili Wu

S. Q. Zheng
Dedicated to my family.
ALGORITHMS FOR WIRELESS SENSOR NETWORKS

by

MIN KYUNG AN, BS, MS

DISSERTATION
Presented to the Faculty of
The University of Texas at Dallas
in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY IN
COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS
August 2013
ACKNOWLEDGMENTS

Most of all, I would like to express my deepest appreciation to my supervising professor, Dr. Dung T. Huynh. I have been extremely lucky to have him as my supervisor. Without his guidance and persistent help, this dissertation would not have been possible. He cared so much about my work, constantly gave me encouragement, and provided advice throughout my time as his student. He also taught me how to do research well, how to improve my writing skills, and how to work well in a team environment. Everything I learned from him will be my priceless experience for my future career as a computer scientist.

I would like to thank my Ph.D. supervising committee members, Dr. Farokh B. Bastani, Dr. Weili Wu and Dr. S. Q. Zheng, for their time and valuable suggestions. In addition, I would like to send my gratitude to Dr. András Faragó and Dr. Ding Z. Du for their support and advice.

I am so thankful for my friends, Dr. Trac N. Nguyen and Nhat X. Lam, for being my great teammates during my Ph.D. studies.

I would like to thank my love, Hyeonggak Kim, for always encouraging me and sharing this long journey with me. I would never have been able to finish my dissertation without him. He was always there cheering me up and stood by me through the good times and bad.

Last but not the least, I am grateful to my parents, Jaegeun An and Jeongseon Yang, for their powerful support and encouragement throughout my academic career. I also would like to thank my sisters, Eunkyoung and Sukyoung, for being my great friends, and my little sister, Obong, and little twin brothers, Ohsan and Yongsan, for being my great math tutors. In addition, I owe my gratitude to my beloved grandmother and maternal grandfather for their
endless love. I am sorry that I could not be there to say goodbye to them. Special thanks also should be given to my brother-in-law, Moonchang Yang, two cutest nieces, Jiyoon and Jihu, and my little adorable doggie, Joey.

March 2013
This dissertation was produced in accordance with guidelines which permit the inclusion as part of the dissertation the text of an original paper or papers submitted for publication. The dissertation must still conform to all other requirements explained in the “Guide for the Preparation of Master’s Theses and Doctoral Dissertations at The University of Texas at Dallas.” It must include a comprehensive abstract, a full introduction and literature review, and a final overall conclusion. Additional material (procedural and design data as well as descriptions of equipment) must be provided in sufficient detail to allow a clear and precise judgment to be made of the importance and originality of the research reported.

It is acceptable for this dissertation to include as chapters authentic copies of papers already published, provided these meet type size, margin, and legibility requirements. In such cases, connecting texts which provide logical bridges between different manuscripts are mandatory. Where the student is not the sole author of a manuscript, the student is required to make an explicit statement in the introductory material to that manuscript describing the student’s contribution to the work and acknowledging the contribution of the other author(s). The signatures of the Supervising Committee which precede all other material in the dissertation attest to the accuracy of this statement.
Data aggregation and Gossiping are two of the most crucial applications in Wireless Sensor Networks (WSNs) which have been the focus of many researchers. A main issue of data aggregation and gossiping is how to assign timeslots to nodes for interference-free data transmission. In this dissertation we study the problems of constructing minimum latency data aggregation schedules, known as the Minimum Latency Aggregation Scheduling (MLAS) problem, and gossiping schedules, known as the Minimum Latency Gossiping (MLG) problem, in two interference models, namely the graph model and the physical interference model also called the Signal-to-Interference-Noise-Ratio (SINR) model. We start by studying MLAS in the 2-dimensional (2D) graph model, and prove an $\Omega(\log n)$ approximation lower bound in metric model (Moscibroda and Wattenhofer, 2005). We also propose a heuristic as well as an $O(1)$-approximation algorithm. We then study the MLAS problem in the 3-dimensional (3D) graph and SINR models, prove its NP-hardness and propose two $O(1)$-approximation algorithms. We then investigate the MLG problem in the 2D graph and SINR models, and prove its NP-hardness and propose an $O(1)$-approximation algorithm.
Along with the MLAS and MLG problems, we also investigate a related problem, namely the Minimum Channel Assignment (MCA) problem, whose main issue is to compute a minimum channel assignment that yields a strongly-connected communication graph spanning all nodes such that the nodes assigned to the same channel can communicate without interference in the 2D SINR model. We show the NP-hardness of the MCA problem, and propose two $O(1)$-approximation algorithms.

Lastly, this dissertation contains a study of the Bounded-Degree Minimum-Radius Spanning Tree problem whose objective is to compute a spanning tree which satisfies certain constraints. Given a disk graph $G = (V, E)$ and an integer constant $\Delta^* \geq 2$, the problem is to construct a spanning tree $T = (V, E_T \subseteq E)$ of $G$ such that the radius of $T$ is minimized while the degree of any node in $T$ is at most $\Delta^*$. We show that this problem is NP-complete, and introduce a $(8, 4)$-bicriteria approximation algorithm for the problem on unit disk graphs. We then introduce a bicriteria approximation algorithm for the problem on disk graphs.
# TABLE OF CONTENTS

ACKNOWLEDGEMENTS .......................................................... v
PREFACE ................................................................. vii
ABSTRACT ............................................................... viii
LIST OF FIGURES ........................................................... xiv
LIST OF TABLES ............................................................. xvi

CHAPTER 1 INTRODUCTION .................................................. 1
  1.1 Introduction .......................................................... 1
  1.2 Problems in Wireless Sensor Networks .............................. 1
    1.2.1 Scheduling Problems .......................................... 2
    1.2.2 Backbone Construction for Wireless Sensor Networks ...... 4
  1.3 Outline of the Dissertation .......................................... 5

CHAPTER 2 MINIMUM LATENCY DATA AGGREGATION SCHEDULE IN WIRELESS SENSOR NETWORKS .................................................. 8
  2.1 Abstract .......................................................... 8
  2.2 Introduction ........................................................ 9
  2.3 Models and Definitions ............................................ 11
  2.4 Approximation Lower Bound ....................................... 13
  2.5 Heuristic Algorithm with Non-Uniform Power Levels ............ 17
  2.6 Nearly Constant Factor Approximation Algorithm with Uniform Power Level .................................................. 20
    2.6.1 Algorithm ..................................................... 20
    2.6.2 Analysis of Algorithm ....................................... 23
  2.7 Simulation Results ................................................ 24
  2.8 Conclusion ........................................................ 26

CHAPTER 3 MINIMUM LATENCY AGGREGATION SCHEDULING PROBLEM IN INTERFERENCE-AWARE 3-DIMENSIONAL WSNS .................................................. 27
  3.1 Abstract .......................................................... 27
3.2 Introduction ......................................................... 27
3.3 Preliminaries ....................................................... 32
  3.3.1 3D Network Models ........................................... 32
  3.3.2 Problem Definition ........................................... 34
  3.3.3 NP-Hardness of the MLAS Problem in 3D WSNs ................. 34
3.4 3-Dimensional Space Filling ...................................... 34
  3.4.1 Space Filling with Cubes ..................................... 35
  3.4.2 Space Filling with Hexagonal Prisms .......................... 35
3.5 Constant Factor Approximation Algorithms ......................... 36
3.6 Analysis .......................................................... 40
  3.6.1 Analysis of CBAS Algorithm ................................. 41
  3.6.2 Analysis of HPBAS Algorithm ............................... 45
3.7 Conclusion ......................................................... 46

CHAPTER 4 MINIMUM LATENCY GOSSIPING IN WIRELESS SENSOR NETWORKS ........................................ 47
  4.1 Abstract ......................................................... 47
  4.2 Introduction ..................................................... 47
  4.3 Preliminaries ..................................................... 51
    4.3.1 Network Model ............................................... 51
    4.3.2 Problem Definition ......................................... 53
  4.4 NP-Hardness ...................................................... 54
  4.5 Constant-Factor Approximation Algorithm ......................... 56
    4.5.1 Algorithm ................................................... 56
    4.5.2 Analysis .................................................... 60
    4.5.3 Simulation .................................................. 63
  4.6 Conclusion ....................................................... 66

CHAPTER 5 CONNECTIVITY PROBLEM IN WIRELESS SENSOR NETWORKS IN THE SINR MODEL ................................... 67
  5.1 Abstract ........................................................ 67
  5.2 Introduction ..................................................... 67
7.2.2 3-Dimensional Wireless Networks .................................. 111
7.2.3 Mobile Networking and Computing ............................... 111
REFERENCES ........................................................................ 112
VITA
For any $\alpha > 1$, even there is no collision, $v_1$ and $v_3$ cannot send data simultaneously because $v_2$ and $v_4$ are interfered by $v_3$ and $v_1$, respectively.

The corresponding MDAS instance

Network partition

Schedules computed by SDA, MDAB, and Cell Coloring where $\alpha = 1$ and nodes have a uniform power level. (a) Unit Disk Graph (b) SDA schedule (c) MDAB schedule (d) Cell Coloring schedule. Black nodes represent dominators, and gray nodes represent connectors.

Transmission ball and interference ball of $u$

Space-filling convex polyhedra

(a) 6-labeling $(k = 1)$ (b) Space-filling with 6-labeling

(a) 60-labeling $(k = 2)$ (b) Space-filling with 60-labeling

216-labeling $(k = 3)$

The inscribed ball that touches each face of the cube.

The corresponding geometric MLG instance

The corresponding geometric MCA instance

(a) 1-labeling $(l = 1)$ (b) 7-labeling $(l = 2)$ (c) 19-labeling $(l = 3)$ (d) Tessellation with 19-labeling

Example of an MIS that consists of black nodes. $c$ is the center node of $G$.

Example of the construction of a CDS which consists of the set of nodes that are connected by bold lines. Black nodes represent dominators and gray nodes represent dominatees.

Example of the construction of an initial tree $T_I$ whose edges are represented by bold lines. Black nodes represent dominators, gray nodes represent connectors, and white nodes represent dominatees.

Example of RRC (Phase 1). Black nodes represent dominators, gray nodes represent connectors, and white nodes represent dominatees.

Partitioning $u$’s broadcasting disk into 8 sectors.
6.6 Example of BTC (Phase 2). Black nodes represent dominators, and gray nodes represent connectors.

6.7 Example of Phase 3 for the case $|C_u| \geq 1$ with $\Delta^* = 4$. Black nodes represent dominators, gray nodes represent connectors, and white nodes represent dominatees.

6.8 Example of $sec(u, \theta, r)$ and $sec(u, \theta'', 2r)$, where $\theta = \frac{2\pi}{8}$ radian, $\theta' = \frac{\pi}{2}$ radian, and $\theta'' = \theta + \theta' = \frac{5\pi}{4}$ radian.

6.9 Bounding the number of connectors in a sector. Radius of innermost circle is $r$, radius of dotted circle is $1.5r$, and radius of outermost circle is $2r$.

6.10 Partitioning $u_m$’s broadcasting disk into $m$ concentric circles (disks) centered at $u_m$.

6.11 Example of Local Tree Construction (LTC-DG) with $\Delta^* = 4$. Black nodes represent dominators, and white nodes represent dominatees. Nodes filled with upward diagonals are dominatees that are selected as local sinks.

6.12 Bounding the number of connectors.
LIST OF TABLES

2.1 Average length of schedules computed by SDA, MDAB, and Cell Coloring algorithms ........................................... 25
4.1 Latencies of Algorithm 8 in Graph Model .......................................................... 65
4.2 Latencies of Algorithm 8 in SINR Model .......................................................... 65
5.1 The size of channel assignments produced by SBCA and HBCA algorithms . . . 84
CHAPTER 1
INTRODUCTION

1.1 Introduction

Wireless Sensor Networks consist of wireless sensor devices whose power source is usually in the form of a battery. These devices, which are also called wireless nodes, turn on their power to emit radio signals, or turn it off to conserve energy. The wireless sensor networks have been hugely welcomed into our lives with their capabilities to communicate between sensor nodes, to monitor nearby environmental conditions, and so on, using their radio signals. Such capabilities have found applications in various areas including health (to monitor disabled patients), military (for command, control, communication, computing, intelligence, surveillance, reconnaissance, and targeting system), home monitoring, and so on. Recent advances in such wireless sensor networks have led to the remarkable development of modern sensor devices over traditional ones. However, these tiny devices still have limited energy resources. This Ph.D. dissertation has focused on overcoming such hardware limitation of wireless sensor networks, and aimed to provide certain desired features of wireless sensor networks such as energy efficiency, low interference, and connectivity by designing efficient algorithms for optimization problems in wireless sensor networks.

1.2 Problems in Wireless Sensor Networks

This dissertation focuses on several problems in wireless sensor networks such as

- scheduling of wireless sensor nodes, and

- computing infrastructure for wireless sensor networks.
In the literature, wireless sensor networks are commonly modeled as graphs where any two nodes are connected via a communication edge if they are covered by each other’s transmission range. When considering problems on such networks, choosing the *interference model* is a crucial step. While a substantial amount of research results have been obtained for the *graph-based interference model*, recently, several researchers have started investigating the problems in the more realistic *physical interference model*, also known as Signal-to-Interference-Noise-Ratio (SINR), which, unlike the graph model, more adequately captures real world phenomena. As the SINR model has been introduced only recently, few works exist and algorithms that guarantee theoretical performances are scarce. The studies in this dissertation have been done for both interference models, and following is a summary of the problems studied.

### 1.2.1 Scheduling Problems

When a node sends its data to other nodes by emitting its radio signal, the transmission can be interfered by signals concurrently sent by other nodes, and therefore it has to re-transmit. Thus, scheduling wireless sensor nodes by assigning them appropriate *timeslots* such that nodes in the same timeslot do not interfere with each other is an important approach to conserve energy. The scheduling problem has been investigated by several researchers as it can be used for well-known crucial applications in wireless sensor networks. This dissertation investigates the scheduling problem for other applications such as *data aggregation* and *gossiping*. Along with the problems of assigning timeslots to nodes, another related problem is to assign appropriate *channels* to the nodes so that a strongly-connected topology can be obtained. This problem is known as the *Minimum Channel Assignment* problem, and it has also been the focus of many researchers since it represents the theoretically achievable efficiency of MAC layer protocols (Moscibroda and Wattenhofer, 2006). This dissertation also investigates the channel assignment problem.
Data Aggregation

Given a set of nodes on the plane, the objective of the problem is to assign appropriate timeslots to the sensor nodes to obtain a good (short) schedule by which data can be aggregated to a so-called base station (or sink) without any collision or interference. Our first investigation of the problem has been done in the graph-based interference model. We have shown that there is no approximation algorithm having an approximation ratio better than $\Omega(\log n)$ for the problem in the metric model (An et al., 2010a), where $n$ is the number of nodes deployed, and proposed an $O(1)$-approximation algorithm for the problem in the uniform power model where all nodes are assigned a uniform transmission power level (An et al., 2011). We then extended our study to the SINR model, and showed the first NP-hardness result in (Lam et al., 2011) and proposed two approximation algorithms with an $O(1)$-approximation ratios in the dual power model where the sensor nodes are assigned a high-power or low-power transmission level (An et al., 2012b). Subsequently, we focused on designing an algorithm that works in the graph-based interference model as well as the SINR model, and proposed a new approximation algorithm with an $O(1)$-approximation ratio in the non-uniform power model. This was the first $O(1)$-approximation algorithm in the non-uniform power model (Lam et al., 2013). Recently, we studied the problem in the more general 3-dimensional (3D) wireless sensor networks adopting both interference models, and proposed two $O(1)$-approximation algorithms which are the first results for 3D wireless sensor networks.

Gossiping

Given a set of nodes on the plane, the objective of the problem, which is also known as all-to-all broadcast, is to distribute the message of each node to all the other nodes in the network. Our approach in (An et al., 2010b) to solve the problem is to assign appropriate timeslots to the sensor nodes to obtain a good (short) schedule by which data can
be distributed without any collision or interference. We have studied the problem in the graph-based interference model and the SINR model. We have shown the first NP-hardness result in the SINR model, and proposed an $O(1)$-approximation algorithm that works in both interference models. In the collision-free graph model, i.e., the graph-based interference model concerning only collision, the approximation ratio is 224 which is currently the best ratio and is an improvement on the approximation ratio of 258 given by (Krzywdzinski, 2010). In the collision-interference-free graph model, i.e., the graph-based interference model concerning both collision and interference, and the SINR model, our results are the first $O(1)$-approximation algorithms.

Channel Assignment for Strong Connectivity

Given a set of nodes on the plane, the problem is to compute a channel assignment that yields a strongly connected communication graph spanning all nodes such that sender nodes in the same channel can communicate with its receivers without interference in the SINR model. Here, the complexity measure is the number of channels used in the assignment, and the objective is to minimize it. In (An et al., 2012a), we have not only proved the first NP-hardness result, but also proposed two approximation algorithms with $O(1)$-approximation ratios which are the first results in the literature.

1.2.2 Backbone Construction for Wireless Sensor Networks

A wireless network consists of wireless devices that communicate each other using wireless signals. Wireless Sensor Networks are one of such infrastructure-less wireless networks. Thus, given a graph which represents a wireless sensor network, it is desirable to compute e.g. a spanning tree which satisfies certain constraints on parameters such as node degree, diameter (or radius) or total cost, etc. Such subgraphs are useful because they can serve as a pre-defined network infrastructure for communication in wireless sensor networks.
Bounded-Degree Minimum-Radius Spanning Tree

The problems of computing a spanning tree which satisfies certain constraints are known as constrained spanning tree problems in the literature, and we have studied one of such problems, namely the Bounded-Degree Minimum-Radius Spanning Tree problem. Given a disk graph $G = (V, E)$ which represents a wireless sensor network and an integer constant $\Delta^* \geq 2$, the objective of the problem is to construct a bounded-degree minimum-radius spanning tree $T = (V, E_T \subseteq E)$ of $G$ such that the radius of $T$ is minimized while the degree of any node in $T$ is at most $\Delta^*$. We have studied the problem on disk graphs as well as unit disk graphs, while all existing works studied the problem in unit disk graphs only. We showed that the problem is NP-complete, and introduced an $(8, 4)$-bicriteria approximation algorithm for the problem on unit disk graphs which is a special case of disk graphs. This result is an improvement over the bound $(10, 7)$ obtained in (Ghosh et al., 2011). We then introduced a bicriteria approximation algorithm that computes a spanning tree of a disk graph, and this is the first result for disk graphs in the literature.

1.3 Outline of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 studies the problem of computing minimum-length schedules for data aggregation in the graph-based interference model. The result of this chapter was originally published in the paper entitled “Minimum Data Aggregation Schedule in Wireless Sensor Networks” in the Proceedings of the ISCA 23rd International Conference on Computer Applications in Industry and Engineering (CAINE 2010), and later extended for publication in the International Journal of Computers and their Applications (IJCA 2011). This chapter shows an $\Omega(\log n)$ approximation lower bound for the metric model under assumption that the nodes in a network can have non-uniform power levels. In addition, it introduces a heuristic algorithm with non-uniform power levels.
as well as a constant factor approximation algorithm with a uniform power level. It also compares the performances of the algorithms against the SDA algorithm proposed in (Chen et al., 2009).

While Chapter 2 studies the problem of computing minimum-length schedules for data aggregation in 2-dimensional (2D) wireless sensor networks, Chapter 3 studies the problem in the more general 3-dimensional (3D) wireless sensor networks adopting both interference models, graph-based interference model and SINR model. This chapter proposes two $O(1)$-approximation algorithms which are the first results for 3D wireless sensor networks.

Chapter 4 discusses the problem of computing minimum-length schedules for gossiping in both graph-based interference model and SINR model. The result of this chapter was published in the paper entitled “Minimum Latency Gossiping in Wireless Sensor Networks” in the *Proceedings of the International Conference on Wireless Networks (ICWN 2010)*. This chapter shows the first NP-hardness result in the SINR model, and proposes an $O(1)$-approximation algorithm that works in both interference models. In the collision-free graph model, the approximation ratio is 224 which is currently the best ratio and is an improvement on the approximation ratio of 258 given by (Krzywdzinski, 2010). In the collision-interference-free graph model and the SINR model, the results are the first $O(1)$-approximation algorithms.

Chapter 5 concentrates on computing a channel assignment that yields a strongly connected communication graph spanning all nodes such that sender nodes in the same channel can communicate with its receivers without interference in the SINR model. The result of this chapter was published in the paper entitled “Connectivity in Wireless Sensor Networks in the SINR Model” in the *Proceedings of the 20th Annual IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2012)*. This chapter shows the first NP-hardness result in the SINR model, and proposes two approximation algorithms with $O(1)$-approximation ratios, assuming the uniform power model, which are the first results in the literature.
Chapter 6 explores one of the constrained spanning tree problems, Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem. This chapter studies the problem on disk graphs as well as unit disk graphs, while all existing works studied the problem in unit disk graphs only. It shows that the problem is NP-complete, and introduces a $(8, 4)$-bicriteria approximation algorithm, which is an improvement over the bound $(10, 7)$ obtained in (Ghosh et al., 2011), for the problem on unit disk graphs. It then introduces a bicriteria approximation algorithm that computes a spanning tree of a disk graph, and this is the first result for disk graphs in the literature.

Chapter 7 contains some concluding remarks summarizing the studies of the chapters, and describing some open problems and future research directions.
CHAPTER 2
MINIMUM LATENCY DATA AGGREGATION SCHEDULE IN WIRELESS SENSOR NETWORKS

2.1 Abstract

Data aggregation is one of the crucial applications in Wireless Sensor Networks (WSNs). It therefore has been the focus of many researchers. An important issue concerning data aggregation is how to assign appropriate time slots to the nodes so that data transmission in the network is free of any collision and interference. As time efficiency and energy consumption are vitally important, a major problem is to find a minimum-length schedule that is collision-free and interference-free.

In this chapter, we study the Minimum Data Aggregation Schedule (MDAS) problem in WSNs. While other studies of the MDAS problem consider only the collision-free model (e.g., (Annamalai et al., 2003; Chen et al., 2009; Du et al., 2006; Huang et al., 2007; Kesselman and Kowalski, 2006)), our objective is to construct schedules in which data transmission is both collision-free and interference-free. We prove an \( \Omega(\log n) \) approximation lower bound for the MDAS problem in the metric model (Moscibroda and Wattenhofer, 2005) under the assumption that the nodes in a network can have non-uniform power levels. In addition, we introduce a heuristic algorithm with non-uniform power levels, and a nearly constant factor approximation algorithm with a uniform power level whose latency is bounded by \( O(\Delta + R) \), where \( \Delta \) is the maximum node degree of a network and \( R \) is the network radius. We compare their performances against the Shortest-Data-Aggregation (SDA) algorithm in (Chen et al., 2009).
2.2 Introduction

A Wireless Sensor Network (WSN) consists of a number of small-sized sensor nodes that monitor nearby environmental conditions and gather data. They communicate with each other through their radio signals. One of the main tasks of these sensor nodes is to collect data periodically and forward it to a destination called the sink node. This type of application is commonly known as data aggregation in the literature.

Despite recent advances in sensing technology, small-sized sensors still have limited energy resources, and one of the main issues in WSNs research is to reduce energy consumption thereby extending the network lifetime. This issue is related with data aggregation since efficient data aggregation can decrease the overall energy consumption in a WSN. An interesting approach is to assign time slots to sensor nodes to obtain a good schedule so that sensor nodes can avoid unnecessary retransmission using their limited power. Since the data collection occurs periodically, reducing the latency of the schedule, that is, constructing schedules with a minimum number of time slots, is also a fundamental issue in WSN research.

The problem of constructing minimum-length data aggregation schedules in the collision-free transmission model of WSN’s has been the focus of many researchers. (Annalamalai et al., 2003) developed a heuristic algorithm that constructs a tree with a schedule that assigns a code and a time slot to each node to communicate with its parent node. In (Du et al., 2006), an algorithm for finding a minimal-length schedule was proposed and it was claimed that there exists a schedule whose length is at most \( \min \{ \log_2 \left( \frac{|S|+2}{3} \right) + 1, 3 \log_3 \left( \frac{2|S|+3}{5} \right) + 2 \} \) for unit disk graphs, where \( S \) is a set of source nodes that have data to be sent to a sink node. Recently in (Chen et al., 2009), a \((\Delta - 1)\)-approximation algorithm for finding collision-free schedules with minimal length was designed, where \( \Delta \) is the maximum node degree of the network, and NP-hardness was also proved for the grid topologies. (Huang et al., 2007) introduced an algorithm based on maximal independent sets whose latency is bounded by \( 23R + \Delta - 18 \), where \( R \) is the network radius. As in WSNs, the problem of constructing...
minimal-length schedules has also been studied in Wireless Ad Hoc Networks. For instance, (Kesselman and Kowalski, 2006) introduced an algorithm that consumes at most $O(n \log n)$ time slots. The algorithms in (Annamalai et al., 2003; Chen et al., 2009; Huang et al., 2007) assumed that each node has a uniform power level. On the other hand, (Du et al., 2006; Kesselman and Kowalski, 2006) designed their algorithms under the assumption that nodes can have non-uniform power levels. In contrast to data aggregation, broadcasting data from a source node (base station) to all the other nodes in the collision-free transmission model of WSNs has been more widely studied. Very early in the 1980s and 1990s, this problem was already investigated by many researchers (Alon et al., 1991; Bar-Yehuda et al., 1987; Bruschi and Del Pinto, 1997). In (Gandhi et al., 2003), it was proved that constructing a minimum-latency broadcast schedule is NP-hard and an algorithm whose schedule length is at most $648R$ was introduced. Later, (Banik and Radhakrishnan, 2007) produced broadcast schedules using three algorithms with latencies at most $24R-23$, $16R-15$, and $R+O(\log R)$, respectively. These three algorithms are based on connected dominating sets, $k$-independent sets, and node coloring. While only collision was considered in those papers, (Calinescu and Tongngam, 2008) and (Chen et al., 2007) focused on interference as well. Given a transmission power level $d$ for each node, the interference range is defined as $\alpha d$ where $\alpha \geq 1$ is the interference factor. (Chen et al., 2007) introduced a heuristic algorithm that finds broadcast schedules for $\alpha > 1$ based on breadth-first search trees. The authors claimed an $O(\alpha^2)$ approximation ratio. Following (Chen et al., 2007), (Calinescu and Tongngam, 2008) presented an algorithm that produces an $O(\alpha R)$ schedule, where an $O(\alpha)$ approximation ratio was proved.

In this chapter, we study the problem of computing minimum-length schedules for data aggregation in WSNs. As in (Calinescu and Tongngam, 2008; Chen et al., 2007), we consider the issues of interference as well as collision. We show an $\Omega(\log n)$ approximation lower bound for the metric model under assumption that the nodes in a network can have non-uniform
power levels. In addition, we introduce a heuristic algorithm with non-uniform power levels as well as a nearly constant factor approximation algorithm with a uniform power level whose latency is bounded by $O(\Delta + R)$, where $\Delta$ is the maximum node degree of a network and $R$ is the network radius.

This chapter is organized as follows. In Section 2.3, we describe our network model and define the Minimum Data Aggregation Schedule (MDAS) problem. Section 2.4 establishes an $\Omega (\log n)$ approximation lower bound for the MDAS problem in the metric model. In Section 2.5, we introduce our heuristic, the Minimum Data Aggregation on Backbone tree (MDAB) algorithm. Section 2.6 contains the nearly constant factor approximation algorithm called Cell Coloring. In Section 6, we compare their performances against the SDA algorithm proposed in (Chen et al., 2009). Section 2.8 contains some concluding remarks.

2.3 Models and Definitions

A WSN is defined in the metric model as $(V, D, p)$ (Moscibroda and Wattenhofer, 2005), where $V$ represents a set of sensor nodes, $D : V \times V \rightarrow \mathbb{R}^+$ represents the distance function that satisfies the triangle inequality, and $p_{\text{max}} : V \rightarrow \mathbb{R}^+$ is the maximum power level function. For a constant $d > 0$, let $R^u_d = \{v | v \in V, D(u, v) \leq d\}$ denote the set of all nodes that can be reached by node $u$ using the power level $d$. Let $p : V \rightarrow \mathbb{R}^+$, $p(u) \leq p_{\text{max}}(u)$, $u \in V$, be a power assignment function. Two nodes $u, v$ can communicate only if they are in the coverage area of each other, i.e., $u \in R^v_{p(v)}$ and $v \in R^u_{p(u)}$. In this case, there is an undirected edge between $u$ and $v$. (Note we assume the bidirectional model.)

In this chapter we consider the collision-free model, where a collision (or conflict) is said to occur at a node $w$ if there exist nodes $u$ and $v$ such that $w \in R^u_{p(u)} \cap R^v_{p(v)}$. In this work, however, we also consider the interference model as well. For a constant $d > 0$, let $F^u_d = \{v | v \in V, D(u, v) \leq \alpha d\}$ where $\alpha \geq 1$ is the interference factor, i.e., $F^u_d$ denotes the set of all nodes in the interference range of node $u$ that is assigned the power level $d$. As defined
in (Calinescu and Tongngam, 2008), given a power assignment $p$, an interference is said to occur at a node $w$ if there exists nodes $u$ and $v$ such that $w \in R_{p(u)}^w \cap F_{p(v)}^w$. (See Figure 2.1.)

Figure 2.1. For any $\alpha > 1$, even there is no collision, $v_1$ and $v_3$ cannot send data simultaneously because $v_2$ and $v_4$ are interfered by $v_3$ and $v_1$, respectively.

The Minimum Data Aggregation Schedule (MDAS) problem in the metric model is defined as follows. A schedule is defined to be a sequence of time slots at each of which several nodes are scheduled to send its aggregated data to one of its neighbors, and every node can be scheduled as a sender only once. Formally, at each time slot $t$, we have an Assignment Vector $\pi_t = (s_1, r_1), ..., (s_k, r_k)$, in which $s_i$ is assigned to send its aggregated data to $r_i$, $1 \leq i \leq k$, without any collision or interference at node $r_i$. So, the power level of nodes $s_i$ and $r_i$ is defined to be $D(s_i, r_i) \leq \min\{p(s_i), p(r_i)\}$. This defines an undirected communication edge $(s_i, r_i)$, and $\pi_t$ induces an undirected communication graph denoted by $G_{\pi_t}$.

A schedule is a sequence of assignment vectors $\Pi = (\pi_1, \pi_2, ..., \pi_L)$ where $L$ is called the length of $\Pi$. Let $S \subseteq V$ be the set of source nodes, and $s \in V$ be the sink node. A schedule $\Pi$ is successful if data from all source nodes in $S$ can be relayed and aggregated at the destination node $s$. That is, for every node $v \in S$ there exists a subschedule $\pi_{i_1}, \pi_{i_2}, ..., \pi_{i_l}$, $1 \leq i_1 \leq i_2 \leq ... \leq i_l \leq i_L$ such that data in $v$ can be transmitted to the sink node $s$. In other words, each $\pi_{ij}$ contains an edge $e_{ij}$ such that $e_{i_1}e_{i_2}...e_{i_l}$ forms a path from $v$ to $s$. The MDAS problem is formally defined as follows:

**Input.** A set of nodes $V$, a distance function $D$ satisfying the triangle inequality, a maximum power level function $p$, a set of source nodes $S \subseteq V$, and a sink node $s \in V$.

**Output.** A successful schedule of minimum length.
2.4 Approximation Lower Bound

In this section, we prove an $\Omega (\log n)$ approximation lower bound for the MDAS problem in the metric model. From the reduction in the proof, it follows that MDAS is NP-complete.

**Theorem 2.4.1.** There is no approximation algorithm having an approximation ratio better than $\Omega (\log n)$ for the MDAS problem in the metric model unless $NP \subseteq DTIME (n^{\log \log n})$.

**Proof.** We construct a polynomial-time approximation-preserving reduction from the set cover problem that is known to be hard to approximate. Let $I_S$ be an instance of the set cover problem consisting of a collection $S$ of subsets of a finite set $E$ of elements. Let $n$ and $m$ denote the cardinalities of $E$ and $S$, respectively. A solution to $I_S$ is a subset $S' \subseteq S$ such that every element $e \in E$ is in at least one set $A \in S'$. Let $OPT_S$ denote an optimal solution to $I_S$.

Given the instance $I_S$ of the set cover problem, we construct in polynomial time an instance $I_M$ of the MDAS problem as follows. $I_M$ consists of $2n + (m + 1)n + 1$ nodes. The set of nodes $V$ in $I_M$ is partitioned into three subsets of nodes. The first subset contains only a single node $s_n$ as the sink node. The second subset $C_S$ and the third subset $C_E$ consist of $(m + 1)n$ and $2n$ nodes, respectively. All nodes in $C_E$ are source nodes. (See Figure 2.2.)

The set $C_S$ is broken down into $n$ layers of nodes. Each layer $k$ contains $m + 1$ nodes, namely $s^0_k, s^1_k, ..., s^m_k$. Consider the first layer of nodes in this set denoted by $C'_S$. The first $m$ nodes in $C'_S$ correspond to the $m$ sets in the collection $S$. For each node $a \in C'_S$, let $S(a)$ be the set in $S$ that $a$ represents. Similarly, the set $C'_E$ of the first $n$ nodes $C_E$ corresponds to the $n$ elements in $E$ of $I_S$. And $e(a)$ denotes the element in $E$ that corresponds to node $a$ in $C'_E$.

We now need to define the distance function $D$ on $V \times V$ for the metric model of the MDAS problem. Let $|S(a)|$ be the cardinality of $S(a)$. For $C'_S = \{a_0, a_1, ..., a_{m-1}\}$, without loss of the generality, we assume that $|S(a_0)| \geq |S(a_1)| \geq ... \geq |S(a_{m-1})|$. Letting $\beta = Min \{\alpha, 2\}$,
we define the distance between nodes $u, v \in V$ as follows: (notice that $s_0^n = s_1^n = \ldots = s_m^n = s_n$)

$$D(u, v) = \begin{cases} 
\left(\frac{\beta+1}{\beta}\right)^k m & u = s_i^k, \ v = s_{i+1}^k, \ 1 \leq i \leq m, \ 0 \leq k < n \\
\beta \left(\frac{\beta+1}{\beta}\right)^k m & u = s_i^k, \ v = s_j^k, \ 1 \leq i < j \leq m, \ 0 \leq k < n \\
m + k & u = a_k, \ v \in C'_E, \ e(v) \in S(u) \\
m & u = s_m^m, \ v \in C_E \setminus C'_E \\
2m & u, v \in C_E
\end{cases}$$

All other distances follow from symmetry or are induced by the shortest path metric. Specifically, notice that the distance between $s_1$ and a node $u \in C_E$ is at least $2m$ and at most $3m-1$. In addition, the distance between a node $u \in C_S$ and a node $v \in C_E$ for which $e(v) \notin S(u)$ is at least $2m$. We define the maximum power level for each node as follows:

$$p_{\text{max}}(u) = \begin{cases} 
2m - 1 & u \in C_E \\
\left(\frac{\beta+1}{\beta}\right)^k m & u = s_i^k, \ 0 \leq i \leq m, \ 0 \leq k < n \\
2m & u = s_n
\end{cases}$$

From this definition, observe that there does not exist any direct connection between nodes in $C_E$, nor between nodes in different columns of $C_S$. Similarly, there is no direct
connection between any node \( a \) in \( C'_E \) and nodes in \( C'_S \) whose corresponding sets do not contain \( e(a) \).

It has been shown in (Feige, 1998; Lund and Yannakakis, 1994) that the set cover problem cannot be approximated with an approximation factor better than \( \Omega(\log n) \) unless \( NP \subseteq DTIME(n^{\log \log n}) \). Let \( OPT_M \) be the optimal solution to instance \( I_M \). The following two implications will be shown in Lemma 2.4.1 and Lemma 2.4.2 below. For some constant \( \gamma \geq 2 \):

\[
|OPT_S| \leq \gamma \implies L(OPT_M) \leq n + n(\gamma + 1)
\]

\[
|OPT_S| > \ln(n)\gamma \implies L(OPT_M) > n(\ln(n)\gamma + 1)
\]

Thus, for the inapproximability factor \( r \) the following holds:

\[
r \geq \frac{n(\ln(n)\gamma + 1)}{n + n(\gamma + 1)} \geq \frac{\ln(n)\gamma + 1}{\gamma + 2} \geq \frac{\ln(n)}{2}
\]

In addition, we have that \( \ln|V| < \ln(2n + n(m + 1)) < 2\ln(n) + c \), for some constant \( c \). Substituting this into the above inequality concludes the proof of the theorem.

In order to prove Lemma 2.4.1 and Lemma 2.4.2, we need the following two facts.

**Fact 2.4.1.** While a node in \( C_S \) is sending, no other nodes can send data without any interference.

**Proof.** Considering a node \( s^i_k \) in \( C_S \) with \( p_{\text{max}}(s^i_k) = \left( \frac{\beta + 1}{\beta} \right)^k \), observe that this node can only connect directly with nodes in the layer immediately above or below. However, since sending data from \( s^i_k \) to \( s^i_{k-1} \) does not bring any benefits, the only possibility is that node \( s^i_k \) sends its data to node \( s^i_{k+1} \). Moreover, during this process, all nodes in the lower layers cannot receive data because they are in the coverage of \( s^i_k \). Thus, Fact 2.4.1 follows.

**Fact 2.4.2.** Given a solution \( Q_M \), we can obtain in polynomial time a transformed solution \( Q'_M \) in which all nodes \( v \in C_E \) are assigned transmission level to reach their nearest active neighbors and \( L(Q_M) = L(Q'_M) \), where active neighbors are ones scheduled in \( Q_M \).
Proof. Consider a feasible solution $Q_M$ in which there exists a node $v \in C_E$ which is assigned a power level to reach a different neighbor $x$ rather than its nearest one $u$. Observe that while $v$ is sending data to $x$, $u$ cannot receive from any other nodes since it is in the range covered by $v$. Thus, instead of sending to $x$, we transform $Q_M$ into $Q'_M$ by assigning a suitable power level to $v$ such that it can only send to $u$. One more problem is that $u$ may be scheduled to send data in the previous time slot. By Fact 2.4.1, it is straightforward to argue that moving this slot to the back of the schedule does not affect the result of the schedule. Therefore, without increasing the number of time slots, we have transformed $Q_M$ into a new feasible solution whose number of nodes in $C_E$, which are assigned the power level to reach a different neighbor rather than their nearest one, is reduced. By repeating this method, we will obtain a new feasible solution $Q'_M$ with the desired property. Since there are $n$ nodes in $C_E$, the transformation is obviously in polynomial time. \hfill \Box

Lemma 2.4.1. If the size of $OPT_S$ is $\leq \gamma$, then $L(OPT_M) \leq n + n (\gamma + 1)$.

Proof. According to Fact 1, we can split the schedule into two independent phases. In the first phase, all nodes in $C_E$ send their data to selected nodes in $C_S$. In the second phase, selected nodes in $C_S$ send received data to $s_n$ through all intermediate nodes sequentially.

Regarding the first phase, applying Fact 2.4.2, we obtain a solution where the power level of each node in $C_E$ is so assigned that it can reach its nearest neighbor among those nodes that correspond to the sets in $OPT_S$. Therefore, this solution requires at most $n$ time slots for this phase. In addition, it requires at least $n$ time slots for the last $n$ nodes in $C_E$ to send data to a node in $C_S$. Thus, in this solution we need exactly $n$ time slots for this phase.

According to Fact 2.4.1, the number of time slots required for the second phase is $n (\gamma + 1)$. Since $OPT_M$ is a minimum-length schedule, the lemma follows. \hfill \Box
Lemma 2.4.2. If the size of $OPT_S$ is $> \ln(n)\gamma$, then $L(OPT_M) > n(\ln(n)\gamma + 1)$.

Proof. Observe that nodes in the first layer of $C_S$ can only send their data to the sink node sequentially through the nodes in their column. And for each starting node in the first layer of $C_S$, it requires at least $n$ steps to reach the sink node. Thus, if the size of $OPT_S$ is $> \ln(n)\gamma$, then at least $\ln(n)\gamma + 1$ such nodes in the first layer of $C_S$ are needed to collect data from all nodes in $C_E$, $L(OPT_M)$ is $> n(\ln(n)\gamma + 1)$.

2.5 Heuristic Algorithm with Non-Uniform Power Levels

In this section, we introduce our heuristic algorithm called the Minimum Data Aggregation on Backbone tree (MDAB) algorithm for the MDAS problem under assumption that the nodes in a network can have non-uniform power levels.

The MDAB algorithm is based on the SDA algorithm (Chen et al., 2009) with uniform power that aggregates data along the shortest path towards the sink node when $\alpha = 1$. MDAB algorithm starts by first constructing a connected graph $G = (V, E)$ in which each edge $(u, v) \in E$ is generated if a node $u$ is reached by another node $v$ with power level $p_{\text{max}}(v)$ and $v$ is reached by $u$ with power level $p_{\text{max}}(u)$. As the SDA algorithm, MDAB also finds a shortest path tree $T'$ rooted at $s$ in $G = (V, E)$ (Chen et al., 2009) from which MDAB computes a new tree $T$ called backbone tree. This backbone tree is constructed by determining a backbone $B$ that is a path from the sink $s$ to one of leaves on $T'$. In Figure 2.4(b), the tree represents the shortest path tree rooted at the sink node on the left top corner and the bold lines represent the backbone on that tree. In order to find the backbone $B$, MDAB starts from the sink $s$, and chooses one of its children nodes that has a maximum number of descendants on $T'$. After that child node is found, MDAB finds another node of its children nodes that has a maximum number of descendants, again. MDAB repeats this procedure until it reaches a leaf node on $T'$. 
Algorithm 1 MDAB Algorithm

**Input:** A set $V$ of nodes that can be assigned non-uniform power levels  
**Output:** Length of schedule

1. $t \leftarrow 1$, $R_0 \leftarrow \emptyset$, $S_0 \leftarrow \emptyset$, $T_0 \leftarrow$ backbone tree of $G$
2. for each edge $v \in V$ do
3. $p_0(v) \leftarrow 0$
4. end for
5. while $T_t \neq \{d\}$ do
6. $T_t \leftarrow T_{t-1} - \bigcup_{i=0}^{t-1} S_i$, $R_t \leftarrow \emptyset$, $S_t \leftarrow \emptyset$
7. $X_t \leftarrow$ leaves of $T_t - \{d\}$, $Y_t \leftarrow$ the parent nodes of nodes $x \in X_t$
8. for each $x_t \in X_t$ do
9. Let $x_t$ be such that $|Y_t \cap R_{D(x_t,y_t)}^x|$ is largest.
   If more than one such $x_t$ exists, then pick an $x_t$ such that $|X_t \cap R_{D(x_t,y_t)}^x|$ is largest.
10. $y_t \in Y_t \leftarrow$ a parent node of $x_t$
11. $p_t(x_t) \leftarrow D(x_t, y_t)$, $p_t(y_t) \leftarrow D(x_t, y_t)$
12. if there do not exist any node $u \in S_t$ or $v \in R_t$ that can cause collision or interference at $y_t$ or $x_t$ then
13. $R_t \leftarrow R_t \cup \{y_t\}$, $S_t \leftarrow S_t \cup \{x_t\}$, $X_t \leftarrow X_t - \{x_t\}$
14. else
15. $X_t \leftarrow X_t - \{x_t\}$, $p_t(x_t) \leftarrow 0$, $p_t(y_t) \leftarrow 0$
16. end if
17. end for
18. $t \leftarrow t + 1$
19. end while
20. return $L \leftarrow t - 1$

Once MDAB obtains a backbone $B$ from the shortest path tree $T'$, the backbone tree $T$ is computed as follows. If any node $v_c \notin B$ whose parent node $v_p$ is in $B$ has a shorter edge to any other node $v_i \in B$ than its original edge $(v_c \rightarrow v_p)$, then disconnect $v_c$ and $v_p$, and connect $v_c$ and $v_i$. In Figure 2.4(c), a new tree is formed by modifying the bold edges. The MDAB algorithm computes a schedule based on this new backbone tree $T$.

The main idea of finding a schedule on the backbone tree instead of the shortest path tree is that the backbone can be thought of as the most crowded part in the tree as it has many descendants. Therefore, the nodes connected to the backbone may increase the length of schedule. By modifying the edges, some nodes may be assigned lower power levels which may reduce interference.
On a backbone tree $T$ obtained from the graph $G$, a number of iterations are performed to find a schedule of minimal length. Algorithm 1 shows the details. At the beginning of each iteration $t$, MDAB starts with a subtree $T_t$ that is $T_{t-1}$ minus those nodes that sent data at the time slots $1, \cdots, t-1$ (Step 6). MDAB selects the set of leaves $X_t$ of $T_t$ and the set of parent nodes $Y_t$ of the leaves in $X_t$. The leaves of $T_t$ already have the aggregated data sent from all nodes in $\bigcup_{i=0}^{t-1} S_i$, where $S_i$ is the set of nodes that sent their data at the time slot $i$ (Steps 7). In Steps 8–17, MDAB examines all the leaves in $X_t$ to decide which of these leaves should be sender nodes in the current time slot. MDAB chooses a leaf node $x_t \in X_t$ such that $|Y_t \cap R^{x_t}_{D(x_t,y_t)}|$ is largest. If more than one such $x_t$ exist, then it chooses one of the nodes such that $|X_t \cap R^{x_t}_{D(x_t,y_t)}|$ is largest. The idea behind this selection criterion is that those nodes that potentially cause collision and interference are assigned to send data earlier than the others. Steps 11–16 ensure that the selection of $(x_t, y_t)$ does not cause collision or interference. MDAB repeats this procedure until only the sink node $s$ is left.

The SDA algorithm (Chen et al., 2009) produces a schedule whose length is at most $\min\{ (\Delta - 1)h + 1, (\Delta - 1) \cdot L(OPT) \}$, where $\Delta$ is the maximum node degree of $G$, $h$ is the height of the shortest path tree rooted at $s$, and the length of optimum schedule $OPT$ of MDAS problem is $\geq \max\{ h, \log_2 |S| \}$. Note that the MDAB algorithm is based on the SDA algorithm, and the differences between these two algorithms are (1) the selection criterion for choosing leaves in each $T_t$ and (2) the height $h$ of the backbone tree can be at most 1 larger than that of the SDA algorithm. This is because some of the leaves of the original shortest path tree may be forced to be connected with a leaf of backbone tree. Moreover, the criterion for choosing leaves in each $T_t$ does not affect the argument for the bound in (Chen et al., 2009). Thus, a similar bound applies to the MDAB algorithm as well.
2.6 Nearly Constant Factor Approximation Algorithm with Uniform Power Level

In this section, we introduce a nearly constant factor approximation algorithm called Cell Coloring for the MDAS problem with a uniform power level.

2.6.1 Algorithm

We assume a uniform power level $P$ that guarantees network connectivity. Given $P$, a connected unit disk graph $G = (V, E)$ is constructed where two nodes $u, v$ are connected by an edge $(u, v) \in E$ if $u$ and $v$ can reach each other using power level $P$. Let $r$ be the maximum length of edges in $G$, and assume that $P = r$. The algorithm starts by partitioning the network into cells, each of which has side length $r/\sqrt{2}$ (See Figure 2.3). This induces a grid where the upper-left corner has coordinates $(1, 1)$. A cell is denoted by Cell-ID $C(x, y)$ if its upper-left corner has coordinates $(x, y)$. We then construct a data aggregation tree as done in (Li et al., 2009).

![Figure 2.3. Network partition](image)

For the data aggregation tree construction, (Li et al., 2009) first finds a breadth first search (BFS) tree (cf. (Cormen et al., 2009)) rooted at a center node $v_c$ in order to obtain a latency bound in terms of the network radius $R$ rather than its diameter $D$. A node $v$ can be
chosen as the center node if the distance from \( v \) to the most distant node from \( v \) is minimum. Once the data is aggregated to the center node \( v_c \), it sends the received data to the sink node \( s \) via a shortest path. This takes only \( O(R) \) time slots. Based on the BFS tree, (Li et al., 2009) finds a Maximal Independent Set (MIS) using an algorithm in (Wan et al., 2002). (A subset \( V' \subseteq V \) of a graph \( G = (V, E) \) is said to be independent if for any vertices \( u, v \in V' \), \( (u, v) \notin E \). An independent set is said to be maximal if it is not a proper subset of another independent set.) We call the nodes in MIS dominators, and the others dominatees. The MIS constructed by (Wan et al., 2002) guarantees that the hop distance between any pair of its complementary subsets is exactly two hops. Therefore, in each cell we can have at most 1 dominator. To obtain a Connected Dominating Set (CDS) of \( G \), (Li et al., 2009) connects the dominators using some connectors that were originally dominatees. (A dominating set (DS) is a subset \( V' \subseteq V \) such that every vertex \( v \) is either in \( V' \) or adjacent to a vertex in \( V' \). A DS is said to be connected (CDS) if it induces a connected subgraph.) Since there may exist dominatees that are not connected to the CDS, (Li et al., 2009) connects each of the remaining dominatees to its neighboring dominator that has the smallest hop-distance to the root of BFS tree. We denote the newly formed tree by \( T_{\text{Agg}} \), and use it as the data aggregation tree in our algorithm.

Once the aggregation tree is obtained, a number of iterations are performed to assign time slots to the nodes. The scheduling is based on a constant \( H \) which guarantees that two senders can transmit data without any interference and collision if they are at least \( H \) hops apart from each other. The details are contained in Algorithms 2 and 3.

Our Cell Coloring algorithm is based on (Li et al., 2009). It schedules the nodes level by level on \( T_{\text{Agg}} \) starting with the dominatees so that they can send data successfully to their upper level dominators (Steps 4–8 in Algorithm 2). Algorithm 2 uses Algorithm 3 as a subroutine to assign the same time slots to nodes if they are \( H \) hops away from each other. After all dominatees are scheduled, only dominators and connectors are left. In Steps 9–17 in
Algorithm 2 Cell Coloring Algorithm

**Input:** A set $V$ of nodes with a uniform power level $P$, and maximum link length $r$

**Output:** Length of schedule

1: Partition a network into cells each of which has side length $\frac{r}{\sqrt{2}}$.
2: Construct an aggregation tree $T_{Agg}$ using an algorithm in (Li et al., 2009).
3: Set the starting time slot $t \leftarrow 1$
4: Let $S \subseteq V$ be the set of dominatees.
5: while $S \neq \emptyset$ do
6: Pick one dominatee as a sender in a cell. Let $S' \subseteq S$ be the set of such dominatees.
7: $t \leftarrow \text{AssignTimeSlot}(S', t)$
8: end while
9: for $i = R$ to 2 do
10: Let $S_i \subseteq V$ be the set of dominators at level $i$ of $T_{Agg}$.
11: $t \leftarrow \text{AssignTimeSlot}(S_i, t)$
12: Let $S_{i-1} \subseteq V$ be the set of connectors at level $i - 1$ of $T_{Agg}$.
13: while $S_{i-1} \neq \emptyset$ do
14: Pick one connector as a sender in a cell. Let $S'_{i-1} \subseteq S_{i-1}$ be the set of such connectors.
15: $t \leftarrow \text{AssignTimeSlot}(S'_{i-1}, t)$
16: end while
17: end for
18: return $t - 1$

Algorithm 3 AssignTimeSlot

**Input:** A set $S$ of sender nodes, a starting time slot $t$

**Output:** Time slot $t$

1: for $t_1 = 0, \ldots, H - 1$ and $t_2 = 0, \ldots, H - 1$ do
2: Let $S' \subseteq S$ be the set of nodes with Cell-ID $C(x, y)$ such that $t_1 = x \mod H$, and $t_2 = y \mod H$
3: for each node $v \in S'$ do
4: Assign time slot $t$ to $v$
5: $S' \leftarrow S' \setminus \{v\}$
6: end for
7: $t \leftarrow t + 1$
8: $S \leftarrow S \setminus S'$
9: end for
10: return $t$

Algorithm 2, all dominators and connectors are scheduled by first assigning time slots to all dominators at level $i$ so that they can send data successfully to the connectors at level $i - 1$
(Steps 10–11 in Algorithm 2). Algorithm 2 then assigns time slots to the connectors (Steps 12–16 in Algorithm 2). Steps 9–17 for coloring the remaining dominators and connectors are repeated level by level until all dominators and connectors are scheduled.

2.6.2 Analysis of Algorithm

In this subsection, we analyze Cell Coloring and bound the latency of a schedule produced by the algorithm. We first prove that any two senders can transmit data without any interference and collision if they are $H$ hops apart.

**Lemma 2.6.1.** Let $H = \lceil \alpha \cdot \sqrt{2} + 2 \rceil$, where $\alpha \geq 1$ is the interference factor. Then any two nodes that are at least $H$ hops away from each other can send data simultaneously without any collision and interference.

**Proof.** Consider a sender node $i$ that is sending data to its receiver node $j$. Let $u$ be the farthest node that interferes with $j$, and let $d$ be the longest distance between $j$ and $u$. Then, $d \leq \alpha \cdot r$. We bound the number of cells between $u$ and $j$ as follows. Assume a straight line between $j$ and $u$, and relay nodes with the power level $P$ on the line. Let $h$ be the number of cells between $u$ and $j$. As $\frac{r}{\sqrt{2}} \cdot h \leq d$, we have $h \leq d \cdot \frac{\sqrt{2}}{r}$ which implies that $h \leq \alpha \cdot \sqrt{2}$. Therefore, there are at most $\lceil \alpha \cdot \sqrt{2} \rceil$ cells between $u$ and $j$, and any other sender must be at least $\lceil \alpha \cdot \sqrt{2} + 1 \rceil$ cells apart from node $j$ to not cause interference. This implies that any two senders must be at least $H = \lceil \alpha \cdot \sqrt{2} + 2 \rceil$ cells apart for successful concurrent transmissions. □

Next, we prove that the latency of a schedule found by the Cell Coloring algorithm is bounded by $O(\Delta + R)$.

**Lemma 2.6.2.** (Li et al., 2009) The number of connectors in a cell is at most 12.

**Theorem 2.6.1.** Cell Coloring algorithm’s latency is $O(\Delta + R)$, and it gives a nearly constant factor approximation ratio.
Proof. Consider Steps 4–8. Since there exists at most $\Delta$ dominatees in a cell, data from all dominatees can be aggregated to their dominators in $\Delta \cdot H^2$ time slots. Now, consider Steps 9–17. Due to the fact that there exists at most 1 dominator in each cell, in Steps 10–12, data from all dominators can be aggregated to their connectors in at most $H^2$ time slots. Next, observing that there exists at most 12 connectors in each cell, in Steps 12–16, data from all connectors can be aggregated to their dominators in at most $12 \cdot H^2$ time slots. Since Steps 9–17 are iterated $R$ times, it takes at most $13 \cdot H^2 \cdot R$ time slots. Thus, the latency of a schedule found by the Cell Coloring algorithm is $\Delta \cdot H^2 + 13 \cdot H^2 \cdot R = O(\Delta + R)$ which gives a nearly constant factor approximation ratio.

2.7 Simulation Results

In our simulation, networks are generated randomly in the Euclidean plane where the number of nodes is 100. The sink node $s$ is deployed at the top left corner and the others are randomly deployed on an area of size $400 \times 400$. Similar to (Chen et al., 2009), the initial graphs are built by assigning a uniform power level that is minimally required for the network to be connected. While the original SDA algorithm (Chen et al., 2009) takes into account only the conflict-free model with $\alpha = 1$, we consider the cases: $\alpha = 1, 2$ and 3. In order to compare the performances of the SDA, MDAB, and Cell Coloring algorithms, they are run on the same graphs. We generated 100 different networks, and averaged the lengths of the schedules produced by the algorithms over the networks generated. Figure 2.4 shows schedules computed by SDA, MDAB, and Cell Coloring where $\alpha = 1$ and nodes have a uniform power level.

Table 2.1 shows that MDAB performs much better than SDA (Chen et al., 2009) and Cell Coloring on average. It can also be observed that as $\alpha$ becomes larger (i.e., the interference range is wider), MDAB’s performance is increasingly better than SDA’s and Cell Coloring’s.
Figure 2.4. Schedules computed by SDA, MDAB, and *Cell Coloring* where $\alpha = 1$ and nodes have a uniform power level. (a) Unit Disk Graph (b) SDA schedule (c) MDAB schedule (d) *Cell Coloring* schedule. Black nodes represent dominators, and gray nodes represent connectors.

Since the *Cell Coloring* algorithm colors the nodes level by level on the data aggregation tree, it gives a larger latency than SDA and MDAB.

Table 2.1. Average length of schedules computed by SDA, MDAB, and *Cell Coloring* algorithms

<table>
<thead>
<tr>
<th>Interference Factor</th>
<th>SDA</th>
<th>MDAB</th>
<th><em>Cell Coloring</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23.22</td>
<td>21.13</td>
<td>68.30</td>
</tr>
<tr>
<td>2</td>
<td>41.04</td>
<td>33.73</td>
<td>76.76</td>
</tr>
<tr>
<td>3</td>
<td>58.27</td>
<td>47.85</td>
<td>86.76</td>
</tr>
</tbody>
</table>
2.8 Conclusion

In this chapter, we have studied the MDAS problem on the collision-free and interference-free networks. We have shown that there is no approximation algorithm having an approximation ratio better than $\Omega (\log n)$ for the MDAS problem in the metric model under the assumption that each node can have various transmission power levels (non-uniform power model). For the non-uniform power model, we introduced a heuristic called the MDAB algorithm that yields a significant improvement over SDA (Chen et al., 2009) although both have similar worst case upper bounds. For the uniform power model, we proposed a nearly constant factor approximation algorithm called Cell Coloring whose latency is bounded by $O(\Delta + R)$, where $\Delta$ is the maximum node degree of the network and $R$ is the network radius. To our knowledge, the complexity of MDAS in the geometric model remains an interesting open problem.
CHAPTER 3
MINIMUM LATENCY AGGREGATION SCHEDULING PROBLEM IN INTERFERENCE-AWARE 3-DIMENSIONAL WSNS

3.1 Abstract

In this chapter, we study the Minimum Latency Aggregation Scheduling (MLAS) problem in Wireless Sensor Networks (WSNs) adopting the two interference models: the graph model and the more realistic physical interference model known as Signal-to-Interference-Noise-Ratio (SINR). The main issue of the MLAS problem is to compute schedules with the minimum number of timeslots, that is, to compute the minimum latency schedules, such that data can be aggregated without any collision or interference. While existing works studied the problem in 2-dimensional (2D) WSNs only, we investigate the problem in the more general 3-dimensional (3D) WSNs, and introduce two approximation algorithms with $O(1)$-approximation ratios that yield schedules whose latency is bounded by $O(\Delta + R)$, where $\Delta$ is the maximum node degree and $R$ is the network radius. To the best known of our knowledge, our results are the first results of the MLAS problem in 3D WSNs.

3.2 Introduction

A Wireless Sensor Network (WSN) consists of a number of sensor nodes which monitor nearby environmental conditions and gather data periodically. The gathered data is forwarded to a destination called the sink node. This type of application is commonly known as data aggregation in the literature, and it is one of the most crucial applications of WSNs.

Although recent advances in WSNs have led to the development of sensor nodes, the small-sized sensors still have limited energy resources. Therefore, researchers have focused on
the issue of prolonging the network lifetime by reducing energy consumption which is caused by the unnecessary retransmission using sensors’ limited power. An interesting approach is to assign timeslots to sensor nodes to obtain a good schedule by which data can be aggregated without any collision or interference. Since the data collection occurs periodically, reducing the latency of the schedule, that is, constructing schedules with a minimum number of timeslots, has been a fundamental issue.

In the literature, the problem of constructing minimum latency data aggregation schedules, namely the Minimum Latency Aggregation Scheduling problem, has been widely investigated by several researchers in two interference models: the graph model and the physical interference model. In the collision-free graph model, (Chen et al., 2005) proved the NP-hardness of the problem, and showed the \((\Delta - 1)\)-approximation algorithm. Later, (Huang et al., 2007) introduced the first constant factor approximation algorithm whose latency is bounded by \(23R + \Delta - 18\) which was improved by (Xu et al., 2009) and (Xu et al., 2011) to \(16R + \Delta - 14\). In (Wan et al., 2009), Wan et al. proposed three approximation algorithms whose latency is bounded by \(15R + \Delta - 4, 2R + O(\log R) + \Delta\) and \(\left(1 + O\left(\frac{\log R}{\sqrt{R}}\right)\right)R + \Delta\), respectively.

While these works considered only collision, some researchers have studied the problem taking into consideration interference as well in the collision-interference-free graph model. (Wan et al., 2009) and (An et al., 2011) proposed constant factor approximation algorithms whose latency is bounded by \(O(\Delta + R)\), and (An et al., 2011) also proved an \(\Omega(\log n)\) approximation lower bound in the metric model. Recently, (Lam et al., 2013) introduced a constant factor approximation algorithm whose latency is bounded by \(O(R + \log n)\) assuming that multiple power levels are present, and the maximum power level is bounded, while (Wan et al., 2009; An et al., 2011) assumed the uniform power model.

Recently, several researchers have started investigating data aggregation scheduling in the more realistic physical interference model known as Signal-to-Interference-Noise-Ratio
(SINR). Unlike the graph model, the SINR model captures real world phenomena adequately by considering the cumulative interference caused by all the other concurrently transmitting nodes. The first investigation of the Minimum Latency Aggregation Scheduling problem in the SINR model was done by Li et al. (Li et al., 2009). (Li et al., 2009) introduced a constant factor approximation algorithm whose latency is bounded by $O(\Delta + R)$ under the uniform power model. (An et al., 2012b) extended it to the dual power model, and introduced two constant factor approximation algorithms whose latency is bounded by $O(\Delta + R)$. (An et al., 2012b) also showed not only an $\Omega(\log n)$ approximation lower bound in the metric SINR model, but also its NP-hardness in the geometric SINR model. (Li et al., 2010) proposed an algorithm that yields $O(\log^3 n)$-latency which was improved by (Halldórsson and Mitra, 2012) to $O(\log n)$. Recently, (Lam et al., 2013) introduced a constant factor approximation algorithm whose latency is bounded by $O(R + \log n)$. Note that (Li et al., 2010) and (Halldórsson and Mitra, 2012) assumed the unlimited power model and (Lam et al., 2013) assumed that multiple power levels are present, and the maximum power level is bounded. Following shows a summary of the works concerning the data aggregation problem in 2-dimensional WSNs.

- **Collision-Free Model**
  - Uniform Power Model: (Chen et al., 2005; Huang et al., 2007; Xu et al., 2009, 2011; Wan et al., 2009; Yu et al., 2009)
  - Non-uniform Power Model: (Lam et al., 2013)

- **Collision-Interference-Free Model**
  - Uniform Power Model: (Wan et al., 2009; An et al., 2011)
  - Non-uniform Power Model: (Lam et al., 2013)
• SINR Model
  – Uniform Power Model: (Li et al., 2009, 2010; Wan et al., 2009)
  – Non-uniform Power Model: (Lam et al., 2013; An et al., 2012b; Halldórsson and Mitra, 2012)

While these studies have been concerned with data aggregation, some other researchers have focused on related applications such as broadcast and gossiping. The broadcast problem is to distribute a unique message from a source (sink) node to all the other nodes, whereas the gossiping problem, which is also known as all-to-all broadcast, is to distribute the message of each node to all the other nodes in the network. For the problem of broadcast, NP-hardness was proved by (Gandhi et al., 2003) which holds for the collision-free graph model, but not for the other models. For the gossiping problem, (An et al., 2010b) proved its NP-hardness which holds for the SINR model under assumption that a node can combine messages and there is no limit on the length of the combined message. Following shows a summary of works on this topic.

• Collision-Free Model
  – Broadcast: (Gandhi et al., 2003; Huang et al., 2007; Gandhi et al., 2009; Tiwari et al., 2009)
  – Gossiping: (Bar-yehuda et al., 1993; Christersson et al., 2002; Manne and Xin, 2006; Cicalese et al., 2006; Huang et al., 2008; Gandhi et al., 2009, 2003; An et al., 2010b; Krzywdzinski, 2010)

• Collision-Interference-Free Model
  – Broadcast: (An et al., 2010b; Tiwari et al., 2009; Chen et al., 2007; Huang et al., 2008; Calinescu and Tongngam, 2011)
Among these works that studied the problems in 2-dimensional WSNs, (Tiwari et al., 2009) was the only one that has investigated the broadcast problem in both 2- and 3-dimensional WSNs. In 3-dimensional (3D) WSNs adopting the graph model, (Tiwari et al., 2009) introduced a constant factor approximation algorithm where the 3D space is partitioned into several truncated octahedrons.

In this chapter, we continue the study of the Minimum Latency Aggregation Scheduling problem in 3D WSNs adopting both the graph model and geometric SINR model. While existing works studied the problem in 2-dimensional (2D) WSNs only, we investigate the problem in the more general 3-dimensional (3D) WSNs, and introduce two constant factor approximation algorithms that yield schedules whose latency is bounded by $O(\Delta + R)$. Our approximation algorithms for the problem are the first results, to the best of our knowledge, for 3D WSNs adopting both interference models.

This chapter is organized as follows. Section 3.3 describes our network models and defines the Minimum Latency Aggregation Scheduling (MLAS) problem. In Section 3.4, we show our 3D-space-filling and labeling techniques. Section 3.5 introduces two constant factor approximation algorithms for the MLAS problem, and we analyze them in Section 3.6. Finally, Section 3.7 contains some concluding remarks.
3.3 Preliminaries

3.3.1 3D Network Models

In this chapter, a wireless sensor network (WSN) consists of a set $V$ of sensor nodes deployed in a 3-dimensional (3D) space, and each node $u \in V$ is assigned a transmission power level $p(u)$. Accordingly, a directed edge $(u, v)$ exists from node $u$ to node $v$, if $v$ resides in the transmission ball with radius $p(u)$ of $u$, i.e., $d(u, v) \leq p(u)$, where $d(u, v)$ denotes the Euclidian distance between $u$ and $v$.

Graph Model

In the graph model, let $B^u_{p(u)} = \{v | v \in V, d(u, v) \leq p(u)\}$ denote the set of all nodes that can be reached by $u$ with the power level $p(u)$. If two nodes $u$ and $v$ reside in the transmission ball of each other, i.e., $u \in B^v_{p(v)}$ and $v \in B^u_{p(u)}$, then $u$ and $v$ can communicate. However, we also need to consider the collision or interference. Given a power level $p(u)$ of $u$, the interference ball of $u$ is defined as a ball with radius $\rho \cdot p(u)$, where $\rho \geq 1$ is the interference factor. (See Figure 3.1.) Given $\rho \geq 1$, let $I^u_{p(u)} = \{v | v \in V, d(u, v) \leq \rho \cdot p(u)\}$ denote the set of all nodes in the interference ball of $u$. Then, collision (or conflict) is said to occur at a receiver node $w$ if there exist other concurrently sending nodes $u$ and $v$ such that $w \in B^u_{p(u)} \cap I^v_{p(v)}$, where $\rho = 1$. On the other hand, interference is said to occur at $w$ if there exist other concurrently sending nodes $u$ and $v$ such that $w \in B^u_{p(u)} \cap I^v_{p(v)}$, where $\rho > 1$. In the literature, the graph model concerning only collision (i.e., when $\rho = 1$) is called the collision-free graph model, whereas the graph model concerning both collision and interference (i.e., when $\rho \geq 1$) is called the collision-interference-free graph model.

In the graph model, the communication graph can be modeled as a bidirectional ball graph $G(V, E)$, where $E = \{(u, v) | u, v \in V, d(u, v) \leq p(u) \text{ and } d(v, u) \leq p(v)\}$. 
In the physical interference model (SINR) (Gupta and Kumar, 2000), if a node $u$ transmits with its power level $p(u)$, then the received power at a receiver $v$ is $p(u) \cdot d(u, v)^{-\alpha}$, where $\alpha \in [2, 6]$ is the path loss exponent. In order that the receiver $v$ can receive the data transmitted by the sender $u$, the ratio of the received power at $v$ to the interference caused by all the other concurrently transmitting nodes and background noise must be beyond an SINR threshold $\beta \geq 1$. Formally, node $v$ can successfully receive data via the communication edge $(u, v)$ only if

$$\text{SINR}_{(u,v)} = \frac{\frac{p(u)}{d(u,v)^{\alpha}}}{N + \sum_{w \in X - \{u,v\}} \frac{p(w)}{d(w,v)^{\alpha}}} \geq \beta$$

(3.1)

where $N > 0$ is the background noise, and $X$ is the set of other concurrently transmitting nodes. Observing that $u$ can send its data to the nodes within the distance $(\frac{p(u)}{N^\beta})^{\frac{1}{\alpha}}$, the network can be modeled as a directed ball graph $G(V, E)$, where $E = \{(u \rightarrow v) | u, v \in V, d(u, v) \leq (\frac{p(u)}{N^\beta})^{\frac{1}{\alpha}} \}$.

Note that in the literature, a communication graph is called a ball graph (BG) if all nodes are assigned various power levels, and if all nodes are assigned the same power level, it is called a unit ball graph (UBG).
3.3.2 Problem Definition

The Minimum Latency Aggregation Scheduling (MLAS) problem is defined as follows. Considering a set of nodes in a 3D space, we assign these nodes a number of timeslots such that nodes scheduled to send data at the same timeslot can send data to its receivers simultaneously without any collision or interference. A schedule is defined as a sequence of such timeslots. Formally, at each timeslot $t$, we have an assignment vector $\pi_t = ((s_{t_1}, p(s_{t_1})), \cdots, (s_{t_m}, p(s_{t_m})))$ in which $s_{t_i}$ is assigned to send data with its power level $p(s_{t_i})$, $1 \leq i \leq m$, and

- (Graph Model) neither collision nor interference occurs at any receiver $r$, or
- (SINR Model) the SINR threshold inequality is satisfied for all receivers $r$,

where $(s_{t_i}, r)$ is an edge in the communication graph $G(V, E)$.

A schedule is a sequence of assignment vectors $\Pi = (\pi_1, \pi_2, \cdots, \pi_M)$, where $M$ is the length of the schedule which is also called its latency. A schedule $\Pi$ is successful if all data of each node $v \in V$ is aggregated to a sink node $s \in V$.

**Input.** A set $V$ of nodes in a 3-dimensional (3D) space, a sink node $s \in V$.

**Output.** A successful minimum latency schedule.

3.3.3 NP-Hardness of the MLAS Problem in 3D WSNs

Note that (Chen et al., 2005), (An et al., 2011) and (An et al., 2012b) showed the NP-hardness of the MLAS problem in the 2-dimensional (2D) collision-free graph model, collision-interference-free graph model, and SINR model, respectively. Observing that a 2D WSN is a special case of 3D WSNs, the MLAS problem in 3D WSNs is also NP-hard.

3.4 3-Dimensional Space Filling

In this section, we introduce our 3-dimensional (3D) space-filling technique. It has been known that there are only five space-filling convex polyhedra: triangle prism, cube, hexagonal
prism, truncated octahedron, and gyrobifastigium. In this chapter, we fill the 3D space with the cube, which is the only platonic solid, and the hexagonal prism.

3.4.1 Space Filling with Cubes

We partition the 3D space containing the network into cubes whose side length \( a \) is \( \frac{r}{\sqrt{3}} \), and space diagonal is \( r \). (See Figure 3.2(a).) Each cube is labeled with the label \( CL(x, y, z) \) if \((x, y, z)\) is its vertex with the smallest \( x-\), \( y-\) and \( z-\) coordinates.

![Cube and Hexagonal Prism Diagram](image.png)

Figure 3.2. Space-filling convex polyhedra

3.4.2 Space Filling with Hexagonal Prisms

The 3D space containing the network is tessellated with hexagonal prisms whose side length \( a \) is \( \frac{r}{\sqrt{3}} \), and space diagonal is \( r \). (See Figure 3.2(b).) The hexagonal prisms are labeled using \( (9k^3 - 3k^2) \)-labeling, where \( k \) is a positive integer. Figure 3.3(a) shows a \( (3 \times 2) \)-labeling when \( k = 1 \). The \( (3 \times 2) \)-labeling consists of 2 layers each of which consists of 3 hexagonal prisms. Figure 3.3(b) shows an example of filling a 3D network space using \( (3 \times 2) \)-labeling. In this \( (3 \times 2) \)-labeling, we can observe that the distance between two hexagonal prisms with the same label is \( \frac{r}{\sqrt{3}} \). Here, the distance between any two hexagonal prisms, denoted by \( hex_i \) and \( hex_j \), respectively, is defined as the distance between two closest vertices \( p \) in \( hex_i \), and \( p' \) in \( hex_j \). Next, Figure 3.4(a) shows \( (12 \times 5) \)-labeling when \( k = 2 \), and it consists of 5 layers each of which consists of 12 hexagonal prisms. Figure 3.4(b) shows an example of hexagonal-prism tessellation on a 3D network using \( (12 \times 5) \)-labeling. In this \( (12 \times 5) \)-labeling, the
distance between two hexagonal prisms with the same label is $4 \cdot \frac{r}{\sqrt{3}}$. When $k = 3$, we have $(27 \times 8)$-labeling that consists of 8 layers each of which consists of 27 hexagonal prisms. (See Figure 3.5.) When a 3D network is filled with the hexagonal prisms with $(27 \times 8)$-labeling, the distance between the hexagonal prisms is $7 \cdot \frac{r}{\sqrt{3}}$.

![Figure 3.3. (a) 6-labeling ($k = 1$) (b) Space-filling with 6-labeling](image)

![Figure 3.4. (a) 60-labeling ($k = 2$) (b) Space-filling with 60-labeling](image)

In general, we have a $K = (3k^2 \times (3k - 1))$-labeling, and the distance between two hexagonal prisms with the same label is $(3k - 2) \cdot \frac{r}{\sqrt{3}}$.

### 3.5 Constant Factor Approximation Algorithms

#### Cube-based Aggregation Scheduling Algorithm

The first algorithm called the *Cube-based Aggregation Scheduling (CBAS)* algorithm starts by partitioning the network into cubes as described in Section 3.4.1, and a number of iterations
are performed to find a schedule based on $T$. Assigning timeslots is also based on the constant value $C$ that guarantees that any two senders can send data to their receivers at the same time if they are $C$ cubes apart from each other. Now, the constant $C$ is set as follows in the two different interference models:

- **Graph model:** $C = \lceil \rho \cdot \sqrt{3} + 3 \rceil$

- **SINR model:** $C = \lceil \left( \frac{P \cdot 4\pi^2}{N(\beta - \alpha - 1)(\alpha - 3)} \right)^{\frac{1}{\alpha - 3}} \cdot \frac{\sqrt{3}}{\delta} \left( \frac{N \cdot \beta}{P} \right)^{\frac{1}{\alpha}} + 2 \rceil$

Then, the CBAS algorithm can be used not only for the graph model, but also for SINR model where the constant $C$ is defined accordingly.

Algorithm 4 shows the details of the CBAS algorithm. It schedules the nodes in $T$ starting with the dominatees so that they can send data without any collision or interference to their dominators (Steps 4 – 5 in Algorithm 4). While scheduling dominatees, the CBAS algorithm (Algorithm 4) uses Algorithm 5 as a subroutine to assign the same timeslots to the dominatees if they are $C$ cubes away from each other. Note that while scheduling, we pick only one dominatee in each cube arbitrarily (Step 2 in Algorithm 5), and repeat the scheduling procedure (Algorithm 5) until all dominatees are scheduled.
Algorithm 4 Cube-based Aggregation Scheduling (CBAS)

**Input**: A set $V$ of nodes in a 3D space

**Output**: Length of Schedule

1: Fill the space with cubes whose edge length is $\frac{r}{\sqrt{3}}$.
2: Construct a data aggregation tree $T$ using the algorithm in (Huang et al., 2007) rooted at a center node $c$.
3: Set the first timeslot $t \leftarrow 1$
4: $S_d \leftarrow$ the set of dominatees
5: $t \leftarrow TA-C(S_d, t)$
6: for $i = R$ to 1 do
7: $S_i \leftarrow$ the set of dominators at level $i$ in $T$
8: if $S_i \neq \emptyset$ then $t \leftarrow TA-C(S_i, t)$
9: $S_c \leftarrow$ the set of connectors at level $i$ in $T$
10: if $S_c \neq \emptyset$ then $t \leftarrow TA-C(S_c, t)$
11: end for
12: Send the aggregated data from the center node $c$ to the sink node $s$ via a shortest path $f$.
13: return $(t - 1) + \text{length of } f$

Algorithm 5 TimeSlot Assignment (TA-C)

**Input**: A set $S$ of sender nodes and a starting timeslot $t$

**Output**: Timeslot $t$

1: while $S \neq \emptyset$ do
2: Pick one node $v_s \in S$ in each cube. Let $S' \subseteq S$ be the set of such nodes.
3: for $t_1 = 0,...,C$, $t_2 = 0,...,C$, $t_3 = 0,...,C$ do
4: $S'' \leftarrow \emptyset$, $S'' \leftarrow \{v_s|v_s \in S'$ with $CL(x,y,z)$ such that $t_1 = x \mod (C + 1)$, $t_2 = y \mod (C + 1)$ and $t_3 = z \mod (C + 1)\}$
5: if $S'' \neq \emptyset$ then
6: for each $v_s \in S''$ do
7: $TS(v_s) \leftarrow t$
8: end for
9: $t \leftarrow t + 1$, $S \leftarrow S - S''$, $S' \leftarrow S' - S''$
10: end if
11: end for
12: end while
13: return $t$

After all dominatees are scheduled, several iterations are performed (Steps 6 – 11 in Algorithm 4) to schedule the remaining dominators and connectors level by level until all of them are scheduled, as follows. At each iteration for level $i$ of $T$, if there exist dominators
that have just received data from their lower level dominatees or connectors at level \(i+1\), then the dominators are scheduled to send the aggregated data to their upper level connectors at level \(i-1\). Otherwise, if there exist connectors that have just received data from their lower level dominators at level \(i+1\), then the connectors are scheduled to send the aggregated data to their upper level dominators at level \(i-1\). While scheduling, the CBAS algorithm (Algorithm 4) also uses Algorithm 5 as a subroutine to assign the same timeslots to them if they are \(C\) cubes away from each other. Notice that there exists only one dominator in a cube, whereas there may exist several connectors in a cube. Therefore while scheduling connectors, at each level \(i\), we pick only one connector in each cube arbitrarily (Step 2 in Algorithm 5), and repeat the scheduling procedure (Algorithm 5) until all connectors at level \(i\) are scheduled. Once all data is aggregated to the center node \(c\), \(c\) sends the aggregated data to the sink node \(s\) via a shortest path (Step 12 in Algorithm 4).

**Hexagonal-Prim-based Aggregation Scheduling**

The second algorithm called Hexagonal-Prim-based Aggregation Scheduling (HPBAS) starts by partitioning a network into hexagonal prisms which are labeled using \(K = (9k^3 - 3k^2)\)-labeling as described in Section 3.4.2, and a number of iterations are performed to find a schedule based on \(T\) obtained as the CBAS algorithm. Assigning timeslots is also based on the constant value \(K\) that guarantees that any two senders can send data to their receivers at the same time if they are located in the hexagonal prisms with the same label according to the \(K\)-labeling. Let us set the constant \(K\) as follows in the two different interference models:

- **Graph model:** \(K = 9k^3 - 3k^2\), where \(k = \left\lceil \frac{\sqrt{5(\rho+2)^2} + 2}{3} \right\rceil\)

- **SINR model:** \(K = 9k^3 - 3k^2\), where \(k = \left\lceil \frac{\sqrt{\frac{\gamma}{\beta}} \cdot \left( \frac{N\delta}{P} \right)^\frac{1}{\alpha} \cdot \left( \frac{P_{4\gamma^2}}{N(\delta-1)(\alpha-3)} \right)^\frac{1}{\alpha-3} + \sqrt{5} + 2 \right\rceil\)
The HPBAS algorithm can be used not only for the graph model, but also for the SINR model where the constant $K$ is defined accordingly.

Algorithm 6 shows the details of the HPBAS algorithm. Similar to CBAS, the HPBAS algorithm first schedules the dominatees (Steps 4 – 5 in Algorithm 6), and then several iterations are performed to schedule the remaining dominators and connectors (Steps 6 – 11 in Algorithm 6). While scheduling, HPBAS uses Algorithm 7 as a subroutine to assign the same timeslots to node which are located in the hexagonal prisms with the same label. As the final step, the aggregated data at the center node $c$ is sent to the sink node $s$ via a shortest path (Step 12 in Algorithm 6).

**Algorithm 6** Hexagonal-Prism-based Scheduling

**Input:** A set $V$ of nodes in a 3D space

**Output:** Length of Schedule

1: Fill the space with hexagonal prisms whose side length is $\frac{r}{\sqrt{5}}$, and label the prisms using $K$-labeling.
2: Construct an aggregation tree $T$ using an algorithm in (Huang et al., 2007) rooted at a center node $c$.
3: Set the first timeslot $t \leftarrow 1$
4: $S_d \leftarrow$ the set of dominatees of $V$.
5: $t \leftarrow TA-X(S_d, t)$
6: for $i = R$ to 1 do
7:    $S_d^i \leftarrow$ the set of dominators at level $i$ in $T$
8:    if $S_d^i \neq \emptyset$ then $t \leftarrow TA-X(S_d^i, t)$
9:    $S_c^i \leftarrow$ the set of connectors at level $i$ in $T$
10:   if $S_c^i \neq \emptyset$ then $t \leftarrow TA-X(S_c^i, t)$
11: end for
12: Send the aggregated data from the center node $c$ to the sink node $s$ via a shortest path $f$.
13: return $(t - 1) +$ length of $f$

3.6 Analysis

In this section, we analyze the Cube-Based Aggregation Scheduling (CBAS) and Hexagonal-Prim-Based Aggregation Scheduling (HPBAS) algorithms (Algorithms 4 and 6).
Algorithm 7 TimeSlot Assignment (TA-X)

\textbf{Input:} A set \( S \) of sender nodes and a starting timeslot \( t \)

\textbf{Output:} Timeslot \( t \)

1: \textbf{while} \( S \neq \emptyset \) \textbf{do}
2: \hspace{1em} \text{Pick one node } v_s \in S \text{ in each cube. Let } S' \subseteq S \text{ be the set of such nodes.}
3: \hspace{1em} \textbf{for} \( i = 1 \) to \( K \) \textbf{do}
4: \hspace{2em} \( S'' \leftarrow \emptyset \), \( S'' \leftarrow \{v_s|v_s \in S' \text{ with } HL(v_s) = i\} \)
5: \hspace{2em} \textbf{if} \( S'' \neq \emptyset \) \textbf{then}
6: \hspace{3em} \textbf{for} each \( v_s \in S'' \) \textbf{do}
7: \hspace{4em} \( TS(v_s) \leftarrow t \)
8: \hspace{3em} \textbf{end for}
9: \hspace{2em} \( t \leftarrow t + 1 \), \( S \leftarrow S - S'' \), \( S' \leftarrow S' - S'' \)
10: \hspace{1em} \textbf{end if}
11: \hspace{1em} \textbf{end for}
12: \textbf{end while}
13: \textbf{return} \( t \)

3.6.1 Analysis of CBAS Algorithm

First, we analyze the CBAS algorithm, and bound the latency of the schedule produced by it. We first prove that any two senders can send data at the same time without any collision and interference if they are \( C \) cubes apart.

**Lemma 3.6.1** (Graph Model). Let \( C = \lceil \rho \cdot \sqrt{3} + 3 \rceil \), where \( \rho \geq 1 \) is the interference factor. Then any two sender nodes that are at least \( C \) cubes apart from each other can concurrently send data without any collision and interference.

**Proof.** Consider a sender node \( v_i \) trying to send data to its receiver \( v_j \), and the farthest sender node \( v_k \) that interferes with \( v_j \). Then, \( d(v_k, v_j) \leq \rho \cdot r \).

Next, letting \( z \) denote the number of cubes between \( v_j \) and \( v_k \), we bound \( z \) as follows. Consider the straight line between \( v_j \) and \( v_k \). Then, as \( \frac{r}{\sqrt{3}} \cdot z \leq d(v_k, v_j) \), we have \( z \leq d(v_k, v_j) \cdot \frac{\sqrt{3}}{r} \) which implies that \( z \leq \rho \cdot \sqrt{3} \). Therefore, there are at most \( \lceil \rho \cdot \sqrt{3} \rceil \) cubes between \( v_j \) and \( v_k \), and any other sender must be at least \( \lceil \rho \cdot \sqrt{3} + 1 \rceil \) cubes apart from the node \( v_j \) not to cause interference. Now, observing that the number of cubes between \( v_i \) and \( v_j \) is at most 1, and considering the cube in which \( v_j \) is located, we can set \( C = \lceil \rho \cdot \sqrt{3} + 3 \rceil \). \( \square \)
Lemma 3.6.2 (SINR Model). For SINR threshold $\beta \geq 1$, path loss exponent $\alpha > 3$, background noise $N > 0$, and some constant $\delta \in (0, 1)$, let

$$C = \left[ \left( \frac{P \cdot 4\pi^2}{N (\delta - \alpha - 1)(\alpha - 3)} \right)^{\frac{1}{\alpha - 3}} \cdot \sqrt{3} \cdot (\delta (\frac{P}{N\beta})^{\frac{1}{\alpha}})^{-1} + 2 \right]$$

Then any two sender nodes that are at least $C$ cubes away from each other can send data at the same time.

Proof. Consider a sender node $v_i$ trying to send data to its farthest possible receiver $v_j$, i.e., $d(v_i, v_j) = \delta (\frac{P}{N\beta})^{\frac{1}{\alpha}}$. In order that the receiver $v_j$ receives data from the sender $v_i$ without interference, for all other concurrently sending nodes, the following must be satisfied:

$$P \left( \delta (\frac{P}{N\beta})^{\frac{1}{\alpha}} \right)^{-\alpha} \geq \beta$$

which implies

$$\frac{P \sum_{v_i' \notin \{v_i, v_j\}} d(v_i', v_j)^{-\alpha}}{N(\delta^{-\alpha} - 1)} \leq \frac{P \int_0^{2\pi} \int_0^{2\pi} \int_x^{\infty} \frac{y^2}{y^{\alpha - 2}} dy d\theta d\varphi}{N(\delta^{-\alpha} - 1)}$$

$$= \frac{P \cdot 4\pi^2 \int_x^{\infty} y^{2-\alpha} dy}{N(\delta^{-\alpha} - 1)}$$

$$= \frac{P \cdot 4\pi^2 \cdot x^{3-\alpha}}{N(\delta^{-\alpha} - 1)(\alpha - 3)} \leq 1 \quad (3.4)$$

where $x$ is the shortest distance between $v_j$ and one of the other concurrently sending nodes.

From inequality (3.4), we get $x \geq \left( \frac{P \cdot 4\pi^2}{N (\delta - \alpha - 1)(\alpha - 3)} \right)^{\frac{1}{\alpha - 3}}$. Thus $X := \left( \frac{P \cdot 4\pi^2}{N (\delta - \alpha - 1)(\alpha - 3)} \right)^{\frac{1}{\alpha - 3}}$ is a lower bound for $x$.

Next, let us bound the number of cubes between $v_j$ and a closest concurrently sending node to $v_j$, say $v_j'$. Let $z$ be the number of cubes between $v_j'$ and $v_j$. We bound $z$ as follows. Consider the straight line between $v_j'$ and $v_j$, and the cubes lying on the line. We have $X \leq z \cdot \frac{1}{\sqrt{3}} \cdot \delta (\frac{P}{N\beta})^{\frac{1}{\alpha}}$ which implies $X \cdot \sqrt{3} \cdot (\delta (\frac{P}{N\beta})^{\frac{1}{\alpha}})^{-1} \leq z$. Therefore, $v_j$ and $v_j'$ should be at least $Z := \lceil X \cdot \sqrt{3} \cdot (\delta (\frac{P}{N\beta})^{\frac{1}{\alpha}})^{-1} \rceil$ cubes apart. Now, observing that the number of cubes between $v_i$ and $v_j$ is at most 1, and considering the cube in which $v_j$ is located, we can set

$$C = \left[ \left( \frac{P \cdot 4\pi^2}{N (\delta - \alpha - 1)(\alpha - 3)} \right)^{\frac{1}{\alpha - 3}} \cdot \sqrt{3} \cdot (\delta (\frac{P}{N\beta})^{\frac{1}{\alpha}})^{-1} + 2 \right].$$
Lemma 3.6.3. The number of connectors in a cube is at most $7^3 - 1$.

Proof. Consider a dominator $v$ in a cube, denoted by $\text{cube}(v)$, and its connectors. As the connectors connect dominators which are 2-hops away from $v$ in the CDS, the number of connectors in one cube cannot exceed the number of dominators that are at most 2 hops away from $v$. Thus, it is sufficient to bound the number of such dominators.

Consider a ball, whose radius is $2r$, that is totally contained within a cube, denoted by $\text{cube}'$, with the side length $4r$ (See Figure 3.6). Then, the number of 2-hop away dominators cannot exceed the number of cubes whose side length is $\frac{r}{\sqrt{3}}$ within $\text{cube}'$. As there exist at most $\lceil 4\sqrt{3} \rceil^3$ cubes whose side length is $\frac{r}{\sqrt{3}}$ within $\text{cube}'$, there are at most $\lceil 4\sqrt{3} \rceil^3$ dominators in $\text{cube}'$. This implies that there exist at most $7^3 - 1$ connectors for $v$, and therefore at most $7^3 - 1$ connectors in a cube.

Lemma 3.6.4 (Lower Bound, Graph Model). (Xu et al., 2011) In order to produce a successful schedule, any data aggregation scheduling algorithm requires

- $\geq \max\{\Delta, \log R\}$ timeslots, for $\rho = 1$,
- $\geq \max\{\frac{\Delta}{\phi}, R\}$ timeslots, where $\phi = \frac{2\pi}{\lceil \arcsin \frac{\rho}{2r} \rceil}$, for $1 < \rho < 3$, and
- $\geq \max\{\Delta, R\}$ timeslots, for $\rho \geq 3$. 

Figure 3.6. The inscribed ball that touches each face of the cube.
Lemma 3.6.5 (SINR Model). (Li et al., 2009) For any node, at most \( \omega = r^\alpha - 1 \) neighboring nodes can send data at the same time, where \( r = \delta \left( \frac{P}{N^\beta} \right)^{\frac{1}{\delta}} \).

Corollary 3.6.1 (Lower Bound, SINR Model). In order to produce a successful schedule, any data aggregation scheduling algorithm requires \( \geq \max \{ \Delta \omega, R \} \) timeslots.

Theorem 3.6.1. The CBAS algorithm produces a successful schedule whose latency is bounded by \( O(\Delta + R) \), and it is therefore a constant-factor approximation algorithm.

Proof. First, consider the Steps 4 – 5 in Algorithm 4 that schedules dominatees. In a cube, there exist at most \( \Delta \) dominatees sharing one dominator \( v \). Thus, gathering data from all the dominatees to the corresponding dominators takes at most \( \Delta \cdot (C + 1)^2 \) timeslots.

Next, consider Steps 6 – 11 in Algorithm 4 that schedule the remaining dominators and connectors. For each iteration that schedules nodes at level \( i \), we consider the following cases:

1. Assigning timeslots to dominators at level \( i \) to send data to their connectors at level \( i - 1 \): In this case, there exists at most 1 dominator in a cube, and therefore gathering data from all the dominators at level \( i \) to the corresponding connectors at level \( i - 1 \) takes at most \( (C + 1)^2 \) timeslots.

2. Assigning timeslots to connectors at level \( i \) to send data to their dominators at level \( i - 1 \): In this case, there exist at most \( 7^3 - 1 \) connectors in a cube (Lemma 3.6.3), and therefore gathering data from all the connectors at level \( i \) to the corresponding dominators at level \( i - 1 \) takes at most \( (7^3 - 1)(C + 1)^2 \) timeslots.

As Steps 6 – 11 repeat at most \( R \) times, it takes at most \( 7^3(C + 1)^2 \cdot R \) timeslots.

Finally, as Step 12 in Algorithm 4 takes at most \( R \) timeslots, the latency of CBAS is bounded by \( (C + 1)^2 \cdot \Delta + 7^3(C + 1)^2 \cdot R = O(\Delta + R) \). Thus, it is a constant factor approximation in both the graph model and the SINR model by Lemma 3.6.4 and Corollary 3.6.1. \( \square \)
3.6.2 Analysis of HPBAS Algorithm

We now analyze the HPBAS algorithm, and bound the latency of the schedule produced by it. We first prove that any two senders can send data at the same time without any collision and interference if they are located in the hexagonal prisms with the same label according to the $K$-labeling.

**Lemma 3.6.6 (Graph Model).** Let $k = \lceil \frac{\sqrt{5(\rho+2)+2}}{3} \rceil$, where $\rho \geq 1$ is the interference factor. Then any two sender nodes which are located in the hexagonal-prisms with the same label according to the $K$-labeling can send data at the same time.

*Proof.* Consider a sender node $v_i$ sending its data to its farthest possible receiver $v_j$, i.e., $d(v_i, v_j) = r$. The proof of Lemma 3.6.1 showed that the farthest distance between $v_j$ and the other sender node that interferes with $v_j$, say $v_k$, is at most $\rho \cdot r$. As the maximum distance between any two nodes is $r$, if any two senders are $\rho \cdot r + 2r$ distance apart from each other, then they can send data at the same time.

Next, the distance between two hexagonal prisms with the same label is $\frac{r}{\sqrt{5}}(3k - 2)$ in $K = (9k^3 - k^2)$-labeling. Letting $\frac{r}{\sqrt{5}}(3k - 2) \geq \rho \cdot r + 2r$, we can set $k = \lceil \frac{\sqrt{5(\rho+2)+2}}{3} \rceil$. \hfill \Box

**Lemma 3.6.7 (SINR Model).** For SINR threshold $\beta \geq 1$, path loss exponent $\alpha > 3$, background noise $N > 0$, and some constant $\delta \in (0, 1)$, let

$$k = \lceil \frac{1}{3} \left\{ \frac{\sqrt{5}}{\delta} \cdot \left( \frac{N\beta}{P} \right)^\frac{1}{\alpha} \cdot \left( \frac{P \cdot 4\pi^2}{N(\delta^{-1}-1)(\alpha-3)} \right)^\frac{1}{\alpha-3} + \sqrt{5} + 2 \right\} \rceil$$

Then any two sender nodes which are located in the hexagonal-prisms with the same label according to the $K$-labeling can send data at the same time.

*Proof.* Consider a sender node $v_i$ sending its data to its farthest possible receiver $v_j$, i.e., $d(v_i, v_j) = r = \delta \left( \frac{P}{N\beta} \right)^\frac{1}{\alpha}$. The proof of Lemma 3.6.2 showed that the shortest distance between $v_j$ and the other concurrently sending node is at least $\left( \frac{P \cdot 4\pi^2}{N(\delta^{-1}-1)(\alpha-3)} \right)^\frac{1}{\alpha-3}$. Since the maximum
distance is \( \delta \left( \frac{P}{N^\beta} \right)^{\frac{1}{\alpha}} \), any two sender nodes can send data at the same time if they are at least of distance

\[
D := \left( \frac{P \cdot 4\pi^2}{N \left( d^2 - 1 \right) (\alpha - 3)} \right)^{\frac{1}{\alpha - 3}} + \delta \left( \frac{P}{N^\beta} \right)^{\frac{1}{\alpha}} \text{ apart from each other.}
\]

Next, the shortest distance between two hexagons with the same label is \( \frac{r}{\sqrt{5}} (3k - 2) \) in \( K = (9k^3 - k^2) \)-labeling. Letting \( \frac{r}{\sqrt{5}} (3k - 2) \geq D \), we can set \( k = \left\lceil \frac{1}{3} \left( \frac{\sqrt{5} \cdot N^\beta}{P} \right)^{\frac{1}{\alpha}} \cdot \left( N^\beta P \right)^{\frac{1}{\alpha - 3}} + \sqrt{5} + 2 \right\rceil \).

**Lemma 3.6.8.** The number of connectors in a hexagonal prism is at most \( 7^{3} - 1 \).

**Proof.** The proof is similar to the proof of Lemma 3.6.3, and is omitted. \( \square \)

**Theorem 3.6.2.** The HPBAS algorithm produces a successful schedule whose latency is bounded by \( O(\Delta + R) \), and it is therefore a constant-factor approximation algorithm.

**Proof.** Using an argument similar to the one the proof of Theorem 3.6.1, the latency of the schedules produced by the HPBCA algorithm (Algorithm 6) can be bounded by \( K \cdot \Delta + 7^3 \cdot K \cdot R = O(\Delta + R) \). Thus, it is a constant factor approximation in both the graph model and the SINR model by Lemma 3.6.4 and Corollary 3.6.1. \( \square \)

### 3.7 Conclusion

In this chapter, we studied the Minimum Latency Aggregation Scheduling (MLAS) problem adopting the two interference models: the graph model and the more realistic physical interference model known as Signal-to-Interference-Noise-Ratio (SINR). While existing works studied the problem in 2-dimensional (2D) WSNs only, we investigated the problem in the more general 3-dimensional (3D) WSNs, and introduced two approximation algorithms with \( O(1) \)-approximation ratios that yield schedules whose latency is bounded by \( O(\Delta + R) \), where \( \Delta \) is the maximum node degree and \( R \) is the network radius. As to future work, we plan to study the other related problems such as broadcast as well as gossiping in 3D WSNs adopting both interference models.
CHAPTER 4
MINIMUM LATENCY GOSSIPING IN WIRELESS SENSOR NETWORKS

4.1 Abstract

Gossiping is one of the most crucial applications in Wireless Sensor Networks (WSNs) which has been the focus of many researchers. A main issue of gossiping is how to assign timeslots to nodes for interference-free data transmission. There are three models concerning gossiping in WSNs: unit-sized, bounded-sized, or unbounded-sized messages. For these models, the problem of constructing minimum latency gossiping schedules has been widely studied in the literature although most of the existing studies are based on the graph model.

In this chapter, we study the Minimum Latency Gossiping (MLG) problem with unbounded-sized messages in the graph model as well as the more realistic physical interference model known as Signal-to-Interference-Noise-Ratio (SINR) where there exist relatively few works (Wan et al., 2009) on the three gossiping models. In the SINR model, we prove the NP-hardness of the MLG problem with unbounded-sized messages. In both the graph model and SINR model, we propose a constant factor approximation algorithm that yields schedules whose latency is bounded by \( O(\Delta + R) \), where \( \Delta \) is the maximum node degree of a network and \( R \) is its radius. We also study the performance of the algorithm through simulation.

4.2 Introduction

The gossiping problem has been the focus of many researchers as it is one of the crucial applications in Wireless Sensor Networks (WSNs) along with the broadcast problem. In a WSN that consists of a number of sensor nodes, the gossiping problem, which is also known as all-to-all broadcast, is to distribute the message of each node to all the other nodes
in the network, whereas the broadcast problem is to distribute a unique message from a source node to all the other nodes. As the small-sized sensors have limited energy resources, researchers have focused on reducing energy consumption while distributing data in a network so that the network life time is extended. An interesting approach is to assign timeslots to nodes to obtain a good (short) schedule thereby avoiding unnecessary transmissions. As data distribution may occur periodically, reducing the latency of the schedule, that is, constructing schedules with a minimum number of timeslots, has been a fundamental issue in such applications.

In the literature on the gossiping problem, there are three models: unit-sized, bounded-sized, or unbounded-sized messages. In the unit-sized-message model, a node can send a single unit-sized message, and therefore combining messages is not allowed. In the bounded-sized-message model, a node can combine messages that have been received so far (up to some limit), whereas in the unbounded-sized-message model, there is no limit on the length of the combined message.

The gossiping problem in the graph model has been investigated by many researchers over the last several years. In the collision-free graph model, for the unit-sized-message model, (Gandhi et al., 2003) introduced a 1974-approximation algorithm, and (Manne and Xin, 2006) proposed an optimal randomized schedule with a latency bounded by $O(n \log n)$, where $n$ is the number of sensor nodes in the network. Later, (Huang et al., 2008) introduced a 27-approximation algorithm which produces gossiping schedules with a latency bounded by $27(n + R - 1)$, where $R$ is the network radius. In (Gandhi et al., 2008), two approximation algorithms with constant factors of 20 and 34 have been studied. For the bounded-sized-message model, (Bar-yehuda et al., 1993) studied the gossiping problem where messages can be combined into a single message whose size is bounded by $\log n$, and (Christersson et al., 2002) gave an improvement over (Bar-yehuda et al., 1993). For the unbounded-sized-message model, (Cicalese et al., 2006) showed that their algorithm produces gossiping schedules
with $O\left(g + \frac{\nu \log n}{\log \nu - \log \log n}\right)$ timeslots, where $\nu = \Omega(\log n)$ and $g$ is the network diameter, and (Krzywdzinski, 2010) introduced a constant factor approximation algorithm whose latency is bounded by $7\Delta + 258R$. Although there have been many studies in the collision-free graph model for the gossiping problem, surprisingly there exists no study in the collision-interference-free graph model, to the best of our knowledge.

The graph model which has been used in many studies, however, is not an adequate model since cumulative interference caused by all the other concurrently transmitting nodes is ignored. Thus, researchers have started investigating problems in WSNs in the more realistic physical interference model which is known as the Signal-to-Interference-Noise-Ratio (SINR) model since its introduction by Gupta et al. in (Gupta and Kumar, 2000). For the SINR model, (Wan et al., 2009) introduced a constant factor approximation algorithm for the gossiping problem in the unit-sized-message model, and (Xin, 2010) proposed a $O(\log n)$-approximation algorithm for the unbounded-sized-message model. However, (Xin, 2010) considered only the concurrently sending nodes within some pre-defined interference area from a receiver, and therefore, some interference caused by senders located far away is ignored.

While these studies have been concerned with gossiping, some other researchers have focused on related applications such as data aggregation and broadcast. The following shows a summary of some related works.

- Collision-Free Model
  - Data Aggregation: (Chen et al., 2005; Huang et al., 2007; Wan et al., 2009; Xu et al., 2011)
  - Broadcast: (Gandhi et al., 2003; Huang et al., 2006, 2007; Gandhi et al., 2008)
• Collision-Interference-Free Model
  – Data Aggregation: (Wan et al., 2009; An et al., 2011)
  – Broadcast: (Chen et al., 2007; Huang et al., 2008; Calinescu and Tongngam, 2011)

• SINR Model
  – Data Aggregation: (Li et al., 2009, 2010; Lam et al., 2011; Halldórsson and Mitra, 2012; Hobbs et al., 2012)
  – Broadcast: (Huang et al., 2008)

Although there have been several approximation algorithms for these related problems, there are surprisingly few studies regarding the complexity of the problems. For the problem of data aggregation, NP-hardness was proved for the collision-free model by (Chen et al., 2005), and (An et al., 2011) and (Lam et al., 2011) showed not only an $\Omega(\log n)$ approximation lower bound for the problem, but also the NP-hardness in the collision-interference-free graph model and the geometric SINR model, respectively. For the broadcast problem, (Gandhi et al., 2003) proved its NP-hardness in the collision-free model. However, to the best of our knowledge, the NP-hardness of the broadcast problem in both the collision-interference-free and SINR models as well as that of the gossiping problem in all network models remain open. (Note that the existing NP-hardness of the broadcast problem in (Gandhi et al., 2003) holds for the collision-free graph model, but not for the other models.)

In this chapter, we continue the study of the gossiping problem with unbounded-sized messages in the graph model and the geometric SINR model. Extending the proof in (Lam et al., 2011), we show that this problem is NP-hard in the geometric SINR model. Additionally, for the uniform power model, we introduce a constant factor approximation algorithm yielding schedules whose latency is bounded by $O(\Delta + R)$ in both the graph model and the geometric SINR model. Our algorithm gives an improved approximation ratio of 224 over
the existing ratio of 285 given by (Krzywdzinski, 2010) in the collision-free graph model. In the collision-interference-free graph model and the SINR model, our approximation algorithm is the first one to the best of our knowledge. Moreover, regarding broadcast, the broadcast subroutine of our gossiping algorithm provides broadcast schedules with a better approximation ratio than existing ones given by (Chen et al., 2007; Huang et al., 2008) for the collision-interference-free model.

This chapter is organized as follows. Section 4.3 describes our network models and defines the Minimum Latency Gossiping (MLG) problem with unbounded-sized messages. In Section 4.4, we show the NP-hardness of MLG in the geometric SINR model. Section 4.5 introduces a constant factor approximation algorithm for the MLG problem and shows some simulation results. Section 4.6 contains some concluding remarks.

4.3 Preliminaries

4.3.1 Network Model

In this chapter, we model a wireless sensor network as \((V, D, p)\), where \(V\) represents a set of \(n\) nodes, and \(D : V \times V \rightarrow R^+\) represents the distance function between nodes. Letting \(p_{\text{max}} : V \rightarrow R^+\) be the maximum power level function, we define a power assignment function as \(p : V \rightarrow R^+, p(u) \leq p_{\text{max}}(u), u \in V\).

Graph Model

In the graph model, given a transmission power level \(p(u)\) for each node \(u\), let \(R^u_{p(u)} = \{v | v \in V, D(u, v) \leq p(u)\}\) denote the set of all nodes that can be reached by \(u\) with the power level \(p(u)\). Two nodes \(u\) and \(v\) can communicate only if they are in the coverage area of each other, i.e., \(u \in R^v_{p(v)}\) and \(v \in R^u_{p(u)}\). However, we also need to consider the collision or interference that interferes with the communication. Given a power level \(p(u)\) of \(u\), the
interference range of \( u \) is defined as \( \rho \cdot p(u) \), where \( \rho \geq 1 \) is the interference factor. Given \( \rho \geq 1 \), let \( R_{p(u)}^u = \{ v | v \in V, D(u, v) \leq \rho \cdot p(u) \} \) denote the set of all nodes in the interference range of \( u \). Then, collision (or conflict) is said to occur at a receiver node \( w \) if there exist other concurrently sending nodes \( u \) and \( v \) such that \( w \in R_{p(u)}^u \cap F_{p(v)}^v \), where \( \rho = 1 \). On the other hand, interference is said to occur at \( w \) if there exist other concurrently sending nodes \( u \) and \( v \) such that \( w \in R_{p(u)}^u \cap F_{p(v)}^v \), where \( \rho > 1 \).

In the literature, the graph model concerning only collision (i.e., when \( \rho = 1 \)) is called the collision-free model, whereas the graph model concerning both collision and interference (i.e., when \( \rho \geq 1 \)) is called the collision-interference-free model. In the graph model, the communication graph can be modeled as a bidirectional graph \( G(V, E) \), where \( E = \{ (u, v) | u, v \in V, D(u, v) \leq p(u) \text{ and } D(v, u) \leq p(v) \} \).

**SINR Model**

In the physical interference model (SINR) (Gupta and Kumar, 2000), if a node \( u \) transmits with its power level \( p(u) \), then the power received at another node \( v \) is \( p(u) \cdot D(u, v)^{-\alpha} \), where \( \alpha \), the path loss exponent, is commonly assumed to be in the interval \([2, 6]\). In order that node \( v \) can receive and decode data sent by \( u \), the ratio of the received power at \( v \) to the interference caused by all the other concurrently transmitting nodes and background noise must be beyond an SINR threshold \( \beta \geq 1 \). Formally, node \( v \) can receive data successfully via the link \((u, v)\) only if

\[
SINR(u, v) = \frac{p(u) \cdot D(u, v)^{-\alpha}}{N + I_v} \geq \beta
\]  

(4.1)

where \( N > 0 \) is the background noise, and

\[
I_v = \sum_{w \notin \{u, v\}, w \in X} p(w) \cdot D(w, v)^{-\alpha}
\]  

(4.2)

is the cumulative interference at \( v \) caused by nodes in \( X \) that is the set of other concurrently transmitting nodes. Observing that \( u \) can send its data to the nodes within the distance
\((\frac{p(u)}{NB})^{\frac{1}{\alpha}}\), the communication graph can be modeled as a directed graph \(G(V, E)\), where 
\(E = \{(u \rightarrow v)|u, v \in V, D(u, v) \leq (\frac{p(u)}{NB})^{\frac{1}{\alpha}}\}\).

### 4.3.2 Problem Definition

In this chapter, we are concerned with gossiping in the \textit{unbounded-sized-message} model, i.e., we assume that multiple messages can be combined as a single message, and there is no limit on the length of a message that one node can transmit.

The Minimum Latency Gossiping (MLG) problem is defined as follows. A schedule is defined as a sequence of timeslots, at each of which, several nodes are scheduled to send its data to its receivers. Formally, at each timeslot \(t\), we have an assignment vector \(\pi_t = \langle (s_{t_1}, p(s_{t_1})), \ldots, (s_{t_k}, p(s_{t_k})) \rangle\) in which \(s_{t_i}\) is assigned to send data with its power level \(p(s_{t_i})\), 
\(1 \leq i \leq k\), and

1. (Graph Model) neither collision nor interference occurs at any receiver \(r\), or
2. (SINR Model) the SINR threshold inequality is satisfied for all receivers \(r\),

where \((s_{t_i}, r)\) is an edge in the communication graph \(G(V, E)\), i.e., all the senders \(s_{t_i}\) can transmit concurrently.

A schedule is a sequence of assignment vectors \(\Pi = (\pi_1, \pi_2, \ldots, \pi_M)\), where \(M\) is the length of the schedule which is also known as its latency. A schedule \(\Pi\) is \textit{successful} if the message \(m(v)\) of each node \(v \in V\) is received by all the other nodes in the network. In a schedule, a node may be scheduled at several timeslots with different power levels. The MLG problem is defined as follows:

**Input.** A set of nodes \(V\), a distance function \(D\) which is defined as the Euclidean distance between nodes, a maximum power level function \(p\).

**Output.** A successful schedule of minimum length.
4.4 NP-Hardness

In this section, we prove the NP-hardness for the minimum latency gossiping (MLG) problem. The structure of this proof is similar to the one in the proof of the NP-hardness of the minimum latency aggregation problem in (Lam et al., 2011), which, as the proof of the NP-hardness of scheduling with power control in geometric SINR (Vlker et al., 2009), is based on a construction in (Goussevskaia et al., 2007). We restate the construction in (Lam et al., 2011) for reader’s convenience and omit the details.

In order to prove MLG’s NP-hardness, we construct a polynomial time reduction from the Partition problem which was proven NP-complete in (Karp, 1972). This decision problem is defined as follows. Given a finite set of distinct and positive integers, the objective is to determine if it is possible to divide this set into two subsets such that the sums of all integers in each subset are equal.

![Figure 4.1. The corresponding geometric MLG instance](image)

Let $I_p$ be an instance of Partition which consists of a set $S$ of $n$ distinct and positive integers $a_1, a_2, ... a_n$. Without loss of generality, assume that $a_1 < a_2 < ... < a_n$. We construct in polynomial time an instance $I_M$ of the MLG problem as follows.
In the instance $I_M$, we have $2n+3$ nodes including $2n$ nodes $s_i$ and $r_i$, $1 \leq i \leq n$, 2 nodes $s_{n+1}$ and $s_{n+2}$ and a center node $s$. These nodes are deployed on the plane at the following positions.

\[
\begin{align*}
pos(s) &= (0,0) \\
pos(s_{n+1}) &= (0, -\left(\frac{24P}{N\beta(A^\alpha A^\beta + 24)}\right)^{\frac{1}{\alpha}}) \\
pos(s_{n+2}) &= (-\left(\frac{24P}{N\beta(A^\alpha A^\beta + 24)}\right)^{\frac{1}{\alpha}}, 0)
\end{align*}
\]

and, for all $1 \leq i \leq n$,

\[
\begin{align*}
pos(s_i) &= \left(\frac{P}{b_i N^\beta}\right)^{\frac{1}{\alpha}}, 0) \\
pos(r_i) &= \left(\frac{P}{b_i N^\beta}\right)^{\frac{1}{\alpha}}, d_0)
\end{align*}
\]

where $P$ is the maximum power value to be defined below. Let $\sigma = \sum_{i=1}^{n} a_i$, $A = \left(\left(\frac{1}{\alpha_s - 1}\right)^{\frac{1}{\alpha}} - \left(\frac{1}{\alpha_n}\right)^{\frac{1}{\alpha}}\right)$, $b_i = \frac{a_i A^\alpha}{12\sigma}$ and $d_0 = \left(\frac{12 P \sigma}{(12\sigma N^{\beta+n(N^\beta)}\frac{1}{\alpha})}\right)^{\frac{1}{\alpha}}$.

With $d(u,v)$ denoting the Euclidean distance between $u$ and $v$, we define the maximum power level for the nodes as follow:

\[
\begin{align*}
p_{\text{max}}(s_i) &= p_{\text{max}}(s_{n+1}) = p_{\text{max}}(s_{n+2}) = P \\
p_{\text{max}}(r_i) &= N\beta d(r_i, r_{i+1})^\alpha, 1 \leq i \leq n, r_{n+1} \equiv s \\
p_{\text{max}}(s) &= N\beta d(s, s_1)^\alpha = \frac{P}{b_1} = \frac{12 P \sigma}{A_1 A^\alpha} > P
\end{align*}
\]

**Fact 4.4.1.** Let $T_i = \{s_j | 1 \leq j \leq n+1 \land i \neq j\}$. It holds for all $1 \leq i \leq n$ that $\text{SINR}(s_i, r_i)$ exceeds $\beta$ when node $s_i$ is assigned to send data to $r_i$ at the same timeslot as the nodes in $T_i$.

**Fact 4.4.2.** For all $1 \leq i \leq n$, $s_i$ can send data only to $r_i$.

**Fact 4.4.3.** $s_{n+1}$ and $s_{n+2}$ can send data only to $s$.

**Fact 4.4.4.** $r_{i+1}$ can receive data from $r_i$ (where $r_{n+1} \equiv s$) if and only if there is no other nodes sending in $r_i$’s timeslot.

**Fact 4.4.5.** It holds for all $1 \leq i < n$ that $r_i$ can send data to $s$ through $r_{i+1}$ only.
Fact 4.4.6. $s_1$ can receive data from $s$ if and only if there is no other node sending in that timeslot.

Lemma 4.4.1. $I_P$ has a solution if and only if $I_M$ has an optimal gossiping schedule of length $n + 3$.

Proof. Omitted. \hfill \square

From Lemma 4.4.1 we obtain

Theorem 4.4.1. The MLG problem is NP-hard.

4.5 Constant-Factor Approximation Algorithm

In this section, we introduce a constant factor approximation algorithm for the MLG problem in the graph model and the physical interference (SINR) model, assuming a uniform power level $P$, i.e., for all $v \in V$, $p(v) = P$. In those models, we make the following assumptions:

For the graph model, we set the maximum link length $r = P$, and assume that the undirected graph $G = (V, E)$, where $E = \{(u, v) | d(u, v) \leq r\}$, is connected and $\rho \geq 1$.

For the SINR model, define $r_{max} = \left(\frac{P}{N^\beta}\right)\frac{1}{\alpha}$ and notice that if node $u$ on link $(u, v)$ of length $r_{max}$ is transmitting, then none of the remaining nodes can transmit concurrently. Thus we are interested in links $(u, v)$, where $d(u, v) \leq \delta \left(\frac{P}{N^\beta}\right)^{\frac{1}{\alpha}}$ for some constant $\delta \in (0, 1)$ as considered in (Li et al., 2009). Thus, in the SINR model, we let $r$ be $\delta \left(\frac{P}{N^\beta}\right)^{\frac{1}{\alpha}}$, and assume that the undirected graph $G = (V, E)$, where $E = \{(u, v) | d(u, v) \leq r\}$, is connected and $\alpha > 2$ (Gupta and Kumar, 2000).

4.5.1 Algorithm

We first introduce some definitions and notations that are used subsequently:
• **Graph Center**: Given a communication graph \( G = (V, E) \), we call node \( c \) a center node if the distance from \( c \) to the farthest node from \( c \) is minimum.

• **Maximal Independent Set (MIS)**: A subset \( V' \subseteq V \) of the graph \( G \) is said to be independent if for any vertices \( u, v \in V', (u, v) \notin E \). An independent set is said to be maximal if it is not a proper subset of another independent set.

• **Connected Dominating Set (CDS)**: A dominating set (DS) is a subset \( V' \subseteq V \) such that every vertex \( v \) is either in \( V' \) or adjacent to a vertex in \( V' \). A DS is said to be connected if it induces a connected subgraph.

**Gossip Tree Construction**

Our algorithm assigns timeslots to nodes based on a gossip tree whose construction is based on that of a data aggregation tree in (Li et al., 2009). After a center node \( c \) is chosen, we construct a breadth-first-search (BFS) tree (cf. (Cormen et al., 2009)) on \( G \) rooted at node \( c \) so that the latency can be bounded in terms of the network radius \( R \) rather than its diameter. (Li et al., 2009) finds an MIS on \( G \) using an algorithm in (Wan et al., 2002) based on the BFS tree. We call nodes in the MIS dominators and the others dominatees. The constructed MIS satisfies the property that the distance between any pair of its complementary subset is exactly two hops (Wan et al., 2002). Next, we connect the dominators using some connectors that were originally dominatees to obtain a CDS of \( G \). If there exist some remaining dominatees that are not connected to the CDS, then each of such dominatees is connected to its neighboring dominator that has the smallest hop distance to \( c \) in the BFS tree. We denote the newly formed tree by \( T \), and use it as the gossip tree in our algorithm.

**Gossip Scheduling Algorithm**

Our algorithm (Algorithm 8) starts by partitioning the network into square cells each of which has diagonal length \( r \), and labels each cell with the label \( CL(i,j) \) if its upper-left
corner has coordinate \((i, j)\). Once the gossip tree \(T\) is obtained, we gather data from each node to the center node \(c\) (Steps 4 – 5 in Algorithm 8). Node \(c\) then broadcasts the collected data combined with its own based on \(T\) (Steps 6 – 14 in Algorithm 8). Assigning timeslots for data gathering and broadcasting is based on a constant \(C\) which guarantees that two senders (receivers) can transmit (receive) data successfully if they are at least \(C\) cells apart from each other. The constant \(C\) is set as follows in the two different models:

- **Graph model:** \(C = \lceil \rho \cdot \sqrt{2} + 2 \rceil\)
- **SINR model:** \(C = \lceil (\frac{P}{N(\delta - \alpha - 1)(\alpha - 2)} \cdot \sqrt{2} \cdot \delta^{-1} (\frac{N \beta}{P})^{\frac{1}{\alpha}} + 2 \rceil\)

### Algorithm 8 Gossiping

**Input:** A set \(V\) of nodes with a uniform power level \(P\)

**Output:** Length of schedule

1. Partition the network into square cells each of which has diagonal length \(r\).
2. Construct a gossip tree \(T\) using the algorithm in (Li et al., 2009).
3. Set the first timeslot \(t \leftarrow 1\).
4. // Data Gathering starts.
5. \(t \leftarrow \text{CCA}(V, T, t)\)
6. // Broadcasting start.
7. \(TS(c) \leftarrow TS(c) \cup \{t\}\)
8. \(t \leftarrow t + 1\)
9. for \(i = 1\) to \(R - 1\) do
10. \(S_i \subseteq V\) be the set of connectors at level \(i\) in \(T\).
11. if \(S_i \neq \emptyset\) then \(t \leftarrow \text{RBS}(S_i, t)\) end if
12. \(S_{i+1} \subseteq V\) be the set of dominators at level \(i + 1\) in \(T\).
13. if \(S_{i+1} \neq \emptyset\) then \(t \leftarrow \text{SBS}(S_{i+1}, t)\) end if
14. end for
15. return \(t - 1\)

**Data Gathering.** In order to gather data to \(c\), we use the data aggregation algorithm called *Cell Coloring* algorithm in (An et al., 2011), which is included in Algorithm 9. The *Cell Coloring* algorithm, which is based on an algorithm in (Li et al., 2009), is originally
built for the graph model assuming $\rho \geq 1$. We use the algorithm not only for the graph model, but also for SINR model where the constant $C$ is defined accordingly.

**Broadcast.** Once all the data is gathered to $c$, $c$ broadcasts the collected data combined with its own to the whole network. We introduce a new broadcast scheduling algorithm which is also based on the data aggregation algorithm of (Li et al., 2009). The details of our broadcast algorithm are contained in Algorithm 8 (Steps 6–14). In Step 7, $c$ (which is also a dominator) broadcasts the data to its neighbors (lower level connectors). In Steps 10–11, the connectors which have just received data from the upper level dominators relay the data to their lower level dominators. These steps are based on the receivers’ locations (Algorithm 11), i.e., the connectors whose receivers (lower level dominators) are $C$ cells apart from each other are assigned to the same timeslot. Then, in Steps 12–13, the dominators relay the data to its dominatees and lower level connectors. These dominators are scheduled based on their (senders’) locations (Algorithm 10, (An et al., 2011)), i.e., the dominators which are $C$ cells apart from each other are assigned to the same timeslot. These Steps 10–13 are repeated until the data is disseminated to the whole network.

In Algorithms 8, 10 and 11, $TS(v)$ denotes the set of timeslots at which node $v$ is activated to send data.

**Algorithm 9** Cell Coloring Algorithm (CCA)

**Input:** A set $V$ of nodes, a tree $T$ and a starting timeslot $t$  
**Output:** Timeslot $t$

1: Let $S \subseteq V$ be the set of dominatees in $T$.  
2: if $S \neq \emptyset$ then $t \leftarrow \text{SBS}(S, t)$ end if  
3: for $i = R$ to 2 do  
4: Let $S_i \subseteq V$ be the set of dominators at level $i$ in $T$.  
5: if $S_i \neq \emptyset$ then $t \leftarrow \text{SBS}(S_i, t)$ end if  
6: Let $S_{i-1} \subseteq V$ be the set of connectors at level $i - 1$ in $T$.  
7: if $S_{i-1} \neq \emptyset$ then $t \leftarrow \text{SBS}(S_{i-1}, t)$ end if  
8: end for  
9: return $t$
Algorithm 10 AssignTimeSlot (SBS)
Input: A set $S$ of nodes and a starting timeslot $t$
Output: Timeslot $t$

1: while $S \neq \emptyset$ do
2:   Pick one node $v_s \in S$ in each cell. Let $S' \subseteq S$ be the set of such nodes.
3:   for $c_1 = 0, ..., C - 1$ and $c_2 = 0, ..., C - 1$ do
4:     $X \leftarrow \emptyset$, $X \leftarrow \{v_s|v_s \in S'$ with $CL(x, y)$ such that $c_1 = x \mod C$ and $c_2 = y \mod C\}$
5:     if $X \neq \emptyset$ then
6:       for each $v_s \in X$ do
7:         $TS(v_s) \leftarrow TS(v_s) \cup \{t\}$
8:       end for
9:     end if
10:   $t \leftarrow t + 1$, $S \leftarrow S \setminus X$
11: end for
12: end while
13: return $t$

Algorithm 11 Receiver-based Scheduling (RBS)
Input: A set $S$ of nodes and a starting timeslot $t$
Output: Timeslot $t$

1: for each $v_s \in S$ do
2:   $Z \leftarrow Z \cup v_s$’s lower level dominators in $T$
3: end for
4: for $c_1 = 0, ..., C - 1$ and $c_2 = 0, ..., C - 1$ do
5:   $X \leftarrow \emptyset$, $X \leftarrow \{v_r|v_r \in Z$ with $CL(x, y)$ such that $c_1 = x \mod C$ and $c_2 = y \mod C\}$
6:   if $X \neq \emptyset$ then
7:     for each $v_r \in X$ do
8:       $v_s \leftarrow v_r$’s upper level connector in $T$
9:     $TS(v_s) \leftarrow TS(v_s) \cup \{t\}$
10:   end for
11: $t \leftarrow t + 1$
12: end if
13: end for
14: return $t$

4.5.2 Analysis

In this section, we analyze Algorithm 8 and bound the latency of the gossip schedules produced by it. First, we set the constant value $C$ for the graph model and the SINR model based on (An et al., 2011, 2012a).
Lemma 4.5.1 (Graph Model). (An et al., 2011) Let \( C = \lceil \rho \cdot \sqrt{2} + 2 \rceil \), where \( \rho \geq 1 \) is the interference factor. Then any two sender (receiver) nodes that are at least \( C \) cells apart from each other can concurrently send (receive) data without any collision and interference.

Lemma 4.5.2 (SINR Model). (An et al., 2012a) For SINR threshold \( \beta \geq 1 \), path loss exponent \( \alpha > 2 \), background noise \( N > 0 \), and some constant \( \delta \in (0, 1) \), let

\[
C = \lceil \left( \frac{P_{2\pi}}{N N (\delta - \alpha - 1) (\alpha - 2)} \right)^\frac{1}{\alpha - 2} \cdot \sqrt{2} \cdot \delta^{-1} (\frac{\beta}{P})^{\frac{1}{\alpha}} + 2 \rceil
\]

Then any two sender nodes that are at least \( C \) cells apart from each other can concurrently send (receive) data without interference.

Next, we prove that the latency of a gossip schedule found by Algorithm 8 is bounded by \( O(\Delta + R) \). We need the following lemmas.

Lemma 4.5.3. (Li et al., 2009) The number of connectors in a cell is at most 12.

Lemma 4.5.4. In Algorithm 8, gathering data from all the other nodes to center node \( c \) takes at most \( \Delta \cdot C^2 + 6 \cdot C^2 \cdot R \) timeslots.

Proof. First consider the dominatees in each cell and their dominator \( v \) (Steps 1–2 in Algorithm 9). Obviously, there are at most \( \Delta \) dominatees in each cell, and one of those \( \Delta \) dominatees must be a connector to connect the dominator \( v \) to another dominator. Therefore, the number of dominatees is bounded by \( \Delta - 1 \), and gathering data from all the dominatees to the corresponding dominators takes at most \( (\Delta - 1)C^2 \) timeslots.

Next, consider the dominators at level \( i \) (Steps 4–5 in Algorithm 9). Since there is at most one dominator in each cell, gathering data from all the dominators at level \( i \) to the connectors at level \( i - 1 \) takes \( C^2 \) timeslots. As this process is repeated at most \( \frac{R}{2} \) times, gathering data from all dominators to upper level connectors takes at most \( \frac{R}{2} \cdot C^2 \) timeslots.

Now, let us consider only the connectors at level \( j \), where \( 1 < j < R \) (Steps 6–7 in Algorithm 9). In one cell, at most 11 of those connectors at level \( j \) have the role of sending
the collected data to their dominators at level $j - 1$; one remaining connector in the cell must relay data from the dominator at level $j - 1$ to another dominator at level $j - 3$. Therefore, gathering data from the connectors at level $j$ to the dominators at level $j - 1$ takes at most $11C^2$ timeslots. As this process is repeated $\frac{R-2}{2}$ times, it takes at most $\frac{R-2}{2} \cdot 11C^2$ timeslots. On the other hand, all the connectors at level 1 send data to a sink, and this requires at most $12C^2$ timeslots.

Thus, the latency of data gathering is at most $(\Delta - 1)C^2 + \frac{R}{2} \cdot C^2 + \frac{R-2}{2} \cdot 11 \cdot C^2 + 12C^2 = \Delta \cdot C^2 + 6 \cdot C^2 \cdot R$. 

\begin{lemma}
In Algorithm 8, broadcasting data takes at most $1 + C^2 \cdot (R - 1)$ timeslots.
\end{lemma}

\begin{proof}
First, note that in Steps 7–8 in Algorithm 8, the center node sends data to its neighbors in one timeslot. Next, consider the connectors at level $i$ (Steps 10–11). Any two of those connectors cannot share one dominator at level $i + 1$ on the gossip tree $T$; otherwise a cycle would result. This means that those connectors can be scheduled based on the receivers’ locations, i.e., the locations of the corresponding dominators at level $i + 1$. As there is at most 1 dominator in each cell, relaying data from the connectors at level $i$ to the dominators at level $i + 1$ takes at most $C^2$ timeslots. Since Steps 10–11 are repeated $\frac{R-1}{2}$ times, at most $\frac{R-1}{2} \cdot C^2$ timeslots are needed.

Now consider the dominators at level $j$ (Steps 12–13). Since at most one dominator can be located in a cell, sending data from the dominators to the connectors at level $j + 1$ takes at most $C^2$ timeslots. As the Steps 12–13 are repeated $\frac{R-1}{2}$ times, it takes at most $\frac{R-1}{2} \cdot C^2$ timeslots.

Thus, the latency of broadcasting is at most $1 + \frac{R-1}{2} \cdot C^2 + \frac{R-1}{2} \cdot C^2 = 1 + C^2 \cdot (R - 1)$. 

\begin{lemma} \textit{(Lower bound).} \textit{(Krzywdzinski, 2010)} If $\Delta$ is the maximum node degree in a network, then every gossip schedule with unbounded-size messages has at least $\Delta + R - 1$ timeslots.
\end{lemma}
Theorem 4.5.1. Algorithm 8 produces gossip schedules whose latency is bounded by $C^2 \cdot \Delta + 7 \cdot C^2 \cdot R + (1 - C^2) = O(\Delta + R)$, and it is a constant-factor approximation with the factor of $14 \cdot C^2$.

Proof. By Lemma 4.5.4 and Lemma 4.5.5, the latency, denoted by $SOL$, of schedules produced by Algorithm 8 is bounded by $C^2 \cdot \Delta + 7 \cdot C^2 \cdot R + (1 - C^2)$. Next, without loss of generality, assume that $n \geq 2$, and therefore $\Delta \geq 1$ and $R \geq 1$. Then, denoting the lower bound by $OPT$, the approximation ratio is

$$\frac{SOL}{OPT} \leq \frac{C^2 \Delta + 7 \cdot C^2 \cdot R + (1 - C^2)}{\Delta + R - 1} \leq \frac{7 \cdot C^2 \cdot \Delta + 7 \cdot C^2 \cdot R}{\Delta + R - 1}$$

$$= 14 \cdot C^2$$

Note that this approximation ratio is 224 in the collision-free graph model, an improvement on the approximation ratio of 258 given by (Krzywdzinski, 2010). To the best of our knowledge, in the collision-interference-free graph model and the SINR model, our results are the first constant-factor approximation algorithms.

Also note that our broadcast scheduling algorithm yields an approximation ratio of $C^2$ by Lemma 4.5.5. In the collision-interference-free graph model, it is $C^2 = \lceil \rho \sqrt{2} + 2 \rceil^2$ which improves on the approximation ratio of $6\lceil \frac{2}{3}(\rho + 2) \rceil^2$ given by (Huang et al., 2008). In addition, (Chen et al., 2007) studies the broadcast algorithm with $\rho = 2$ giving an approximation ratio of 26, whereas our approximation ratio is 25.

4.5.3 Simulation

In our simulation, networks are generated randomly in the Euclidean plane where the number of nodes is 500. The nodes are randomly deployed on an area of size $m \times m$, where
\(m = 300, 400\) and 500. For each \(m\), we generate 100 different networks, and average the latencies produced by the algorithm over the networks. For the simulation, we set the various parameters as follows:

- **Graph model:**
  - Choice of \(\rho\): We use \(\rho = \{1, 2, \cdots, 10\}\)
  - Initial power assignment: We first use Kruskal’s algorithm (Cormen et al., 2009) to find the minimum spanning tree \(T_{MST}\) using edge weights defined as the distance between any two nodes. Then, we set the initial power \(P = r\), where \(r\) is the length of the longest edge in \(T_{MST}\). Given the initial power assignment, we obtain the initial graph \(G = (V, E)\), where \(E = \{(u, v)|d(u, v) \leq r\}\).

- **SINR model:**
  - Choices of SINR parameters: We use \(\alpha = 5, N = 1\) and \(\beta = 1\).
  - Choice of \(\delta\): We use \(\delta = \{0.1, 0.2, \cdots, 0.9\}\).
  - Initial power assignment: Given a uniform power assignment, if \(\delta\) is too small, the graph may not be connected. In order to make the initial graph connected even with the smallest \(\delta = 0.1\) in our simulation, we set the initial power \(P = \beta N(\frac{r}{\delta})^\frac{1}{n}\), and obtain the initial graph \(G = (V, E)\), where \(E = \{(u \rightarrow v)|d(u, v) \leq \delta(\frac{P}{N\beta})^\frac{1}{2}\}\).

Table 4.1 shows the performance of Algorithm 8 in the graph model. For fixed \(\rho\), as the network becomes denser (i.e., the network node degree becomes larger and the network radius smaller) the latency decreases, whereas as the network becomes sparser (i.e., the network node degree becomes smaller and the network radius larger) the latency increases. For fixed network density, as \(\rho\) becomes larger (i.e., the interference range becomes wider), the latency increases.
Table 4.1. Latencies of Algorithm 8 in Graph Model

<table>
<thead>
<tr>
<th>$\rho$</th>
<th>$300 \times 300$</th>
<th>$400 \times 400$</th>
<th>$500 \times 500$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>313.13</td>
<td>321.52</td>
<td>321.59</td>
</tr>
<tr>
<td>2</td>
<td>357.85</td>
<td>368.71</td>
<td>369.98</td>
</tr>
<tr>
<td>3</td>
<td>430.32</td>
<td>440.18</td>
<td>440.23</td>
</tr>
<tr>
<td>4</td>
<td>461.79</td>
<td>469.39</td>
<td>469.41</td>
</tr>
<tr>
<td>5</td>
<td>513.79</td>
<td>522.64</td>
<td>523.09</td>
</tr>
<tr>
<td>6</td>
<td>536.89</td>
<td>542.38</td>
<td>544.81</td>
</tr>
<tr>
<td>7</td>
<td>557.04</td>
<td>562.19</td>
<td>564.57</td>
</tr>
<tr>
<td>8</td>
<td>596.11</td>
<td>599.17</td>
<td>601.54</td>
</tr>
<tr>
<td>9</td>
<td>613.90</td>
<td>617.61</td>
<td>619.30</td>
</tr>
<tr>
<td>10</td>
<td>642.07</td>
<td>645.58</td>
<td>647.34</td>
</tr>
</tbody>
</table>

Table 4.2 shows the performance of Algorithm 8 in the SINR model. For fixed $\delta$, as the network becomes denser (i.e., the network node degree becomes larger, but the network radius becomes smaller) the latency decreases, whereas as the network becomes sparser (i.e., the network node degree becomes smaller, but the network radius becomes larger) the latency increases. For fixed density, as $\delta$ becomes smaller (i.e., the network node degree becomes smaller, but the network radius becomes larger), the latency increases, whereas as $\delta$ becomes larger (i.e., the network node degree becomes larger, but the network radius becomes smaller), the latency decreases.

Table 4.2. Latencies of Algorithm 8 in SINR Model

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>$300 \times 300$</th>
<th>$400 \times 400$</th>
<th>$500 \times 500$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>654.91</td>
<td>662.98</td>
<td>663.09</td>
</tr>
<tr>
<td>0.2</td>
<td>651.21</td>
<td>660.08</td>
<td>662.20</td>
</tr>
<tr>
<td>0.3</td>
<td>646.19</td>
<td>648.97</td>
<td>651.39</td>
</tr>
<tr>
<td>0.4</td>
<td>605.93</td>
<td>615.83</td>
<td>615.86</td>
</tr>
<tr>
<td>0.5</td>
<td>512.63</td>
<td>523.91</td>
<td>524.17</td>
</tr>
<tr>
<td>0.6</td>
<td>401.23</td>
<td>414.80</td>
<td>414.91</td>
</tr>
<tr>
<td>0.7</td>
<td>341.18</td>
<td>354.35</td>
<td>355.08</td>
</tr>
<tr>
<td>0.8</td>
<td>295.06</td>
<td>305.52</td>
<td>307.22</td>
</tr>
<tr>
<td>0.9</td>
<td>264.31</td>
<td>274.54</td>
<td>275.66</td>
</tr>
</tbody>
</table>
From these tables, we can observe that having smaller network radius rather than having smaller network node degree gives better results. This is because the variation of network node degree affects only the latencies of data aggregation schedules, whereas the variation of the network radius affects the latencies of the schedules for data aggregation as well as broadcast.

4.6 Conclusion

In this chapter, we have studied the Minimum Latency Gossiping (MLG) problem with unbounded-sized messages in the graph model and SINR model. We have proved the NP-hardness of the problem in the SINR model, and proposed a constant factor approximation algorithm whose latency is bounded by $O(\Delta + R)$ in both graph and SINR models assuming a uniform power level. We have also studied the performance of the algorithm through simulation. As to future work, we plan to study the gossiping problem with unit-sized and bounded-sized messages.
CHAPTER 5
CONNECTIVITY PROBLEM IN WIRELESS SENSOR NETWORKS IN THE SINR MODEL

5.1 Abstract

In this chapter, we study the Minimum Channel Assignment (MCA) problem for strong connectivity in wireless sensor networks in the physical model known as Signal-to-Interference-Noise-Ratio (SINR). The main issue is to compute a minimum channel assignment that yields a strongly-connected communication graph spanning all nodes such that the nodes assigned to the same channel can communicate without interference in the SINR model. The complexity measure is the number of channels, and our objective is to minimize it. We show the NP-hardness of the MCA problem, and propose an algorithm that compute a channel assignment for 2-dimensional grid networks. The algorithm produces an assignment with a constant number of channels for the network. We also propose two constant-factor approximation algorithms that yield channel assignments in which the number of channels is bounded by $O(\Delta)$, where $\Delta$ is the maximum node degree of a network. We also study the performance of the algorithms through simulation.

5.2 Introduction

In Wireless Sensor Networks (WSNs), the Minimum Channel Assignment (MCA) problem concerning the strong connectivity has been the focus of many researchers as it represents...
the theoretically achievable efficiency of MAC layer protocols (Moscibroda and Wattenhofer, 2006). Given a set of nodes on the plane, the MCA problem is to compute a channel assignment that constructs a strongly connected communication graph spanning all nodes such that sender nodes assigned the same channel can communicate with its receivers without interference in the SINR model. Here, the complexity measure is the number of channels used in the assignment, and the objective is to minimize it.

As WSNs are commonly modeled as graphs in the literature, researchers have adopted two interference models: graph-based model and physical interference model. Although the graph-based model has been widely used in many studies, it is not an adequate model since cumulative interference caused by all the other concurrently transmitting nodes is ignored. Thus, researchers have started focusing on the more realistic physical interference model which is known as the Signal-to-Interference-Noise-Ratio (SINR) model since its introduction by Gupta and Kumar in (Gupta and Kumar, 2000). Unlike the graph-based model, in the SINR model, the signal sent by a node $u$ fades, and background noise and signals sent concurrently by other nodes can interfere with $u$’s signal. Thus, the signal sent to node $v$ may not be strong enough to be received and, hence, transmitted data is lost.

For the SINR model, the MCA problem has been investigated by several researchers over the last few years. (Moscibroda and Wattenhofer, 2006) was the first study which introduced an algorithm using $O(\log^4 n)$ channels, where $n$ is the number of nodes in a network. On a special network known as the highway network, (Moscibroda and Wattenhofer, 2006) also showed the deficiency of the uniform power model where all nodes are assigned a uniform power level, and the linear power model where for a sender $s$ and a receiver $r$, a power level of $s$ is set as $\rho \cdot d(s,r)^\alpha$, where $\rho > 1$ is a constant, $d(s,r)$ is the distance between $s$ and $r$, and $\alpha > 2$ is the path loss exponent. Later, (Moscibroda, 2007) proposed a better algorithm using $(\log^2 n)$ channels, and (Halldórsson and Mitra, 2012) improved the bound to $O(\log n)$. These works studied the problem assuming arbitrary power levels, whereas (Avin
et al., 2009) studied it on 1- and 2-dimensional networks with a uniform power level ignoring the background noise, and showed that the number of channels needed on the networks is constant.

While these studies have been concerned with assigning minimum number of channels inducing a strongly connected communication graph, some other researchers have focused on other applications such as scheduling (Grönkvist and Hansson, 2001; Goussevskiaia et al., 2007; Lebhar and Lotker, 2009; Goussevskiaia et al., 2009), data aggregation (Li et al., 2009, 2010; Lam et al., 2011) and broadcast (Huang et al., 2008) in the SINR model. Although there have been several approximation algorithms for these problems in the SINR model, there are surprisingly few studies regarding the complexity of the problems. (Goussevskiaia et al., 2007) proved that the scheduling problem in the geometric SINR model is NP-hard without power control, and later (Vlker et al., 2009) extended the NP-hardness result for the case with power control assuming arbitrary power levels. For the problem of data aggregation, (Lam et al., 2011) showed not only an \( \Omega(\log n) \) approximation lower bound for the problem in the metric SINR model, but also the NP-hardness in the geometric SINR model.

In this chapter, we continue the study of the MCA problem for strong connectivity in the geometric SINR model. Based on the proof in (Lam et al., 2011), we show that this problem is NP-hard in the geometric SINR model. Additionally, we also introduce the first constant factor approximation algorithms yielding channel assignments in which the number of channels used is bounded by \( O(\Delta) \) in the uniform power model.

This chapter is organized as follows. Section 5.3 describes our network model and defines the Minimum Channel Assignment (MCA) problem for strong connectivity in geometric SINR. In Section 5.4, we show its NP-hardness whereas Section 5.5 studies the MCA problem in 2-dimensional networks. Section 5.6 introduces two constant-factor approximation algorithms for the MCA problem and shows some simulation results. Section 5.7 contains some concluding remarks.
5.3 Preliminaries

5.3.1 Network Model

In this chapter, a wireless sensor network is modeled as \((V, D, p)\), where \(V\) represents a set of \(n\) nodes, and \(D : V \times V \rightarrow R^+\) represents the distance function between two nodes. Letting \(p_{\text{max}} : V \rightarrow R^+\) is the maximum power level function, we define \(p : V \rightarrow R^+, p(u) \leq p_{\text{max}}(u), u \in V\), as a power assignment function.

Our network model adopts the physical interference model (SINR) (Gupta and Kumar, 2000) where if a node \(u\) transmits its data to its receiver \(v\) with its power \(p(u)\), the received power at \(v\) is \(\frac{p(u)}{D(u,v)^\alpha}\), where \(\alpha \in [2, 6]\) is the path loss exponent. In the SINR model, the receiver \(v\) can receive and decode data from \(u\) only if the ratio of the received power at \(v\) to the cumulative interference caused by all the other concurrently transmitting nodes and background noise is beyond an SINR threshold \(\beta \geq 1\). Formally, the receiver \(v\) can receive data without interference via the link \((u,v)\) only if

\[
SINR_{(u,v)} = \frac{\frac{p(u)}{D(u,v)^\alpha}}{N + \sum_{w \in X - \{u,v\}} \frac{p(w)}{D(w,v)^\alpha}} \geq \beta
\]

where \(N > 0\) is the background noise, and \(X\) is the set of other concurrently transmitting nodes. Observing that \(u\) can send its data to the nodes within the distance \(\left(\frac{p(u)}{N^\alpha}\right)^{\frac{1}{\alpha}}\), the network can be modeled as a directed graph \(G(V,E)\), where \(E = \{(u \rightarrow v) | u, v \in V, D(u,v) \leq \left(\frac{p(u)}{N^\alpha}\right)^{\frac{1}{\alpha}}\}\).

5.3.2 Problem Definition

The Minimum Channel Assignment (MCA) problem concerning strong connectivity is defined as follows. Consider a set of nodes in the plane. We assign these nodes a number of channels (Kowalski and Rokicki, 2010) (also called timeslots or frequencies (Moscibroda and Wattenhofer, 2006; Moscibroda, 2007; Halldórsson and Mitra, 2012)) such that nodes assigned the same channel can send data to its receivers simultaneously without interference.
Formally, for each channel $c$, we have an assignment set $\pi_c = \{(s_1, p(s_1)), \ldots, (s_k, p(s_k))\}$ where the nodes $s_i$ can send data with power level $p(s_i)$, $1 \leq i \leq k$, and the SINR inequality is satisfied for its receiver $r$, where $(s_i, r)$ is a link in the graph $G(V,E)$, i.e., all the senders $s_i$ can transmit simultaneously without interference. A channel assignment is a collection of assignment sets $\Pi = \{\pi_1, \pi_2, \ldots, \pi_M\}$, and $M$ is its size. A channel assignment $\Pi$ is successful if the union of all links, i.e., the union of pairs of senders and receivers in the channel assignment, induces a strongly connected graph.

**Input.** A wireless sensor network where $V$ is the set of nodes, $D$ is the distance function defined as the Euclidean distance between nodes, and $p_{max}$ is the maximum power level function.

**Output.** A minimum successful channel assignment.

### 5.4 NP-Hardness

In this section, we show the NP-hardness of the Minimum Channel Assignment (MCA) problem. The structure of this proof is similar to the one in the proof of the NP-hardness of the minimum latency aggregation problem in (Lam et al., 2011). We restate the construction in (Lam et al., 2011) for reader’s convenience and omit the details.

In order to prove MCA’s NP-hardness, we construct a polynomial time reduction from the Partition problem which was proven NP-complete in (Karp, 1972). This decision problem is defined as follows. Given a finite set of distinct and positive integers, the objective is to determine if it is possible to divide this set into two subsets such that the sums of all integers in each subset are equal.

Let $I_P$ be an instance of the Partition problem consisting of a set $S$ of $n$ distinct and positive integers $a_1, a_2, \ldots, a_n$. Without loss of generality, assume that $a_1 < a_2 < \ldots < a_n$. We construct in polynomial time an instance $I_M$ of the MCA problem as follows.
Figure 5.1. The corresponding geometric MCA instance

In the instance $I_M$, we have $2n + 3$ nodes including $2n$ nodes $s_i$ and $r_i$, $1 \leq i \leq n$, 2 nodes $s_{n+1}$ and $s_{n+2}$ and a center node $s$. These nodes are deployed on the plane at the following positions:

$$
pos(s) = (0, 0)$$
$$
pos(s_{n+1}) = (0, -\left(\frac{24P}{N\beta(A^n\beta+24)}\right)^{\frac{1}{\alpha}})$$
$$
pos(s_{n+2}) = \left(\left(-\frac{24P}{N\beta(A^n\beta+24)}\right)^{\frac{1}{\alpha}}, 0\right)$$

and for all $1 \leq i \leq n$,

$$
pos(s_i) = \left(\left(\frac{P}{b_iN\beta}\right)^{\frac{1}{\alpha}}, 0\right)$$
$$
pos(r_i) = \left(\left(\frac{P}{b_iN\beta}\right)^{\frac{1}{\alpha}}, d_0\right)$$

where $P$ is the maximum power value to be defined below. Let $\sigma = \sum_{i=1}^{n} a_i$, $A = \left(\left(\frac{1}{a_{n-1}}\right)^{\frac{1}{\alpha}} - \left(\frac{1}{a_n}\right)^{\frac{1}{\alpha}}\right)$, $b_i = \frac{a_iA^\alpha}{12\sigma}$ and $d_0 = \left(\left(\frac{12P\sigma}{12\sigma N\beta+nN\beta}\right)^{\frac{1}{\alpha}}\right)\frac{1}{\alpha}$.

With $d(u, v)$ denoting the Euclidean distance between $u$ and $v$, we define the maximum power level for the nodes as follow:

$$
p_{\text{max}}(s_i) = p_{\text{max}}(s_{n+1}) = p_{\text{max}}(s_{n+2}) = P$$
$$
p_{\text{max}}(r_i) = N\beta d(r_i, r_{i+1})^\alpha, 1 \leq i \leq n, r_{n+1} \equiv s$$
$$
p_{\text{max}}(s) = N\beta d(s, s_1)^\alpha = \frac{P}{b_1} = \frac{12P\sigma}{a_1A^\alpha} > P$$
Fact 5.4.1. Let $T_i = \{s_j | 1 \leq j \leq n+1 \wedge i \neq j\}$. It holds for all $1 \leq i \leq n$ that $\text{SINR}(s_i, r_i)$ exceeds $\beta$ when node $s_i$ is assigned the same channel to send data to $r_i$ as the nodes in the set $T_i$.

Fact 5.4.2. For all $1 \leq i \leq n$, $s_i$ can send data only to $r_i$.

Fact 5.4.3. $s_{n+1}$ and $s_{n+2}$ can send data only to $s$.

Fact 5.4.4. $r_{i+1}$ can receive data from $r_i$ (where $r_{n+1} \equiv s$) if and only if there is no other concurrently sending node using the same channel as $r_i$’s.

Fact 5.4.5. It holds for all $1 \leq i < n$ that $r_i$ can send data to $s$ through $r_{i+1}$ only.

Fact 5.4.6. $s_1$ can receive data from $s$ if and only if there is no other concurrently sending node using the same channel as $s$’s.

Lemma 5.4.1. $I_p$ has a solution if and only if $I_M$ has a channel assignment whose size is $n + 3$.

Proof. Omitted. □

Theorem 5.4.1. The MCA problem is NP-hard.

Proof. Follows immediately from Lemma 5.4.1. □

5.5 Minimum Channel Assignment on 2-D Grid Networks

In this section, we study the Minimum Channel Assignment (MCA) problem on 2-dimension grid networks, assuming a uniform power level $P$, i.e., for all $v_i \in V$, $p(v_i) = P$. From the SINR inequality (5.1), we can compute the maximum link length as $r_{\text{max}} = \left(\frac{P}{\beta N}\right)^{\frac{1}{\alpha}}$. Considering a pair of a sender $v_i$ and a receiver $v_j$ on the link $(v_i, v_j)$, where $d(v_i, v_j) = r_{\text{max}}$, when $v_i$ uses its channel $c$ to send its data to $v_j$, other nodes cannot be assigned the same
channel \( c \); only \( v_i \) can be a sender using the channel \( c \). Thus, the links whose length is \( r_{\text{max}} \) are not considered, and we are interested in links \((v_i, v_j)\), where \( d(v_i, v_j) \leq \delta \left( \frac{P}{\beta N} \right)^{\frac{1}{\alpha}} \) for some constant \( \delta \in (0, 1) \) as in (Lam et al., 2011).

We consider a network in which each node \( v_{i,j} \in V \) is located at position \((i, j)\), where \( i, j = 0, 1, ..., \) and \( d(v_{i,j}, v_{i+1,j}) = d(v_{i,j}, v_{i,j+1}) = 1 \). The network is represented as a 2-dimensional grid \( G_2 = (V, E) \), where \( E = \{(v_{i,j} \rightarrow u_{i,j})|d(v_{i,j}, u_{i,j}) = 1 = \delta \left( \frac{P}{\beta N} \right)^{\frac{1}{\alpha}}\} \), and we show that there exists a successful channel assignment algorithm yielding channel assignments of size \( O(1) \). In order to show that, we need the following lemma.

**Lemma 5.5.1.** Let \( H_2 = \lceil \left( \frac{\beta \cdot 2\pi}{(1-\delta^\alpha)(\alpha-2)} \right)^{\frac{1}{\alpha-2}} + 1 \rceil \) for a SINR threshold \( \beta \geq 1 \), path loss exponent \( \alpha > 2 \), background noise \( N > 0 \), and some constant \( \delta \in (0, 1) \). Then, any two sender nodes that are at least \( H_2 \) hops away from each other on \( G_2 \) can be assigned the same channel.

**Proof.** Let us first bound the shortest distance between any two sender nodes that can be assigned the same channel. Considering a sender node \( v_i \) trying to send its data to a receiver \( v_j \) using a channel \( c \), in order that the receiver \( v_j \) receives data from the sender \( v_i \) without interference, for all other sender nodes that are sending data concurrently using channel \( c \), the following must be satisfied:

\[
\frac{P \cdot 1^{-\alpha}}{N + \sum_{v_k \in S_c \setminus \{v_i\}} P \cdot d(v_k, v_j)^{-\alpha}} \geq \beta
\]

where \( S_c \) denotes the set of sender nodes that are assigned the channel \( c \). The inequality (5.2) implies

\[
\frac{\beta \sum_{v_k \in S_c \setminus \{v_i\}} \frac{1}{d(v_k, v_j)^{\alpha}}}{(1-\delta^\alpha)} \leq \frac{\beta \int_x^\infty 2\pi y \cdot y^{-\alpha} dy}{(1-\delta^\alpha)} \leq 1
\]

where \( x \) is the shortest distance between the receiver \( v_j \) (of a sender \( v_i \) that is assigned channel \( c \)) and one of the other senders in \( S_c \setminus \{v_i\} \). From inequality (5.3), we get \( x \geq \left( \frac{\beta \cdot 2\pi}{(1-\delta^\alpha)(\alpha-2)} \right)^{\frac{1}{\alpha-2}} \).
Thus, \((\frac{\beta \cdot 2\pi}{(1-\delta^\alpha)(\alpha-2)})^{\frac{1}{\alpha-2}}\) is a lower bound for \(x\), and we can set \(H_2 = \lceil ((\frac{\beta \cdot 2\pi}{(1-\delta^\alpha)(\alpha-2)})^{\frac{1}{\alpha-2}} + 1 \rceil\) as the shortest hop distance between two senders that can be assigned the same channel on \(G_2\).

Algorithm 12 computes a channel assignment \(\Pi\) for a set \(V\) of nodes on \(G_2\). It assigns each channel \(c \in \{0, 1, ..., (H_2 - 1)^2\}\) to the sender nodes in the set \(\pi_c\). Each of such sender nodes is at least \(H_2\) hops away from each other on \(G_2\), and it uses the assigned channel \(c\) to send its data to its receivers \(v_{i-1,j}, v_{i,j-1}, v_{i,j+1}\) and \(v_{i+1,j+1}\).

**Algorithm 12 2-Dimensional Grid**

**Input:** A set \(V\) of nodes on the 2-dimensional grid  
**Output:** A Channel Assignment  
1: for \(v_{i,j} \in V\) do  
2: \(c \leftarrow (i \mod H_2) + ((j \mod H_2) \cdot H_2)\)  
3: \(\pi_c \leftarrow \pi_c \cup \{(v_{i,j}, P)\}\)  
4: end for  
5: return \(\Pi = \{\pi_0, \pi_1, ..., \pi_{(H_2-1)^2}\}\)

**Theorem 5.5.1.** Algorithm 12 produces a successful channel assignment \(\Pi\) whose size is \(O(1)\).

**Proof.** For all \(j = 0, 1, ...,\), let \(V_j = \{v_{i,j}|i = 0, 1, ...\}\), and \(Q_k = \{v_{i,j} \in V_j|(j \mod H_2) = k\}\). Then, for each \(k \in \{0, 1, ..., (H_2 - 1)\}\), Algorithm 12 assigns \(H_2\) channels to the nodes in \(Q_k\). That is, it assigns channels 0, 1, ..., \((H_2 - 1)\) to the nodes in \(Q_0\), and assigns channels \(H_2 + 1, H_2 + 2, ..., 2H_2 - 1\) to the nodes in \(Q_1\), and so on. Therefore, Algorithm 12 requires \((H_2)^2 = O(1)\) channels. And, as each node \(v_{i,j}\) uses the assigned channel to send its data to its receivers \(v_{i-1,j}, v_{i,j-1}, v_{i,j+1}\) and \(v_{i+1,j+1}\) without interference using the assigned channel, the channel assignment induces a strongly connected graph. \(\square\)
5.6 Constant-Factor Approximation Algorithms

In this section, we introduce two constant-factor approximation algorithms for the MCA problem, assuming a uniform power level \( P \). As done in Section 5.5, we also consider only the links \((u, v)\), where \( d(u, v) \leq \delta(\frac{P}{\beta N})^{\frac{1}{\alpha}} \), for some constant \( \delta \in (0, 1) \). We further make the assumption that a network represented as the directed graph \( G = (V, E) \), where \( E = \{(u \to v)|d(u, v) \leq \delta(\frac{P}{\beta N})^{\frac{1}{\alpha}}\} \), is strongly connected, and \( \alpha > 2 \) (Gupta and Kumar, 2000).

5.6.1 Tree Construction

We start this section by introducing some standard notations that are used subsequently:

- **Maximal Independent Set (MIS):** A subset \( V' \subseteq V \) of the graph \( G \) is said to be independent if for any vertices \( u, v \in V' \), \((u, v) \notin E \). An independent set is said to be maximal if it is not a proper subset of another independent set.

- **Connected Dominating Set (CDS):** A dominating set (DS) is a subset \( V' \subseteq V \) such that every vertex \( v \) is either in \( V' \) or adjacent to a vertex in \( V' \). A DS is said to be connected if it induces a connected subgraph.

Our algorithms assign channels to nodes based on a channel assignment tree whose construction is based on that of a data aggregation tree in (Li et al., 2009). We pick any node \( v \) as a root node, and construct a breadth-first-search (BFS) tree \( T_{BFS} \) (cf. (Cormen et al., 2009)) on \( G \). Then, as done in (Li et al., 2009), a Maximal Independent Set (MIS) is found using an algorithm in (Wan et al., 2002) based on \( T_{BFS} \). The nodes in the MIS are called dominators, and the others are called dominatees. Here, the MIS constructed by (Wan et al., 2002) guarantees that the distance between any pair of its complementary subsets is exactly two hops. Based on the MIS, (Li et al., 2009) obtains a Connected Dominating Set (CDS) of
$G$ by connecting the dominators using some connectors that were originally dominatees. If there exist dominatees that are not connected to the CDS, then each of such dominatees is connected to its neighboring dominator that has the smallest hop-distance to $v$. We denote this newly formed tree by $T$, and use it as the channel assignment tree in our algorithm.

5.6.2 Algorithms

In this section, we introduce two constant-factor approximation algorithms each of which partitions a network into square cells or hexagons.

Square-Based Channel Assignment

The first algorithm called Square-Based Channel Assignment (SBCA) algorithm (Algorithm 13) is based on the data aggregation algorithm in (Li et al., 2009). SBCA starts by partitioning a network into square cells whose diagonal length is $\delta \left( \frac{P}{\beta N} \right)^{\frac{1}{2}}$. These square cells induce grids where each square cell is labeled with the label $SL(i, j)$ if its upper-left corner has coordinate $(i, j)$.

**Algorithm 13** Square-Based Channel Assignment

**Input:** A set $V$ of nodes

**Output:** A Channel Assignment $\Pi$ and its size

1: Partition the network into square cells each of which has diagonal length $\delta \left( \frac{P}{\beta N} \right)^{\frac{1}{2}}$.
2: Construct a scheduling tree $T$ using the algorithm in (Li et al., 2009).
3: Set the first channel $c \leftarrow 1$
4: Channel assignment $\Pi \leftarrow \emptyset$
5: $V_a \leftarrow$ the set of dominatees
6: $V_b \leftarrow$ the set of dominators
7: $V_c \leftarrow$ the set of connectors
8: $(\Pi, c) \leftarrow$ Assignment-Square($\Pi, V_a, c$)
9: $(\Pi, c) \leftarrow$ Assignment-Square($\Pi, V_b, c$)
10: $(\Pi, c) \leftarrow$ Assignment-Square($\Pi, V_c, c$)
11: **return** $\Pi$ and $c - 1$
Algorithm 14 Assignment-Square

**Input:** A Channel assignment \( \Pi \), a set \( Q \) of nodes, and a starting channel \( c \)

**Output:** Updated channel assignment \( \Pi \) and the new channel

1. **while** \( \exists \) an unmarked node in \( Q \) **do**
2. 
3. 
4. 
5. 
6. 
7. 
8. 
9. 
10. 
11. 
12. 
13. 
14. **return** \( (\Pi, c) \)

Once the channel assignment tree \( T \) is obtained as in Section 5.6.1, SBCA assigns channels based on \( T \). Assigning channels is also based on the constant value \( C \) that guarantees that any two senders can be assigned the same channel if they are \( C \) cells apart from each other.

In the SBCA algorithm, the channel assignment procedure is divided into three phases (Steps 5 – 10 in Algorithm 13):

- **Phase 1:** Assigning channels to dominatees to communicate with their dominators.

- **Phase 2:** Assigning channels to dominators to communicate with their dominatees and connectors.

- **Phase 3:** Assigning channels to connectors to communicate with their dominators.

At each Phase, SBCA (Algorithm 13) uses Algorithm 14 as a subroutine which assigns the same channel to nodes if they are \( C \) cells apart from each other (Steps 3 – 12 in Algorithm 14).
**Hexagon-Based Channel Assignment**

We start this section by introducing a new hexagonal tessellation technique used in our algorithm. First, a network is tessellated with hexagons and the hexagons are labeled using \((3l^2 - 3l + 1)\)-labeling, where \(l\) is a positive integer. Figure 5.2(a) shows an 1-labeling, and by enclosing the 1-labeling with a layer of hexagons we get a 7-labeling as shown in Figure 5.2(b). Enclosing one more layer gives a 19-labeling as shown in 5.2(c), and recursively we get in general \((3l^2 - 3l + 1)\)-labeling. Figure 5.2(d) shows an example of a hexagonal tessellation on a network using 19-labeling.

![Hexagon-Based Channel Assignment](image)

The second algorithm named the **Hexagon-Based Channel Assignment (HBCA)** algorithm (Algorithm 15) starts by partitioning the network into hexagons each of which has diagonal length \(\delta(\frac{P}{N\beta})^{\frac{1}{2}}\), and the hexagons are labeled by \(L\)-labeling, where \(L = 3l^2 - 3l + 1\), \(l = \lceil \frac{1}{6}[(\frac{4\tau}{\delta} \cdot (\frac{\beta N}{P})^\frac{1}{2} + 4)^2 - 3]^\frac{1}{2} + \frac{7}{6} \rceil\) and \(\tau = (\frac{P2\pi}{N(\delta - \alpha - 1)(\alpha - 2)})^{\frac{1}{2-\alpha}}\). Then it constructs a channel assignment tree based on which the nodes are assigned channels.

Once the channel assignment tree \(T\) is obtained as in Section 5.6.1, HBCA assigns channels based on \(T\). Assigning channels is also based on the labels of hexagons in the \(L\)-labeling. The constant value \(L\) guarantees that any two senders can be assigned the same channel if they are located in the hexagons with the same label.

Similar to SBCA, the channel assignment procedure of HBCA is divided into three phases (Steps 5 – 10 in Algorithm 15) and at each Phase, HBCA uses Algorithm 16 as a subroutine.
Algorithm 15 Hexagon-Based Channel Assignment

**Input:** A set $V$ of nodes

**Output:** A Channel Assignment $\Pi$ and its size

1: Partition the network into hexagons each of which has diagonal length $\delta (\frac{P}{\alpha N})^{\frac{1}{3}}$, and label the hexagons using $L$-labeling.
2: Construct a scheduling tree $T$ using the algorithm in (Li et al., 2009).
3: Set the first channel $c \leftarrow 1$
4: Channel assignment $\Pi \leftarrow \emptyset$
5: $V_a \leftarrow$ the set of dominatees
6: $V_b \leftarrow$ the set of dominators
7: $V_c \leftarrow$ the set of connectors
8: $(\Pi, c) \leftarrow$ Assignment-Hexagon$(\Pi, V_a, c)$
9: $(\Pi, c) \leftarrow$ Assignment-Hexagon$(\Pi, V_b, c)$
10: $(\Pi, c) \leftarrow$ Assignment-Hexagon$(\Pi, V_c, c)$
11: return $\Pi$ and $c - 1$

Algorithm 16 Assignment-Hexagon

**Input:** A Channel assignment $\Pi$, a set $Q$ of nodes, and a starting channel $c$

**Output:** Updated channel assignment $\Pi$ and the next channel

1: while $\exists$ an unmarked node in $Q$
2: Pick one unmarked node $v \in Q$ in each hexagon. Let $Q' \subseteq Q$ be the set of such nodes.
3: for $i = 1$ to $L$
4: $X_i \leftarrow \emptyset$, $X_i \leftarrow \{v|v \in Q'$ whose $HL$ is $i$\}
5: if $X_i \neq \emptyset$
6: for each $v \in X_i$
7: $\pi_c \leftarrow \pi_c \cup \{(v, P)\}$, $\Pi \leftarrow \Pi \cup \{\pi_c\}$, Mark $v$.
8: end for
9: $c \leftarrow c + 1$
10: end if
11: end for
12: end while
13: return $(\Pi, c)$

Algorithm 16 assigns the same channel to nodes if they are located in the hexagons with the same label (Steps 3 – 12 in Algorithm 16).
5.6.3 Analysis

In this section, we analyze the Square-Based Channel Assignment (SBCA) and the Hexagon-Based Channel Assignment (HBCA) algorithms (Algorithms 13 and 15).

Analysis of SBCA

First, we analyze the SBCA algorithm, and bound the size of the channel assignment produced by it. We first prove that any two senders can be assigned the same channel if they are $C$ square cells apart.

**Lemma 5.6.1.** For SINR threshold $\beta \geq 1$, path loss exponent $\alpha > 2$, background noise $N > 0$, and some constant $\delta \in (0, 1)$, let

$$C = \left\lceil \left( \frac{P \cdot 2\pi}{N (\delta^{-\alpha} - 1)(\alpha - 2)} \right)^{\frac{1}{\alpha - 2}} \cdot \sqrt{2} \cdot \delta^{-1} \left( \frac{N \beta}{P} \right)^{\frac{1}{\alpha}} + 2 \right\rceil$$

Then any two sender nodes that are at least $C$ square cells away from each other can be assigned the same channel.

**Proof.** Consider a sender node $v_i$ trying to communicate with its farthest possible receiver $v_j$, i.e., $d(v_i, v_j) = \delta \left( \frac{P}{N \beta} \right)^{\frac{1}{\alpha}}$, using the assigned channel $c$. In order that the receiver $v_j$ receives data from the sender $v_i$ without interference, for all other sender nodes that use the same channel $c$ the following must be satisfied:

$$\frac{P \left( \delta \left( \frac{P}{N \beta} \right)^{\frac{1}{\alpha}} \right)^{-\alpha}}{N + \sum_{v_k \in \{v_i, v_j\}} P \cdot d(v_k, v_j)^{-\alpha}} \geq \beta$$

which implies

$$\frac{P \sum_{v_k \in \{v_i, v_j\}} d(v_k, v_j)^{-\alpha}}{N(\delta^{-\alpha} - 1)} \leq \frac{P \cdot 2\pi \int_{x}^{\infty} y^{1-\alpha} dy}{N(\delta^{-\alpha} - 1)} = \frac{P \cdot 2\pi \cdot x^{2-\alpha}}{N(\delta^{-\alpha} - 1)(\alpha - 2)} \leq 1$$  \hspace{1cm} (5.4)
where $x$ is the shortest distance between $v_j$ and one of the other senders that uses the same channel $c$. From inequality (5.5), we get $x \geq \left(\frac{P_{c} \cdot 2 \pi}{N(\delta - \alpha - 1)(\alpha - 2)}\right)^{\frac{1}{\alpha - 2}}$. Then $\left(\frac{P_{c} \cdot 2 \pi}{N(\delta - \alpha - 1)(\alpha - 2)}\right)^{\frac{1}{\alpha - 2}}$ is a lower bound for $x$.

Next, let us bound the number of square cells between $v_j$ and a closest sender to $v_j$ that also uses channel $c$, say $v'_i$. Letting $z$ be the number of square cells between nodes $v'_i$ and $v_j$, we assume that the relay nodes are on a straight line between these two nodes. Then, $\left(\frac{P_{c} \cdot 2 \pi}{N(\delta - \alpha - 1)(\alpha - 2)}\right)^{\frac{1}{\alpha - 2}} \leq \frac{z}{\sqrt{2}} \cdot \delta \left(\frac{P_{c}}{N^\beta}\right)^{\frac{1}{\alpha}}$ which implies $\left(\frac{P_{c} \cdot 2 \pi}{N(\delta - \alpha - 1)(\alpha - 2)}\right)^{\frac{1}{\alpha - 2}} \cdot \delta^{-1} \cdot \sqrt{2} \cdot \left(\frac{P_{c}}{N^\beta}\right)^{-\frac{1}{\alpha}} \leq z$. Therefore, $v_j$ and $v'_i$ should be at least $\left[\left(\frac{P_{c} \cdot 2 \pi}{N(\delta - \alpha - 1)(\alpha - 2)}\right)^{\frac{1}{\alpha - 2}} \cdot \frac{\sqrt{2} \cdot \delta \left(\frac{P_{c}}{N^\beta}\right)^{\frac{1}{\alpha}}}{\delta}\right] \geq \left[\left(\frac{P_{c} \cdot 2 \pi}{N(\delta - \alpha - 1)(\alpha - 2)}\right)^{\frac{1}{\alpha - 2}} \cdot \sqrt{2} \cdot \left(\frac{P_{c}}{N^\beta}\right)^{-\frac{1}{\alpha}}\right]$ cells apart, and any two senders should be at least $C$ cells apart. 

**Lemma 5.6.2.** (Li et al., 2009) The number of connectors in a square cell is at most 12.

**Lemma 5.6.3.** (Li et al., 2009) For any node, at most $\omega = r^\alpha - 1$ neighboring nodes can be assigned the same channel where $r = \delta \left(\frac{P_{c}}{N^\beta}\right)^{\frac{1}{\alpha}}$.

**Corollary 5.6.1** (Lower Bound). In order to produce a successful channel assignment, any channel assignment algorithm requires $\geq \frac{\Delta}{\omega}$ channels.

**Proof.** Considering a node $v$ whose degree is $\Delta$ and its neighboring nodes, for strong connectivity, all nodes need to be a sender at least once, i.e., all nodes need to be assigned at least one channel as a sender. By Lemma 5.6.3, at one channel at most $\omega$ nodes can be assigned, and therefore any algorithm requires at least $\frac{\Delta}{\omega}$ channels, i.e., the size of the channel assignment produced is at least $\frac{\Delta}{\omega}$. 

**Theorem 5.6.1.** The SBCA algorithm produces a successful channel assignment whose size is bounded by $\Delta \cdot C^2 + 13 \cdot C^2 = O(\Delta)$, and it is therefore a constant-factor approximation algorithm.

**Proof.** Let us consider the Phase 1. There exist at most $\Delta$ dominatees in each square cell. Thus, at most $\Delta \cdot C^2$ channels are used by dominatees to communicate with their dominators.
For the Phase 2, as there exists at most 1 dominator in each square cell, at most $C^2$ channels are used by dominators to communicate with their connectors and dominatees. Lastly, the Phase 3 needs at most $12 \cdot C^2$ channels for connectors to communicate with their dominators, as there exist at most 12 connectors in each square cell (Lemma 5.6.2). Thus, the channel assignment $\Pi$ produced by SBCA induces a strongly connected graph, and its size is bounded by $\Delta \cdot C^2 + 13 \cdot C^2 = O(\Delta)$. By Lemma 5.6.3 and Corollary 5.6.1, SBCA is a constant-factor approximation.

5.6.4 Simulation Results

In this section, we compare the performance of the Square-Based Channel Assignment (SBCA) and the Hexagon-Based Channel Assignment (HBCA) algorithms on the same networks through simulation. In our simulation, the networks are randomly generated on the Euclidean plane. We randomly deploy 1000 nodes on an area of size $m \times m$, where $m = 100, 200, 300, 400$ and 500. For each $m$, we generate 50 different networks, and average the sizes of the channel assignments produced by the algorithms over the networks. For the simulation, $\delta$ and the initial power $P$ of nodes are set as follows:

- **Choice of $\delta$:** We use $\delta = \{0.1, 0.12, ..., 0.96, 0.98\}$.
- **Initial Power Assignment:** We first use Kruskal’s algorithm (Cormen et al., 2009) to find the minimum spanning tree $T_{MST}$ using edge weights defined as the distance between any two nodes. Given a uniform power, if $\delta$ is too small, then the graph may not be connected. In order to make the initial graph connected even with the smallest $\delta = 0.1$ in our simulation, we set the initial power $P = \beta N \left(\frac{d}{\sqrt{1}}\right)^{\frac{1}{\alpha}}$, where $d$ is the distance of the longest edge on $T_{MST}$. Given the initial power assignment, we get the initial graph $G = (V, E)$, where $E = \{(u \to v) | d(u, v) \leq \delta \left(\frac{P}{\sqrt{N}}\right)^{\frac{1}{\alpha}}\}$.
Table 5.1. The size of channel assignments produced by SBCA and HBCA algorithms

<table>
<thead>
<tr>
<th>$m$</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>400</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta$</td>
<td>SBCA</td>
<td>HBCA</td>
<td>SBCA</td>
<td>HBCA</td>
<td>SBCA</td>
</tr>
<tr>
<td>0.10</td>
<td>332.40</td>
<td>406.05</td>
<td>476.20</td>
<td>574.35</td>
<td>591.00</td>
</tr>
<tr>
<td>0.12</td>
<td>452.85</td>
<td>566.50</td>
<td>641.40</td>
<td>696.95</td>
<td>781.20</td>
</tr>
<tr>
<td>0.14</td>
<td>606.15</td>
<td>687.70</td>
<td>834.70</td>
<td>872.60</td>
<td>936.30</td>
</tr>
<tr>
<td>0.16</td>
<td>809.45</td>
<td>873.65</td>
<td>961.10</td>
<td>954.90</td>
<td>996.95</td>
</tr>
<tr>
<td>0.18</td>
<td>908.50</td>
<td>995.10</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>0.20</td>
<td>981.40</td>
<td>995.10</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>0.22</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>0.24 ~ 0.98</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
</tbody>
</table>

Table 5.1 shows the performance of the SBCA and HBCA algorithms with $\alpha = 5$, $N = 0.01$ and $\beta = 1$. As seen in the table, for fixed $\delta$, as $m$ becomes smaller (i.e., the networks become denser) the size of channel assignments decreases, whereas as $m$ becomes larger (i.e., the networks become sparser) the size increases. For fixed $m$, as $\delta$ becomes smaller (i.e., the network node degree becomes smaller) the size of channel assignments decreases, and as $\delta$ becomes larger (i.e., the network node degree becomes larger) the size increases. However, at some point both algorithms always give the worst performance (i.e., the size is 1000) because the networks become nearly complete graphs in our simulation. It can also be observed that SBCA outperforms HBCA in most cases.

5.7 Conclusion

In this chapter, we have studied the Minimum Channel Assignment (MCA) problem concerning strong connectivity in the SINR model. We have proved the NP-hardness of the problem, and proposed an algorithm yielding a channel assignment whose size is $O(1)$ on 2-dimensional networks. We also proposed two constant-factor approximation algorithms that produce channel assignments whose size is bounded by $O(\Delta)$, and studied the performance of the algorithms through simulation. As to future work, we plan to study approximation
algorithms for the MCA problem with power control assuming arbitrary power levels in the
SINR model.
CHAPTER 6
BOUNDDED-DEGREE MINIMUM-RADIUS SPANNING TREES IN
WIRELESS SENSOR NETWORKS

6.1 Abstract

In this chapter, we study the problem of computing a spanning tree of a given undirected disk graph such that the radius of the tree is minimized subject to a given degree constraint $\Delta^*$. We first introduce a $(8, 4)$-bicriteria approximation algorithm for unit disk graphs (which is a special case of disk graphs) that computes a spanning tree such that the degree of any nodes in the tree is at most $\Delta^* + 8$ and its radius is at most $4 \cdot OPT$, where $OPT$ is the minimum possible radius of any spanning tree with degree bound $\Delta^*$. We also introduce an $(\alpha, 2)$-bicriteria approximation algorithm for disk graphs that computes a spanning tree whose maximum node degree is at most $\Delta^* + \alpha$ and whose radius is bounded by $2 \cdot OPT$, where $\alpha$ is a non-constant value that depends on $M$ and $k$ with $M$ being the number of distinct disk radii and $k$ being the ratio of the largest and the smallest disk radius.

6.2 Introduction

Given an undirected graph, the problem of computing a spanning tree which satisfies certain constraints such as node degree, diameter (or radius) or total cost has been the focus of many researchers as the tree can serve as the pre-defined network infrastructure in wireless sensor networks (WSNs) and other applications. Such problems are known as constrained spanning tree problems in the literature.

The Minimum-Degree Spanning Tree (MDeST) problem is one of the constrained spanning tree problems, and its objective is to compute a spanning tree such that the maximum
node degree of the tree is minimized. The NP-completeness of the problem has been proved in (Garey and Johnson, 1979) for general graphs whereas (Fürer and Raghavachari, 1992) showed an iterative polynomial time approximation algorithm, which is currently the best, that computes a spanning tree whose maximum node degree is at most $\Delta^* + 1$, where $\Delta^*$ is the degree of the optimal tree.

A problem related to the MDeST problem is the Degree-Bounded Minimum-Cost Spanning Tree (DBMCST) problem. Given a nonnegative cost function on the edges of a graph and a degree constraint $\Delta^*$, its objective is to compute a minimum cost spanning tree with the maximum node degree $\Delta^*$. (Könemann and Ravi, 2002) proposed a bicriteria approximation algorithm that computes a spanning tree with maximum node degree $O(\Delta^* + \log |V|)$ and cost $O(\text{opt}_c)$, where $V$ is the set of nodes of the given graph and $\text{opt}_c$ is the minimum cost of any spanning tree whose maximum node degree is at most $\Delta^*$. Later, (Goemans, 2006) introduced an improved bicriteria approximation algorithm that computes a spanning tree with maximum node degree $\Delta^* + 2$ and cost $\text{opt}_c$. It has again been improved by a bicriteria approximation algorithm in (Singh and Lau, 2007) that yields a spanning tree with maximum node degree $\Delta^* + 1$ and cost $\text{opt}_c$.

Besides bounding the node degree, the problem of computing a spanning tree with bounded diameter has also been investigated by several researchers. The objective of the Minimum-Diameter Spanning Tree (MDiST) problem is to compute a spanning tree such that the diameter of the tree is minimized. (Ho et al., 1991) described an $O(|V|^3)$ algorithm that solves the MDiST problem in the polynomial time. Later, (Hassin and Tamir, 1995) showed that the MDiST problem on general graphs is reducible to the absolute 1-center problem (Hakimi, 1964) which can be solved in $O(|V||E| + |V|^2 \log |V|)$ time (Kariv and Hakimi, 1979), where $E$ is the set of edges of the given graph.

While these works concerned only the node degree constraint or the diameter constraint, recently (Ghosh et al., 2011) has focused on bounding both the node degree and radius.
The Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem is to construct a spanning tree such that the radius of the tree is minimized subject to a given degree constraint $\Delta^*$. (Ghosh et al., 2011) introduced a $(10, 7)$-bicriteria approximation algorithm for the BDMRST problem for UDGs that computes a spanning tree whose maximum node degree is bounded by $\Delta^* + 10$ and whose diameter is at most $7 \cdot OPT$, where $OPT$ is the minimum possible radius for the given degree bound $\Delta^*$.

A problem closely related to BDMRST is the problem of computing a spanning tree of a given undirected edge-weighted unit disk graph such that the radius of the tree is minimized subject to a given degree constraint $\Delta^*$. The edges of the given graph are endowed with a metric length function. (Ravi, 1994) introduced an $(O(\log^2 |V|), O(\log |V|))$-bicriteria approximation algorithm that computes a spanning tree whose maximum node degree is bounded by $O(\Delta^* \cdot \log^2 |V|)$ and whose diameter is bounded by $O(\log |V| \cdot OPT')$, where $OPT'$ is the minimum possible diameter for the given degree bound $\Delta^*$. Later, (Könemann et al., 2004) proposed a $(1, \sqrt{\log \Delta^*} |V|)$-bicriteria approximation algorithm that computes a spanning tree whose maximum node degree is bounded by $\Delta^*$ and whose diameter is bounded by $O(\sqrt{\log \Delta^*} |V| \cdot OPT')$. The algorithm in (Könemann et al., 2004), however, works only for undirected complete unit disk graphs, and $\Delta^* \geq 3$.

In this chapter, we continue the study of the Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem on disk graphs (DGs) as well as unit disk graphs (UDGs). Note that the sole existing work (Ghosh et al., 2011) studied the problem for unit disk graphs (UDGs) only. We first introduce a $(8, 4)$-bicriteria approximation algorithm that constructs a spanning tree for a UDG whose maximum node degree is at most $\Delta^* + 8$ and whose radius is bounded by $4 \cdot OPT$. This is an improvement over the bound $(10, 7)$ obtained in (Ghosh et al., 2011). We also propose a simple $(\alpha, 2)$-bicriteria approximation algorithm that computes a spanning tree for a DG where the maximum node degree is at most $\Delta^* + \alpha$ and whose radius is bounded by $2 \cdot OPT$, where $\alpha$ is a non-constant value that depends on
$M$ and $k$ with $M$ being the number of distinct disk radii and $k$ being the ratio of the largest and the smallest disk radius. To the best of our knowledge, the latter is the first result for DGs.

This chapter is organized as follows. Section 6.3 describes our network model and defines the Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem. Section 6.4 introduces a bicriteria approximation algorithm for the BDMRST problem for unit disk graphs (UDGs), whereas Section 6.5 deals with a bicriteria approximation algorithm for disk graphs (DGs).

6.3 Preliminaries

6.3.1 Network Model

A wireless sensor network (WSN) consists of a set $V$ of sensor nodes, and each node $u \in V$ is assigned a transmission power level $p(u)$. The signal sent by a node $u$ can be received by another node $v$ if the distance between $u$ and $v$, denoted by $d(u,v)$, is $\leq p(u)$. We consider the bidirectional case where two nodes $u$ and $v$ can communicate via a communication edge which exists between nodes $u$ and $v$, if and only if $d(u,v) \leq p(u)$ and $d(v,u) \leq p(v)$. In this chapter, the communication graph is modeled as an undirected graph $G(V,E)$, where $E = \{(u,v) \mid u,v \in V, d(u,v) \leq p(u) \text{ and } d(v,u) \leq p(v)\}$. In the literature, a communication graph is called a disk graph (DG) if all nodes are assigned various power levels, and if all nodes are assigned the same power level, it is called a unit disk graph (UDG).

We now introduce some definitions and notations that will be used subsequently:

- **Graph Center**: Given a communication graph $G = (V,E)$, we call a node $c$ a center node if the distance from $c$ to the farthest node from $c$ is minimum.

- **Maximal Independent Set (MIS)**: A subset $V' \subseteq V$ of the graph $G$ is said to be independent if for any vertices $u,v \in V', (u,v) \notin E$. An independent set is said to be maximal if it is not a proper subset of another independent set.
• **Connected Dominating Set (CDS):** A dominating set (DS) is a subset \( V' \subseteq V \) such that every vertex \( v \) is either in \( V' \) or adjacent to a vertex in \( V' \). A DS is said to be *connected* if it induces a connected subgraph.

### 6.3.2 Problem Definition

Given a disk graph \( G = (V, E) \) and an integer constant \( \Delta^* \geq 2 \), the objective of the Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem is to construct a bounded-degree minimum-radius spanning tree \( T = (V, E_T \subseteq E) \) of \( G \) such that the radius of \( T \) is minimized while the degree of any node in \( T \) is at most \( \Delta^* \). Formally, BDMRST is defined as follows:

**Input.** A disk graph \( G = (V, E) \), where \( V \) is a set of nodes and \( E \) is a set of communication edges, and a degree constraint \( \Delta^* \geq 2 \).

**Output.** A minimum-radius spanning tree \( T = (V, E_T \subseteq E) \) whose degree is bounded by \( \Delta^* \).

In this chapter, we focus on designing an \((\alpha, \beta)\)-bicriteria approximation for the BDMRST problem. An \((\alpha, \beta)\)-bicriteria approximation algorithm provides a spanning tree for any instance of BDMRST satisfying the following two conditions as in (Ghosh et al., 2011):

1. The degree of any nodes in the spanning tree is at most \( \alpha + \Delta^* \).

2. The radius of the spanning tree is at most \( \beta \cdot \text{OPT} \), where \( \text{OPT} \) is the minimum possible radius of any spanning tree whose degree is bounded by \( \Delta^* \).

### 6.3.3 NP-Completeness of the BDMRST Problem

First note that (Itai et al., 1982) showed that the Hamilton Path problem on grid graphs is NP-complete. Now, observe that a special case of the BDMRST problem is to compute
Hamiltonian path on UDGs, which is clearly NP-complete (Itai et al., 1982) since grid graphs are a special case of UDGs. Thus, we have

**Theorem 6.3.1.** The Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem is NP-complete.

### 6.4 Bounded-Degree Minimum-Radius Spanning Tree on Unit Disk Graphs

In this section, we introduce a (8,4)-bicriteria approximation algorithm for the Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem for unit disk graphs. Given a UDG $G = (V, E)$ where all nodes are assigned the same power level $r$, we assume without loss of generality that $G$ is connected.

#### 6.4.1 Algorithm

Our algorithm, BDMRST-UDG (Algorithm 17), starts with an initial tree which is the data aggregation tree $T_I$, that is a spanning tree of $G$, computed in (Li et al., 2009) as follows. A center node $c$ of $G$ is picked as the root node, and a breadth-first-search (BFS) tree $T_{BFS}$ (cf. (Cormen et al., 2009)) on $G$ is constructed. We then ((Li et al., 2009)) find a Maximal Independent Set (MIS) of $G$ using an algorithm in (Wan et al., 2002) starting with $T_{BFS}$. The nodes in the MIS are called dominators, and the others are called dominatees. Here, the MIS constructed by (Wan et al., 2002) guarantees that the shortest hop-distance between any two complementary pairs, $A$ and $MIS \setminus A$, where $A \subseteq MIS$, is exactly two hops. For example, in Figure 6.1, consider one pair $A = \{v_1, v_2\}$ and $B = MIS \setminus A = \{v_3, v_4, v_5, c\}$. The shortest hop-distance between the sets $A$ and $B$ on $G$ is exactly two hops.

Next, based on the MIS, we obtain a Connected Dominating Set (CDS) of $G$ by connecting the dominators using some connectors that were originally dominatees. If there exist dominatees that are not connected to the CDS, then each of such dominatees is connected to
its neighboring dominator that has the smallest hop-distance to $c$. We use the newly formed tree denoted by $T_I = (V, E_T \subseteq E)$ as an initial tree. Figure 6.2 shows an example for the construction of the CDS, and Figure 6.3 shows an example for the construction of an initial tree $T_I$. Once $T_I$ is obtained we compute a backbone tree which is a subtree $T_B \subseteq T_I$ that consists of the dominators and their connectors, and the tree edges of $T_I$ connecting them.

Our algorithm is divided into three phases:

- **Phase 1 (Removing Redundant Connectors):** We remove redundant connectors on the initial backbone tree $T_B$, and obtain the modified backbone tree $T'_B$. 

Figure 6.3. Example of the construction of an initial tree $T_I$ whose edges are represented by bold lines. Black nodes represent dominators, gray nodes represent connectors, and white nodes represent dominatees.

- **Phase 2 (Modifying Backbone Tree):** We modify the backbone tree $T'_B$ by reconnecting the connectors, and obtain the modified backbone tree $T''_B$.

- **Phase 3 (Constructing Local Trees):** For the dominatees, we build several local trees by reconnecting the dominatees, and obtain the final bounded-degree minimum-radius spanning tree $T_F$.

**Phase 1.** Once $T_I$ is obtained, we remove redundant connectors so that the number connectors of a dominator is minimized. Denoting the level (depth) of a node $u$ in a tree $T$ by $\ell_T(u)$, let us consider a dominator $u$ with $\ell_{T_B}(u) = \ell$ and its 2-hop-away dominators $v$ and $w$ with $\ell_{T_B}(v) = \ell_{T_B}(w) = \ell + 2$. Let $v'$ and $w'$ with $\ell_{T_B}(v') = \ell_{T_B}(w') = \ell + 1$ be the corresponding connectors in $T_B$ for $v$ and $w$, respectively. There may exist the case that $v$ and $w$ can be shared by one of the connectors, $v'$ and $w'$ (See Lemma 6.4.1). If that happens, we remove one of the connectors, and therefore three dominators, $u$, $v$ and $w$, are connected via only one connector. This phase is denoted by RRC and its details are contained in Algorithm 18. Figure 6.4 shows an example of RRC.

**Phase 2.** For each dominator $u$, we partition its broadcasting disk into 8 sectors each of which has an angle of $\frac{2\pi}{8}$ radian (See Figure 6.5). The $i$th sector of $u$ is denoted by $sec_i$,
Algorithm 17  Bounded-degree Minimum-radius Spanning Tree of a UDG (BDMRST-UDG)

**Input:** A UDG $G = (V, E)$, and a degree constraint $\Delta^* \geq 2$

**Output:** Minimum-radius tree $T_F$ whose node degree is bounded by $\Delta^* + 8$

1: $T_I \leftarrow$ data aggregation tree of $G$ as constructed in (Li et al., 2009)
2: $T_B = (V_B, E_B) \leftarrow$ the set $V_B \subseteq V$ of dominators and their connectors, and the tree edges $E_B$ of $T_I$ connecting them
3: $T'_B \leftarrow$ RRC($T_B$) // Phase 1
4: for each dominator $u \in V$ do
5: Partition $u$’s broadcasting disk into 8 sectors each of which has an angle of $\frac{2\pi}{8}$ radian.
6: end for
7: $T''_B \leftarrow$ BTC($T'_B$) // Phase 2
8: $T_L = (V_L, E_L) \leftarrow$ the set $V_L \subseteq V$ of dominators and their dominatees, and the tree edges $E_L$ of $T_I$ connecting them
9: $T_F \leftarrow$ LCT-UDG(($T''_B \cup T_L$), $\Delta^*$) // Phase 3
10: return Bounded-degree Minimum Radius Spanning Tree $T_F$

**Figure 6.4.** Example of RRC (Phase 1). Black nodes represent dominators, gray nodes represent connectors, and white nodes represent dominatees.

**Figure 6.5.** Partitioning $u$’s broadcasting disk into 8 sectors.

$1 \leq i \leq 8$. We note that the connectors of $u$ in each $sec_i$ form a complete graph. Now, for each $sec_i$, letting $C_u$ be the set of $u$’s connectors in $T'_B$, with $\ell_{T'_B} = \ell_{T_B}(u) + 1$, we choose a node $v' \in C_u$ as a local sink such that $d(v', u)$ is smallest. As the next step, we build a local
Algorithm 18 Removing Redundant Connectors (RRC)

**Input:** Backbone tree $T_B = (V_B, E_B)$

**Output:** The modified backbone tree $T'_B$

1: for each dominator $u \in V_B$ do
2: $E'_B \leftarrow E_B$
3: $X \leftarrow u$'s 2-hop-away dominators in $T_B$ with $\ell_{T_B} = \ell_{T_B}(u) + 2$
4: $Y \leftarrow u$'s connectors in $T_B$ with $\ell_{T_B} = \ell_{T_B}(u) + 1$
5: for each connector $v' \in Y$ do
6: $Z \leftarrow \{w \mid (v', w) \notin E'_B, w \in X\}$
7: for each dominator $w \in Z$ do
8: Pick $w'$ which is the connector that connects $u$ and $w$.
9: if $d(w, v') \leq r$ then
10: $E'_B \leftarrow E'_B - \{(w, w')\}$, $E'_B \leftarrow E'_B \cup \{(w, v')\}$
11: $Z \leftarrow Z - \{w\}$
12: if $\exists$ no more dominators in $Z$ connected to $u$ via $w'$ then
13: Mark $w'$ as a dominatee.
14: $Y \leftarrow Y - \{w'\}$
15: end if
16: end if
17: end for
18: end for
19: return $T'_B = (V_B, E'_B)$

In Case 1, all remaining connectors are connected to the local sink $v'$, but $v'$ is still connected to its dominator $u$. Then, $v'$ is connected to at most 3 connectors as children in $T''_B$. In Case 2, we divide $C_u$ into two subsets, $C^1_u$ and $C^2_u$, such that $|C^1_u|$ and $|C^2_u|$ are almost equal. (If $|C_u|$ is even, we divide it into the two equal-sized subsets, otherwise one subset’s size is one greater than the other subset’s size.) Without loss of generality, assume that the local sink $v'$ of $sec_i$ is in $C^1_u$. Then, we choose a node $w' \in C^2_u$ arbitrarily as a temporary local sink of
the set $C_u^2$. Next, for each $C_u^1$ and $C_u^2$, the remaining connectors in each set are connected to their local sink. Then, we connect $w'$ to $v'$, but $v'$ is still connected to its dominator $u$. Here, $v'$ is connected to at most 4 connectors as children, and $w'$ is connected to at most 3 connectors as children in $T_B''$. The details are contained in Algorithm 19, and Figure 6.6 shows an example of Phase 2 for the case $|C_u| = 8$.

**Phase 3.** In this phase, we consider only dominatees. For each dominator $u$, the dominatees of $u$ in each sector $sec_i$, $1 \leq i \leq 8$, form a complete graph. Letting $D_u$ be the set of dominatees of $u$ in each $sec_i$, we choose one node $x \in D_u$ such that $d(x, u)$ is smallest. We then construct a local tree $T_x$ rooted at $x$ spanning all nodes in $D_u$ as follows. Let $Y \subseteq D_u$ be the set of nodes that are not connected to $T_x$ yet, and $X \subseteq D_u$ be the set of nodes that are connected to $T_x$ but their degree is 1, where the local sink $x$ is initially an element of $X$. We select up to $\Delta^* - 1$ nodes closest to $x$ from $Y$, if they exist, and connect them to $x$. Then, $x$ is removed from $X$, and the (up to) $\Delta^* - 1$ newly selected nodes are removed from $Y$ and added to $X$. Next, we pick one node $x'$, whose level (that is its depth in the tree $T_x$) is smallest, from $X$, and we select (up to) $\Delta^* - 1$ nodes closest to $x'$ from $Y$, if they exist, and connect them to $x'$. Then, $x'$ is removed from $X$, and the newly connected $\Delta^* - 1$ nodes are also removed from $Y$ and become new elements of $X$. We repeat this process until all nodes in $D_u$ are connected in $T_x$. The details are contained in Algorithm 20.

Next, we consider the following two cases:

- **Case 1:** $|C_u| \geq 1$
- **Case 2:** $|C_u| = 0$

In Case 1, there already exists the local sink $v'$, which is a connector, for $sec_i$ which was chosen in Phase 2. We connect $x$ to $v'$. In Case 2, there exist only dominatees in $sec_i$, and therefore we choose $x$ as the local sink for $sec_i$. In this case, $x$ is still connected to its dominator $u$. The details are contained in Algorithm 20, and Figure 6.7 shows an example of Phase 3 in case of $|C_u| \geq 1$ with $\Delta^* = 4$. 
Algorithm 19 Backbone Tree Construction (BTC)

Input: Backbone Tree $T_B' = (V_B, E_B')$

Output: Modified backbone tree $T_B''$ whose maximum node degree is bounded by 10

1: $E_B'' \leftarrow E_B'$
2: for each dominator $u \in V_B$ do
3:   for each sector $sec_i$ of $u$ do
4:     $C_u \leftarrow$ all connectors of $u$ in $sec_i$ with $\ell_{T_B'} = \ell_{T_B'}(u) + 1$
5:     Choose a local sink $v' \in C_u$ such that $d(v', u)$ is smallest.
6:     if $1 < |C_u| \leq 4$ then
7:       for each connector $w' \in (C_u - \{v'\})$ do
8:         $E_B'' \leftarrow E_B'' - \{(u, w')\}$, $E_B'' \leftarrow E_B'' \cup \{(w', v')\}$
9:     end for
10:   else if $4 < |C_u| \leq 8$ then
11:     if $|C_u| \mod 2 = 0$ then
12:         Divide $C_u$ into two subsets, $C_u^1$ and $C_u^2$, such that $|C_u^1| = |C_u^2| = \frac{|C_u|}{2}$.
13:     else
14:         Divide $C_u$ into two subsets, $C_u^1$ and $C_u^2$, such that $|C_u^1| = |C_u^2| + 1 = \frac{|C_u| + 1}{2}$.
15:     end if
16:     for each $C_u^i \in \{C_u^1, C_u^2\}$ do
17:       if $v' \in C_u^i$ then
18:         for each $x' \in (C_u^i - \{v'\})$ do
19:           $E_B'' \leftarrow E_B'' - \{(x', u)\}$, $E_B'' \leftarrow E_B'' \cup \{(x', v')\}$
20:       end for
21:     else if $v' \notin C_u^i$ then
22:       Choose a node $w' \in C_u^i$ arbitrarily.
23:       for each $y' \in (C_u^i - \{w'\})$ do
24:         $E_B'' \leftarrow E_B'' - \{(y', u)\}$, $E_B'' \leftarrow E_B'' \cup \{(y', w')\}$
25:       end for
26:     end if
27:     end for
28:     $E_B'' \leftarrow E_B'' - \{(w', u)\}$, $E_B'' \leftarrow E_B'' \cup \{(w', v')\}$
29:   end if
30: end for
31: end for
32: return $T_B'' = (V_B, E_B'')$
Algorithm 20 Local Tree Construction (LTC-UDG)

**Input:** Tree \( \hat{T} = (T_B' \cup T_L) \) and degree constraint \( \Delta^* \geq 2 \)

**Output:** Tree \( T_F \)

1. \( E_T \leftarrow \) Tree edges of \( \hat{T} \)
2. Let \( Q \) be a set of dominatees in \( V \).
3. for each dominator \( u \in V \) do
   4. \( E_T \leftarrow E_T - \{(x, y) \mid d(x, y) \leq r \text{ and } x, y \in (Q \cup \{v\})\} \)
   5. for each sector \( sec_i \) of \( u \) do
      6. \( D_u \leftarrow \) all dominatees of \( u \) in \( sec_i \)
      7. \( C_u \leftarrow \) all connectors of \( u \) in \( sec_i \)
      8. if \( |D_u| \geq 1 \) then
         9. Choose one node \( x \in D_u \) such that \( d(x, u) \) is smallest.
      10. /* Steps 10 – 18: Construct a spanning tree \( T_x \) rooted at \( x \) spanning all nodes in \( D_u \) */
      11. \( X \leftarrow \{x\}, Y \leftarrow D_u - \{x\}, V_x \leftarrow \{x\}, E_x \leftarrow \emptyset, T_x \leftarrow (V_x, E_x) \)
      12. while \( X \neq \emptyset \) or \( Y \neq \emptyset \) do
          13. Pick a node \( x' \in X \) whose level (the depth on \( T_x \)) is smallest.
          14. Select \( \Delta^* - 1 \) nodes which are closest to \( x' \) from \( Y \). If \( |Y| < \Delta^* - 1 \), then pick all nodes in \( Y \). Let \( Y' \subseteq Y \) be the set of such selected nodes.
          15. Connect each node \( y' \in Y' \) to \( x' \).
          16. \( E_T \leftarrow E_T \cup \{(y', x')\}, E_x \leftarrow E_x \cup \{(y', x')\} \)
          17. \( Y \leftarrow Y - Y', X \leftarrow X - \{x'\}, X \leftarrow X \cup Y' \)
      18. end while
      19. if \( |C_u| \geq 1 \) then
         20. Let \( v' \in Y \) be the local sink in \( sec_i \).
         21. \( E_T \leftarrow E_T \cup \{(x, v')\} \)
      22. else if \( |C_u| = 0 \) then
         23. Let \( x \) be the local sink in \( sec_i \).
         24. \( E_T \leftarrow E_T \cup \{(x, u)\} \)
      25. end if
   8. end for
5. end for
28. return \( T_F = (V, E_T) \)
6.4.2 Analysis

In this section, we analyze the BDMRST-UDG algorithm (Algorithm 17), and show that it gives an (8, 4)-bicriteria approximation.

Lemma 6.4.1. (Li et al., 2009; Wan et al., 2006) Suppose that dominators $v$ and $w$ are within 2 hops from dominator $u$, and $v'$ and $w'$ are the corresponding connectors for $v$ and $w$, respectively. If $\angle vuw \leq 2 \arcsin \frac{1}{4}$, then either $\overline{wv'} \leq r$ or $\overline{vw'} \leq r$.

Lemma 6.4.2. The number of connectors in one sector is bounded by 8.
Figure 6.8. Example of $\text{sec}(u, \theta, r)$ and $\text{sec}(u, \theta'', 2r)$, where $\theta = \frac{2\pi}{8}$ radian, $\theta' = \frac{\pi}{2}$ radian, and $\theta'' = \theta' + \theta + \theta' = \frac{5\pi}{4}$ radian.

Proof. First, let $\text{sec}(u, \theta, r)$ denote a sector with an angle of $\theta$ radian of a circle (broadcasting disk) centered at a node $u$ with the radius of $r$ (the gray sector in Figure 6.8). Consider a dominator $u$ and let us first bound the number of connectors in a sector $\text{sec}_i(u, \frac{2\pi}{8}, r)$, $1 \leq i \leq 8$, (one sector in Figure 6.5) before removing redundant connectors.

Let $\text{sec}(u, \theta'', 2r)$ be the union of the three sectors, $\text{sec}(u, \frac{\pi}{2}, 2r)$, $\text{sec}_i(u, \frac{2\pi}{8}, 2r)$ and $\text{sec}(u, \frac{\pi}{2}, 2r)$, where $\theta'' = \frac{\pi}{2} + \frac{2\pi}{8} + \frac{\pi}{2} = \frac{5\pi}{4}$ (See Figure 6.8). Since a connector connects a dominator $u$ to other dominators which are 2 hops away from $u$ in the CDS, the connectors in $\text{sec}_i(u, \frac{2\pi}{8}, r)$ can connect $u$ to only dominators which reside in the area of $\text{sec}(u, \theta'', 2r) - \text{sec}(u, \theta'', r)$. Therefore, the number of connectors in $\text{sec}_i(u, \frac{2\pi}{8}, r)$ cannot exceed the number of dominators in the area of $\text{sec}(u, \theta'', 2r) - \text{sec}(u, \theta'', r)$, and it is sufficient to bound the number of such dominators. Observing that $\theta'' < 4$, let us bound the number of dominators in the area of $\text{sec}(u, 4, 2r) - \text{sec}(u, 4, r)$. We partition the area into 16 cells as in Figure 6.9. In Figure 6.9, $\angle p_1p_0p_3 = \angle p_3p_0p_5 = \angle p_5p_0p_7 = 1$ radian, where $p_k$, $0 \leq k \leq 9$, represents a point, and $\angle p_1p_0p_2 = \angle p_3p_0p_4 = \angle p_5p_0p_6 = \angle p_7p_0p_8 = 0.5$ radian. Here, the largest distance in each cell is $\leq r$ (Chen et al., 2007), and therefore there exists at most 1 dominator in each cell. This implies that there exist at most 16 connectors in $\text{sec}_i(u, \frac{2\pi}{8}, r)$.

Next, let us bound the number of connectors after removing redundant connectors. Consider the case that we have the maximum number of connectors, namely 16. Without loss
of generality, let us assume that a connector $v_j$ connects $u$ to a dominator $w_j$ in the cell $c_j$, where $1 \leq j \leq 16$. Considering two dominators $w_q$ and $w_{q+8}$, $1 \leq q \leq 8$, and their corresponding connectors $v_q$ and $v_{q+8}$, as $\angle w_quw_{q+8} \leq 0.5$ radian $\leq (2 \arcsin \frac{1}{4})$ radian, by Lemma 6.4.1, one of the two connectors $v_q$ and $v_{q+8}$ is removed during the removal procedure. Therefore, after removing all redundant connectors, we can have at most 8 connectors remaining in $sec_i(u, \frac{2\pi}{8}, r)$. 

\[ \square \]

**Lemma 6.4.3.** The node degree of a dominator is bounded by 8.

**Proof.** Consider a dominator $u$. It is connected to nodes in its 8 sectors each of which has at most one local sink which is a connector or a dominatee. Therefore, the node degree of a dominator in $T_F$ is bounded by 8. 

\[ \square \]

**Lemma 6.4.4.** The node degree of a connector is bounded by 10.

**Proof.** Consider a dominator $u$ with $\ell_{TB}(u) = \ell$. After Phase 1 (RRC), $u$ is connected to at most 8 connectors with $\ell_{TB}(u) = \ell + 1$ in each sector $sec_i$, $1 \leq i \leq 8$ (Lemma 6.4.2). Let us denote the set of such connectors in $sec_i$ by $C$. After Phase 1, each $v' \in C$ is connected at most 5 connectors (Thai and Du, 2006) in $T'_B$ (1 dominator with $\ell_{TB} = \ell_{T'_B} = \ell$, and the other 4 dominators with $\ell_{TB} = \ell_{T'_B} = \ell + 2$). In Phase 2, at most 4 connectors with
$\ell_{T_B} = \ell_{T'_B} = \ell + 1$ become children of $v'$ in $T''_B$ (See Figure 6.6). In Phase 3, at most 1 dominatee with $\ell_{T_B} = \ell + 1$ becomes a child of $v'$ in $T_F$ (See Figure 6.7). For each sector of every dominator in the final tree $T_F$, we can observe that a connector is connected to at most 10 nodes, and thus the Lemma holds.

**Corollary 6.4.1.** The node degree of any nodes in the backbone tree $T_B$ of $T_F$ is at most 10.

**Lemma 6.4.5.** The radius $R(T_F)$ of $T_F$ is at most $4 \cdot OPT$.

*Proof.* Let us first bound the radius $R(T_B)$ of the backbone tree $T_B$ in $T_F$. $T_B$ is constructed based on the initial tree $T_I$ whose radius is $R$. After Phase 1, $T_B$ is modified to $T'_B$ and its radius is still the same as the radius of $T_I$, namely $R$. It is because, during Phase 1, a dominator, denoted by $v$, with $\ell_{T_B}(v) = \ell + 2$ is reconnected to a dominator, denoted by $u$, with $\ell_{T_B}(u) = \ell$ via the new connector, denoted by $w'$, with $\ell_{T_B}(w') = \ell + 1$ which is the same level as its original connector’s, denoted by $v'$ (See Figure 6.4). After Phase 2, $T'_B$ is transformed into $T''_B$. Observe that during Phase 2, at each level of the tree, a connector’s level is increased by at most 2. Therefore, after Phase 2, the radius of $T'_B$ is increased to at most $3R$ which implies that the radius of $T''_B$ is at most $3R$. Next, let us bound the radius of the local trees. In Phase 3, each of the local trees were constructed from a complete graph, and therefore it is an *optimal tree* in which every node has the node degree $\Delta^*$ (except the leaves and the last parent). Thus the radius of each local tree is $OPT$. Furthermore, noting that $R$ is also the radius of a graph $G$ where the maximum node degree is at most $\Delta$, having such a degree constraint with $2 \leq \Delta^* \leq \Delta$ can only increase the radius of a tree, and hence $R \leq OPT$. Therefore $R(T_F) \leq 3 \cdot R + OPT = 4 \cdot OPT$. □

**Theorem 6.4.1.** Algorithm BDMRST-UDG is an $(8, 4)$-approximation algorithm.

*Proof.* The degree of any nodes in $T_F$ is at most $\max\{\Delta^*, 10\} \leq \Delta^* + 8$ for any $\Delta^* \geq 2$. The radius of $T_F$ is at most $4 \cdot OPT$ by Lemma 6.4.5. Therefore, BDMRST-UDG is an $(8, 4)$-approximation algorithm. □
In BDMRST-UDG (Algorithm 17), Step 1 takes $O(n^2)$ time (Li et al., 2009; Wan et al., 2002). During Step 3 (the sub-procedure of removing redundant connectors, contained in Algorithm 18), the outermost for-loop in Algorithm 18 takes $O(n)$ time, but all inner for-loops in Algorithm 18 take constant time. Therefore, Step 3 takes $O(n)$ time. Steps 4 – 6 simply take $O(n)$ time. In Step 7 (the sub-procedure of constructing the backbone tree, contained in the Algorithm 19), the outermost for-loop in the Algorithm 19 takes $O(n)$ time, but all inner for-loops in the Algorithm 19 take constant times, and therefore the Step 7 takes $O(n)$ time. The last step (Step 9) for constructing the local spanning trees, contained in Algorithm 20, takes $O(n^3)$ time since the outermost for-loop in Algorithm 19 takes $O(n)$ time and the inner while-loop takes $O(n^2)$ time. Thus, BDMRST-UDG has $O(n^3)$ time complexity.

6.5 Bounded-Degree Minimum-Radius Spanning Tree on Disk Graphs

In this section, we introduce a simple $(\alpha, \beta)$-bicriteria approximation algorithms for the Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem on disk graphs. Given a disk graph $G = (V, E)$, let $P$ be the set of power levels \{${r_1, r_2, \ldots, r_M}$\}, i.e., each node $u \in V$ is assigned a power level $p(u) \in P$ by a power assignment function $p : V \rightarrow P$. We assume that $G$ is connected, and $r_1 \leq r_2 \leq \cdots \leq r_M$.

6.5.1 Algorithm

Like the algorithm for unit disk graph in Section 6.4, namely BDMRST-UDG, our new algorithm (Algorithm 21) also starts with an initial tree $T_I = (V, E_T \subseteq E)$ which is a data aggregation tree in (Li et al., 2009). Once $T_I$ is obtained, we also compute a backbone tree which is a subtree $T_B \subseteq T_I$ that consists of the dominators and their connectors, and the tree edges of $T_I$ connecting them.
Figure 6.10. Partitioning $u_m$’s broadcasting disk into $m$ concentric circles (disks) centered at $u_m$.

**Algorithm 21** Bounded-degree Minimum-radius Spanning Tree of a Disk Graph (BDMRST-DG)

**Input:** A disk graph $G = (V, E)$ and a degree constraint $\Delta^* \geq 2$

**Output:** Minimum-radius tree $T_F$ whose node degree is bounded by $\Delta^*$

1: $T_I \leftarrow$ data aggregation tree of $G$ as constructed in (Li et al., 2009)
2: for each dominator $u \in V$ do
3: Partition its broadcasting disk into several pieces as described in Section 6.5.1.
4: end for
5: $T_F \leftarrow$ LTC-DG($T_I, \Delta^*$)
6: return Bounded-degree Minimum-radius Spanning Tree $T_F$

In this algorithm, we focus on constructing local trees by replacing tree edges of $T_I$. Let us denote a node $u$ whose power level $p(u) = r_m$, $1 \leq m \leq M$, by $u_m$. For each dominator $u_m$, we partition its broadcasting disk into 8 sectors each of which has an angle of $\frac{2\pi}{8}$ radian (See Figure 6.5). We again partition the broadcasting disk into $m$ concentric circles (disks) centered at $u_m$. Each of the concentric circles has radius $r_1, \ldots, r_m$. Figure 6.10 shows an example of partitioning the broadcasting disk of $u_4$ into 4 concentric disks with radius $r_1, r_2, r_3$ and $r_4$. Denoting the $i$th sector of $u_m$ by $sec_i$, $1 \leq i \leq 8$, and a concentric disk with radius $r_j$ by $d_j$, $1 \leq j \leq m$, let $a_{i1}$ represent the area of $sec_i \cap d_1$, $a_{i2}$ represent the area of $sec_i \cap (d_2 - d_1)$, and in general, $a_{ij}$ represent the area of $sec_i \cap (d_j - d_{j-1})$. Then, the dominatees of $u_m$ in each area $a_{ij}$, $1 \leq i \leq 8$, $1 \leq j \leq m$, form a complete graph. In each $a_{ij}$, we build a local tree rooted at one of the dominatees as follows. Let $Q_{ij}$ be the set of dominatees located in $a_{ij}$ that are connected to $u_m$. We pick a node $w \in Q_{ij}$ as a local
sink such that \( d(w, u_m) \) is smallest. We then construct a local tree \( T_w \) rooted at \( w \) spanning all nodes in \( Q_{ij} \) as done in Phase 3 in BDMRST-UDG (Algorithm 17). The details are contained in Algorithm 22, and Figure 6.11 shows an example of the construction of local trees.

Figure 6.11. Example of Local Tree Construction (LTC-DG) with \( \Delta^* = 4 \). Black nodes represent dominators, and white nodes represent dominatees. Nodes filled with upward diagonals are dominatees that are selected as local sinks.

6.5.2 Analysis

In this section, we analyze the BDMRST-DG algorithm (Algorithm 21), and show that it gives an \((\alpha, \beta)\)-bicriteria approximation.

Lemma 6.5.1. (Thai and Du, 2006) The degree of each connector in \( T_F \) is bounded by
\[
K = 6(3\lceil \log_2 k \rceil + 2), \text{ where } k = \frac{r_M}{r_1}.
\]

Lemma 6.5.2. Each dominator has at most \( 4\pi M(k + 1) \) connectors.

Proof. Consider a dominator \( u_m \) whose power level is \( r_m, 1 \leq m \leq M \). \( u_m \) is connected to 2-hop-away dominators via its connectors, and therefore it is sufficient to bound the number of such dominators to bound the number of connectors of \( u_m \). Let us bound the number of 2-hop-away dominators whose power level is \( r_i, 1 \leq i \leq M \). Let \( \text{sec}(v, \theta, r) \) denote a
Algorithm 22 Local Tree Construction (LTC-DG)

**Input:** Initial tree $T = (V, E_T)$ and degree constraint $\Delta^* \geq 2$

**Output:** Modified tree $T_F$

1. Let $Q$ be a set of dominatees in $V$.
2. for each dominator $u_m \in V$ do
3. for $i = 1$ to $8$ do
4. $Q_i \leftarrow$ all dominatees of $u_m$ in $sec_i$
5. if $|Q_i| \geq 2$ then
6. for $j = 1$ to $M$ do
7. $Q_{ij} \leftarrow \{v | v \in Q_i$ and $v$ is located in $a_{ij}\}$
8. if $|Q_{ij}| \geq 2$ then
9. Choose one node $w \in Q_{ij}$ such that $d(w, u_m)$ is smallest.
10. for each node $w' \in (Q_{ij} - \{w\})$ do
11. $E_T \leftarrow E_T - \{(w', u_m)\}$
12. end for
13. /* Steps 15 – 22: Construct a spanning tree $T_w$ rooted at $w$ spanning all nodes in $Q_{ij}$ */
14. $X \leftarrow \{w\}$, $Y \leftarrow Q_{ij} - \{w\}$, $V_w \leftarrow \{w\}$, $E_w \leftarrow \emptyset$, $T_w \leftarrow (V_w, E_w)$
15. while $X \neq \emptyset$ or $Y \neq \emptyset$ do
16. Pick a node $x' \in X$ whose level (the depth on $T_w$) is smallest.
17. Select $\Delta^* - 1$ nodes (which are closest to $x'$) from $Y$. If $|Y| < \Delta^* - 1$, then pick all nodes in $Y$. Let $Y' \subseteq Y$ be the set of such selected nodes.
18. Connect each node $y' \in Y'$ to $x'$.
19. $E_T \leftarrow E_T \cup \{(y', x')\}$, $E_w \leftarrow E_w \cup \{(y', x')\}$
20. $Y \leftarrow Y - Y'$, $X \leftarrow X - \{x'\}$, $X \leftarrow X \cup Y'$
21. end while
22. end if
23. end for
24. end if
25. end for
26. end for
27. return $T_F = (V, E_T)$
sector with an angle of $\theta$ radian of a circle (broadcasting disk) centered at a node $v$ with the radius of $r$, and consider a sector $\text{sec}(u_m, \theta, r_m + r_i)$, where $\theta = \frac{r_i}{r_m + r_i}$. We partition the area of $\text{sec}(u_m, \theta, r_m + r_i) - \text{sec}(u_m, \theta, r_m)$ into 2 cells as in Figure 6.12. Here, the largest distance in each cell is $\leq r_i$ (Chen et al., 2007), and hence there exists at most 1 dominator whose level is $r_i$ in each cell. This implies that there exist at most 2 connectors for the dominators in $\text{sec}(u_m, \theta, r_m)$. As $u_m$ has $\frac{2\pi(r_m+r_i)}{r_i}$ sectors, the number of connectors for the dominators whose power level is $r_i$ is $D_i = 2 \cdot \frac{2\pi(r_m+r_i)}{r_i} = 4\pi\left(\frac{r_m}{r_i} + 1\right)$. Note that $D_i$ is largest when $r_m = r_M$, and the 2-hop-away dominators of $u_m$ can be assigned different power levels in $P$. Therefore, the number of connectors of a dominator is bounded by

$$\sum_{i=1}^{M} \left(4\pi\left(\frac{r_M}{r_i} + 1\right)\right) = 4\pi \cdot r_M \sum_{i=1}^{M} \frac{1}{r_i} + \sum_{i=1}^{M} 4\pi < 4\pi \cdot r_M \cdot \frac{M}{r_1} + 4\pi \cdot M = 4\pi M(k + 1).$$

**Lemma 6.5.3.** A dominator is connected to at most $8M$ dominatees in $T_F$.

**Proof.** Consider a dominator $u_m$, where $1 \leq m \leq M$. For each $a_{ij}$, $1 \leq i \leq 8$, $1 \leq j \leq m$, of $u_m$, at most one dominatee which is a local sink is connected to $u_m$, and hence $u_m$ is connected to at most $8m$ dominatees in $T$. Thus, the number of dominatees that are connected to a dominator in $T$ is bounded by $8M$. \qed

**Corollary 6.5.1.** The degree of each dominator in $T_F$ is at most $L = 4\pi M(k + 1) + 8M$.

**Corollary 6.5.2.** The degree of any node in $T_B$ is at most $\max\{K, L\}$. 
Lemma 6.5.4. The radius $R(T_F)$ of $T_F$ is at most $2 \cdot OPT$.

Proof. It is obvious that the radius $R(T_B)$ of $T_B$ is $R$. Next, let us bound the radius of each local tree. For each dominator $u_m, 1 \leq m \leq M$, a local tree in an area $a_{ij}, 1 \leq i \leq 8, 1 \leq j \leq m$, is constructed on a complete graph. Therefore, it is possible to construct an optimal spanning tree whose maximum node degree is $\Delta^*$ and whose radius is minimized subject to the degree constraint $\Delta^*$ in each area. Thus, the radius $R(T_F)$ of $T_F$ is $R + OPT$. Since $2 \leq \Delta^* \leq \Delta, R \leq OPT$, it follows that $R(T_F) = R + OPT \leq OPT + OPT \leq 2 \cdot OPT$. $\square$

Theorem 6.5.1. It is a $(\alpha, \beta)$-approximation algorithm, where $\alpha = K + L - 2$ and $\beta = 2$.

Proof. The degree of any nodes in $T_F$ is at most $\max\{\Delta^*, \max\{K, L\}\} \leq \Delta^* + K + L - 2$ for any $\Delta^* \geq 2$. The radius of the tree is at most $2 \cdot OPT$ by Lemma 6.5.4. Therefore, it is a $(\alpha, \beta)$-approximation algorithm, where $\alpha = K + L - 2$ and $\beta = 2$. $\square$

In BDMRST-DG (Algorithm 21), Step 1 takes $O(n^2)$ time (Li et al., 2009; Wan et al., 2002), and Steps 2 – 4 simply take $O(n)$ time. In Step 5 for constructing the local spanning trees in Algorithm 22, the outermost for-loop (Steps 2 – 26) takes $O(n)$ time, and the second for-loop (Steps 3 – 25) takes constant time. The other for-loop (Steps 6 – 23) takes $O(n)$ time, but Steps 10 – 21 including the innermost while-loop take $O(n^2)$ time. Thus, BDMRST-DG has $O(n^4)$ time complexity.
CHAPTER 7
CONCLUSIONS

7.1 Summaries

This dissertation has focused on several problems in Wireless Sensor Networks (WSNs) such as scheduling of wireless sensor nodes (Chapters 2, 3, 4 and 5) in two interference models, the graph model and the physical interference (SINR) model, and computing infrastructures for WSNs (Chapter 6).

Chapter 2 studied the Minimum Latency Aggregation Scheduling (MLAS) problem in the 2-dimensional (2D) graph-based interference model. This chapter showed an \( \Omega(\log n) \) approximation lower bound for the metric model under the assumption that the nodes in the network can have non-uniform power levels. In addition, it introduced a heuristic algorithm for non-uniform power model as well as an \( O(1) \)-approximation algorithm for WSNs with a uniform power level. It also compared the performances of the algorithms against the SDA algorithm proposed in (Chen et al., 2009).

Chapter 3 then investigated the MLAS problem in the more general 3-dimensional (3D) wireless sensor networks adopting both interference models, the graph-based interference model and the SINR model. This chapter proved the NP-hardness of the MLAS problem, and proposed two \( O(1) \)-approximation algorithms which are the first results for 3D wireless sensor networks.

Chapter 4 discussed the Minimum Latency Gossiping (MLG) problem in the 2D graph-based interference model as well as the SINR model. This chapter showed the first NP-hardness result in the SINR model, and proposed an \( O(1) \)-approximation algorithm that works in both interference models. In the collision-free graph model, the approximation
ratio is 224 which is currently the best ratio and is an improvement on the approximation ratio of 258 given by (Krzywdzinski, 2010). In the collision-interference-free graph model and the SINR model, the results are the first $O(1)$-approximation algorithms.

Chapter 5 concentrated on the Minimum Channel Assignment (MCA) problem in the 2D SINR model. This chapter showed the first NP-hardness result in the SINR model, and proposed two approximation algorithms with $O(1)$-approximation ratios, assuming the uniform power model, which are the first results in the literature.

Lastly, Chapter 6 explored the Bounded-Degree Minimum-Radius Spanning Tree (BDMRST) problem. This chapter studied the problem on disk graphs as well as unit disk graphs, while all existing works studied the problem in unit disk graphs only. It showed that the problem is NP-complete, and introduced an $(8, 4)$-bicriteria approximation algorithm, which is an improvement over the bound $(10, 7)$ obtained in (Ghosh et al., 2011), for the problem on unit disk graphs. It then introduced a bicriteria approximation algorithm that computes a spanning tree of a disk graph. This result is the first for disk graphs in the literature.

### 7.2 Future Works

My future research goal is to continue exploring a broad range of applications in WSNs, and expand my research areas to mobile networks. Below is a description of my future research directions.

#### 7.2.1 Scheduling with Power Control

In contrast to the traditional WSNs, recent advances in WSNs have led to the development of new wireless sensor devices that can collaboratively determine and adjust their transmission power levels. However, in the literature, most studies of the scheduling problems in WSNs have been done assuming *no power control*, i.e., a uniform power assignment is typically used and selecting power levels is not part of the problems. Most of the works in this
dissertation also have studied scheduling problems without power control. Considering the desired features of WSNs such as energy efficiency, connectivity or fault tolerance, we observe that assigning large transmission power levels to nodes causes inefficient energy consumption, shortening the network lifetime, whereas assigning too small transmission power levels to nodes may result in disconnected network topologies. Therefore, some researchers have started investigating problems in WSNs with power control. I also plan to continue and expand the studies of the scheduling problems with power control concerning data aggregation, gossiping, data collection, beaconing, broadcasting, and channel assignment.

7.2.2 3-Dimensional Wireless Networks

Most works in WSNs have studied the problems of WSNs in the 2D space, and there are surprisingly few studies done for the 3D model. Most of the studies in this dissertation have also been done for 2D WSNs, except Chapter 3. I plan to not only continue the current studies, but also to explore other problems for applications such as data collection, beaconing, and broadcasting in 3D WSNs.

7.2.3 Mobile Networking and Computing

The well-known problems such as computation of efficient routing protocols and management of fault tolerance in WSNs that consist of static wireless sensor devices become more challenging in the mobile environment due to the dynamic nature of the network topologies caused by the movement of the sensor devices. As mobile networks have dynamic, unreliable, resource-constrained and unpredictable topologies, algorithms proposed for static wireless networks cannot be directly applied to the mobile wireless networks. Therefore, new efficient and robust algorithms are very much desirable, and I plan to extend results from my previous research in static wireless networks to mobile networks.
REFERENCES


VITA

Min Kyung An was born in Jeju, Republic of Korea (South Korea), on August 6, 1981. She received her B.S. in Computer Science and Statistics from Jeju National University, Republic of Korea, in 2004, and received her M.S. in Computer Science from The University of Texas at Arlington in 2007. During her M.S. studies, she received the Graduate Studies Abroad Program Scholarship funded by the Korean government. She began her Ph.D. studies in the Department of Computer Science at The University of Texas at Dallas in 2008 under the supervision of her thesis advisor, Dr. Dung T. Huynh. Her major areas include wireless ad hoc and sensor networks, design and analysis of approximation algorithms, graph theory, and program analysis.

In the following, all publications she authored/co-authored during her Ph.D. study at The University of Texas at Dallas are listed.


