

ALGORITHMIC ASPECTS OF THROUGHPUT-DELAY PERFORMANCE FOR
FAST DATA COLLECTION IN WIRELESS SENSOR NETWORKS

by

Amitabha Ghosh

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

August 2010

Copyright 2010

Amitabha Ghosh

Dedication

To my loving parents, whose sacrifice and blessings have always guided me through life.

Acknowledgments

I have been very fortunate to interact and work with some of the brightest minds during my graduate studies in the last 5 years. First and foremost, is my academic advisor Prof. Bhaskar Krishnamachari, without whose support, enthusiasm, and meaningful guidance this thesis would not have been possible. I have learned things from Bhaskar that go well beyond the regular, traditional Ph.D. advice, which have helped me to grow both in my professional and personal life. Besides my academic advisor, I am grateful to have interacted with my collaborators Prof. Anil Vullikanti from the Department of Computer Science and Bioinformatics Institute at Virginia Tech, Dr. Ozlem Durmaz Incel from the Networking Laboratory at the Bogazici University, Turkey, and my colleague/lab-mate Yi Wang. I would also like to thank my committee members Prof. Michael Neely, Prof. Cauligi Raghavendra, Prof. John Silvester, Prof. Murali Annavaram, and Prof. Gaurav Sukhatme whose useful feedback has helped to improve the quality of this thesis.

Specifically, all the materials in Chapter 3 and parts of Chapter 5 were joint efforts with Bhaskar, Anil, and Ozlem. Some of the results presented in Chapter 3 were published in the 6th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), 2009 [50], and are under revision along with some additional materials

contributed by Ozlem on empirical evaluations of raw-data convergecast in the IEEE Transactions on Mobile Computing (TMC), 2009 [69]. The approximation algorithms described in Chapter 3 on multi-channel scheduling, and those in Chapter 5 on bicriteria formulations are the result of many brainstorming sessions with Anil and Bhaskar. Some of these results got published as a poster in the 29th Annual IEEE Conference on Computer Communications (INFOCOM), 2010 [52], and also under submission in the IEEE/ACM Transactions on Networking, 2010 [51]. Chapter 6 on topology control is a joint work with Yi that we did as a class project guided by Bhaskar. This work was published in the 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2007 [53]. Lastly, the materials in Chapter 4 on SINR multi-channel scheduling are again joint work with Anil and Bhaskar.

Apart from the visible contributions cited above, I would like to take this opportunity to thank all the past and present members of the Autonomous Networks Research Group (ANRG) [6] of which I have been a member throughout my graduate studies. In particular, I thank Marco, Shyam, Kiran, Dongjin, Joon, Sundeep, Avinash, Hua, Ying, and Mahesh for not only being my lab mates but also great friends who have helped me on numerous occasions and have shared many happy moments that I will always cherish.

In the last five years, I spent a lot of time playing table tennis as part of the USC table tennis team Ping Pong Posse [142]. I made some great friends there. In particular, I would like to thank Adam, Imran, Kedar, Jordon, Jason, Jimmy, Subal, Misha, Jack,

Jeremy, Akhilesh, Lei, Cathy, Sarah, and Alice who accompanied me at various local tournaments as well as in the collegiate nationals organized by NCTTA [114]. Table tennis has become an integral part of my life, and, through the support and encouragement of my friends has often given me the strength and a sense of accomplishment when I had felt down.

I would also like to thank my house mates, Jeff, Sophie, Sherry, Annabelle, Min, Shuyan, and Tao Ran, with whom I had a wonderful time living under the same roof during the final year of my Ph.D. Through our traditional weekly family dinners, the barbecue parties in the backyard, the birthday trips to Yogurt Land, the hikes to the beautiful trails of Santa Monica and Malibu mountains, and the skiing adventures to the Big Bear Lake [127], we shared numerous moments that have made my life richer.

I have also been blessed to have some wonderful friends in my life who have seen me through times of joy and sorrow. In particular, I feel a deep sense of gratitude toward my long time friends Prateek, Shraya, Ayush, Subbu, and Janhavi who have helped me during times of crisis as well as celebrated with me during times of happiness; my loving sister Pinky; my friends from undergrad, Anup, Amit, and Nihar; and my friend Neelanjana with whom I have shared an invisible loving bond.

Table of Contents

Dedication	ii
Acknowledgments	iii
List Of Figures	ix
Abstract	xiv
Chapter 1: Introduction	1
1.1 Wireless Sensor Networks	3
1.2 Data Communication Patterns in Sensor Networks	8
1.3 Data Acquisition Models in Sensor Networks	11
1.4 Link Scheduling in Wireless Networks	12
1.4.1 Medium Access Control Mechanisms	14
1.4.2 Communication Using Multiple Frequency Channels	15
1.5 Routing Topologies in Sensor Networks	20
1.6 Power Control Mechanisms in Sensor Networks	22
1.7 Contributions and Organization	24
Chapter 2: Background	27
2.1 Link Scheduling for Fast Data Collection	27
2.1.1 Single-Channel Scheduling	29
2.1.1.1 Maximizing Raw-Data Convergecast Throughput	29
2.1.1.2 Maximizing Aggregated Convergecast Throughput	36
2.1.2 Multi-Channel Scheduling	40
2.1.2.1 Maximizing Raw-Data Convergecast Throughput	43
2.2 Spanning Trees for Fast Data Collection	47
2.2.1 Bicriteria Network Design Problems	49
2.3 Topology Control in Wireless Networks	55
2.3.1 Why Topology Control?	55

Chapter 3: Maximizing Convergecast Throughput in Tree-Based Sensor Networks	60
3.1 Motivation	60
3.2 Preliminaries	62
3.2.1 Model and Assumptions	62
3.2.2 Problem Formulation	66
3.3 Multi-Channel Scheduling on General Graphs	67
3.3.1 Complexity of Multi-Channel Scheduling and Frequency Assignment Problems	68
3.3.2 Scheduling Under Sufficient Frequencies	75
3.4 Multi-Channel Scheduling on Unit Disk Graphs	80
3.4.1 Known Routing Topologies	80
3.4.1.1 Frequency Assignment	81
3.4.1.2 Time Slot Assignment	85
3.4.2 Unknown Routing Topologies	88
3.5 Multi-Channel Scheduling on General Disk Graphs	91
3.5.1 An Integer Linear Programming Formulation	92
3.6 Evaluation	96
3.6.1 Frequency Bounds	97
3.6.2 Multiple Frequencies on Schedule Length	98
3.6.3 Scheduling Under SINR Model	99
3.7 Summary	100
Chapter 4: Multi-Channel SINR Scheduling for Fast Convergecast	102
4.1 Motivation	102
4.2 Preliminaries	104
4.2.1 The SINR Model	104
4.2.2 Problem Formulation	106
4.3 Multi-Channel Scheduling Under SINR	107
4.3.1 Overall Approach	107
4.3.1.1 Link Diversity	107
4.3.1.2 Reusing Time Slots Under SINR	108
4.3.2 $O(g(E_T))$ Approximation Algorithm	113
4.4 Summary	116
Chapter 5: Optimal Spanning Trees for Maximizing Throughput and Minimizing Packet Delays	118
5.1 Motivation	118
5.2 Bicriteria Problem Formulation	124
5.3 Routing Tree Construction	125
5.3.1 An Approximation Algorithm	126
5.3.2 Algorithm Analysis	130
5.4 Evaluation	134

5.4.1	Schedule Length and Maximum Delay	135
5.4.2	Multiple Frequencies on Schedule Length	137
5.5	Summary	140
Chapter 6: Efficient Distributed Topology Control in 3-Dimensional Wireless Networks		143
6.1	Motivation	143
6.2	Preliminaries and Approach	147
6.3	Phase 1: Multi-Dimensional Scaling in 3-D	149
6.4	Phase 2: Orthographic Projections	150
6.5	Phase 2: Spherical Delaunay Triangulation	156
6.6	Evaluation	163
6.7	Summary	171
Chapter 7: Conclusions and Future Work		173
7.1	Multi-Channel TDMA Scheduling	174
7.2	Optimal Routing Topologies	176
7.3	Transmission Power Control	177
7.4	Future Work	178
7.4.1	Real Testbed Implementation	178
7.4.2	Other Joint Objectives	179
7.4.3	Traffic Patterns	179
7.4.4	Cross Layer Solutions	180
7.4.5	Realistic Interference Models	181
References		183

List Of Figures

1.1	(a) Components of a sensor node. (b) A Micaz node.	4
1.2	(a) Multicast - data flows from a single node s to a set of nodes a , b , and c . (b) Convergecast - data flows from nodes a , b , and c to a single node s	10
1.3	Multi-channel hidden terminal problem	16
2.1	Optimal time scheduling for a 10-node line network with minimum schedule length 11. Upper part shows the schedule for packet distribution, bottom part shows the schedule for convergecast, which is obtained by symmetry, i.e., by reflecting the upper schedule with respect to the fictitious horizontal line. Note that nodes that are closer than or at 2 hops do not transmit concurrently to respect interference issue.	31
2.2	(a) A linear network with initial state assignment (R, I, T) depending on the hop distance from the sink s . (b) State transition of the nodes. (c) A multi-line network as a composition of multiple linear networks.	34
2.3	Raw-data convergecast using algorithm LOCAL-TIMESLOTASSIGNMENT: (a) Schedule length 7 when secondary conflicts are eliminated. (a) Schedule length 10 when secondary conflicts are present.	45
2.4	Benefits of topology control: (a) Network without topology control in which nodes are configured at maximum transmission range leading to high node degree. (b) Network without topology control in which nodes are configured at minimum transmission range leading to network partition. (c) Network with topology control in effect in which nodes adjust their transmission range leading to low average node degree and yet a connected network.	57

3.1	(a) Concurrent transmissions on adjacent edges e_1 and e_2 cause primary conflict. (b) Concurrent transmissions on edges e_1 and e_2 cause secondary conflict if their receivers are on the same frequency, say f_1 , and if either of the receivers is within the range of the other transmitter. . . .	64
3.2	Aggregated convergecast: (a) Schedule length of 6 time slots using one frequency. Dotted lines represent secondary conflicts. (b) Schedule length of 3 time slots using two frequencies. Note that, all the secondary conflicts are eliminated. (c), (d) Nodes from which aggregated data is received by their corresponding parents in each time slot over 2 consecutive frames for (a) and (b), respectively.	65
3.3	Reduction for the Multi-Channel Scheduling Problem: (a) Gadget for each v_i in G' when the number of frequencies q is 2. (b) Instance G' of the vertex color problem. (c) Instance G of the Multi-Channel Scheduling Problem as constructed from G' for $q = 2$	70
3.4	Reduction from the Vertex Color problem to the Frequency Assignment Problem. The left part of the figure shows as instance of the Vertex Color problem, which is colored using 3 colors. The right part of the figure shows the corresponding instance of the Frequency Assignment Problem, which also requires 3 frequencies to remove all the secondary conflicts.	73
3.5	(a) Original graph G ; receiver nodes are shaded. (b) Constraint graph G_C and a frequency assignment to the receivers according to Largest Degree First. Here, 4 frequencies are sufficient to remove all the secondary conflicts, i.e., frequencies on adjacent nodes are different.	77
3.6	An optimal time slot assignment according to Algorithm BFS-TIMESLOT-ASSIGNMENT yielding a schedule length 3 after all the secondary conflicts are removed using 4 frequencies.	78
3.7	(a) Frequency assignment according to Algorithm FREQUENCY-GREEDY. Load on frequencies: $\ell(f_1) = 5$, $\ell(f_2) = 5$. White colored nodes transmit on frequency f_1 ; gray colored nodes transmit on frequency f_2 . (b) Four pair-wise disjoint sets of time slots γ_1 , γ_2 , γ_3 , and γ_4 schedule the whole network. Each set γ_i maps to a unique color. Edges whose receivers lie in non-adjacent cells can be scheduled simultaneously, e.g., edges (u, v) and (u', v')	84

3.8	Number of frequencies required to remove all the secondary conflicts as a function of network density on shortest path trees.	97
3.9	Effect of multiple frequencies: Schedule Lengths for SPT with network size for $K = 1, 3,$ and 5 frequencies.	99
3.10	Percentage of nodes whose schedules conflict in the SINR model for different network sizes and three different number of frequencies ($K = 1, 3, 5$) on an SPT.	100
4.1	Coloring the grid cells in C_k corresponding to the length class E_k using four distinct colors $\gamma_1, \gamma_2, \gamma_3,$ and γ_4 . Size of each grid cell is $\eta_k = \delta \cdot 2^k$	109
4.2	For an edge $e_i = (s_i, r_i) \in E_k$, the distance between its receiver r_i and the sender s_j of another edge $e_j = (s_j, r_j) \in E_k$ whose corresponding receiver r_j lies in one of the non-adjacent layer 1 cells of the same color, is at least $\delta \cdot 2^k - 2^{k+1}$	111
5.1	Shortest path tree (SPT): High node degrees and minimum hop distances to the sink. Dark lines represent tree edges and dotted lines represent interfering links.	119
5.2	(a) Minimum interference tree (MIT): Low node degrees but large number of hops to the sink. Dark lines represent tree edges, dotted lines represent interfering links. (b) Cost of an edge (u, v) is equal to the number of nodes lying within the union of the two disks centered at nodes u and v , each of radius $d(u, v)$; here cost is 11.	120
5.3	Schedule length on minimum interference trees with different network sizes for $K = 1, 3,$ and 5 frequencies.	121
5.4	Backbone tree construction: Filled black circles represent local roots (chosen arbitrarily from each non-empty cell), and shaded cells are non-empty adjacent cells of s . Iteration 1: Local roots $r_1, r_2, r_3, r_4, r_5,$ and r_6 from non-empty adjacent cells of s are connected. Nodes r_3 and r_5 are connected to sink s via <i>helper node</i> w_1 , and node r_4 via helper node w_2 . Node r_7 is out of range of any helper node, and is not connected in the first iteration.	128
5.5	Local tree construction on an induced complete graph within each cell for maximum node degree $\Delta^* = 4$; filled black circle represents the local root.	129

5.6	Traversing the edge (u_k, u_{k+1}) along the shortest path P_G in graph G . Local roots r_k and r_{k+1} are at most distance $3R$ away from each other.	131
5.7	(a) Schedule Length, and (b) Maximum Delay (tree radius) with increasing network size on three different types of trees (BDMRST, SPT, and MIT) for single frequency scheduling.	136
5.8	Maximum Node Degree with increasing network size on three different types of trees (BDMRST, SPT, and MIT) for single frequency scheduling.	137
5.9	Effect of multiple frequencies on schedule lengths for BDMRST with different network sizes for $K = 1, 3,$ and 5 frequencies.	138
5.10	Node Degree Distribution of BDMRST, SPT, and MIT for two different network sizes with (a) $N = 150,$ and (b) $N = 800$ nodes.	139
5.11	A BDMRST constructed on the same deployment of 800 nodes of Figure 5.1 and 5.2(a). The node degrees are more uniform compared to those on an SPT and MIT.	140
6.1	Three non collinear points $p_i, p_j,$ and p_k on the surface of a sphere uniquely determine a spherical cap. The radius of the base of the cap is $r,$ and h is its height. \vec{n} denotes normal to the cap.	148
6.2	An empty sector of angle θ around u_i 's projected location on the xy plane.	151
6.3	Projected locations on xy and zx planes of node u_i and its neighbors $N_i(P^{max})$ when u_i transmits at maximum power.	153
6.4	Spherical Delaunay Triangulation illustrating empty circle property: Spherical cap $Cap(a, b, c)$ is empty in its interior.	157
6.5	The 3-D cone with $Cap(p, q, r)$ as its base and u_i as its apex is empty. Black dots show the actual locations of the neighbors, blue dots show their projected locations on the surface of the spherical ball. $\triangle pqr$ is a spherical Delaunay triangle.	158
6.6	Spherical Delaunay triangulation using the Quickhull algorithm of a set of 100 points randomly distributed on the surface of a sphere of radius 50.	160

6.7	(a) Network topology of the original maximum power graph. (b) Network topology after running the SDT-based algorithm. Here $n = 200$ and $P^{max} = 40$. Node degrees have drastically reduced.	164
6.8	(a) Node degrees of the maximum power graph and that of the final topology for $n = 200$, $P^{max} = 40$. (b) Final assigned minimal transmission power levels of nodes, for $n = 200$, $P^{max} = 40$	165
6.9	(a) Dependency of average node degree with network size. (b) Dependency of the average node degree with maximum power, for $n = 200$; node degree remains almost flat in the final topology based on the SDT algorithm.	166
6.10	Dependency of average transmission power with network size.	168
6.11	CPU execution time of the SDT based algorithm and the one [12] based on the procedure $gap-3D_{\alpha}()$ for different random topologies with $P^{max} = 40$	169
6.12	Probability of network connectivity as the θ constraint is satisfied on 1, 2, or all 3 orthogonal planes.	170

Abstract

Convergecast, namely the many-to-one flow of data from a set of sources to a common sink over a tree-based routing topology, is a fundamental communication primitive in wireless sensor networks. For real-time, mission-critical, and high data-rate applications, it is often critical to maximize the aggregated data collection rate (throughput) at the sink node, as well as minimize the time (delay) required for packets to get there. In this thesis, we look into the algorithmic aspects of jointly optimizing both throughput and delay for aggregated data collection in sensor networks. Our contributions are in designing efficient algorithms with provably good, worst-case performance bounds for arbitrarily deployed networks. To the best of our knowledge, we are the first ones to address these two mutually conflicting performance objectives – throughput and delay – under the same optimization framework and develop techniques to meet the stringent requirements for fast data collection.

Our approach in addressing the throughput-delay performance trade-off comprises three techniques: (i) multi-channel scheduling, (ii) routing over optimal topologies, and (iii) transmission power control. We exploit the benefits of multiple frequency channels to design efficient TDMA scheduling algorithms, both under the graph-based and the

SINR-based interference models. In particular, by decoupling the joint frequency and time slot assignment problem into two separate subproblems of frequency assignment and time slot assignment, we show that our scheduling algorithms have constant factor and logarithmic approximation ratios on the optimal throughput for random geometric graphs as well as for SINR-based models.

In order to further enhance the data collection rate and bound the maximum delay, we study the degree-radius trade-off in spanning trees and propose algorithms under the bi-criteria optimization framework. In particular, we construct bounded-degree-minimum-radius spanning trees that have constant factor approximations on the maximum node degree as well as the tree radius. We also show that our multi-channel scheduling algorithms perform much better on such trees in maximizing the aggregated throughput and minimizing the maximum delay, thus achieving the best of both worlds. Lastly, we design efficient, distributed power control schemes for sensor networks deployed in 3-D, where very high density of nodes causes high interference resulting in low network throughput. Our proposed algorithms have low computational overhead compared to the state-of-the-art, and by using local geometric information and tools from computational geometry produce sparse yet connected topologies in 3-D, thus reducing interference and allowing for high throughput.

Chapter 1

Introduction

We live in an era of immense technological revolution where computers and digital communications continue to transform the ways we live. From Richard Feynman's famous 1959 lecture titled *There's Plenty of Room at the Bottom* [39], to the fabrication of modern Micro-Electro-Mechanical Systems (MEMS) have led the widespread development and use of tiny computing machines that have permeated our lives. Moving away from Freeman Dyson's view on technology, that the ones "which have had the most profound effects on human life are usually simple" [34], we are entering into a time of miniaturization and disappearing of technologies [49], [139] leading to the world of embedded computing and nanodevices. While not necessarily being simpler, MEMS-enabled embedded devices [36] continue to become smaller, cheaper, and yet more powerful and capable of high storage, making them useful to a wide range of devices.

Over the last decade, Wireless Sensor Networks (WSN) [3] have appeared as one of the most prominent enabling technologies of MEMS, which combines automated sensing, embedded computing, and wireless capabilities into tiny devices, bringing promises

of understanding and instrumenting nature at scales that were unimaginable before. Just like the invention of microscope has let us see things that were previously invisible to the naked eye, wireless sensor networks have enabled us not only to detect and measure a physical phenomenon with accuracy even at the microscopic level, but also to communicate the measured information across distances using the wireless medium.

The earliest research efforts on WSN date back to the late 1990's, when the United States Defense Advanced Research Project Agency (DARPA) focused on developing low-power sensing devices to enable large-scale, distributed, networked sensor systems through the SensIT project [32]. Since then, numerous research and commercial efforts, such as the WINS [121] and Sensorsim [118] from UCLA, Smart Dust [77] and PicoRadio [19] from UC Berkeley have advanced the field from traditional simple low data-rate environmental monitoring applications, to more complex ones ranging from smart-homes and factory automation, to high data-rate mission-critical applications, such as security-surveillance, structural health monitoring, and health-care. This thesis is motivated by the need of such complex applications which require fast and timely collection of large amounts of data. Our aim is to provide mechanisms that address the challenges of fast data collection and develop algorithmic solutions that fulfill their requirements.

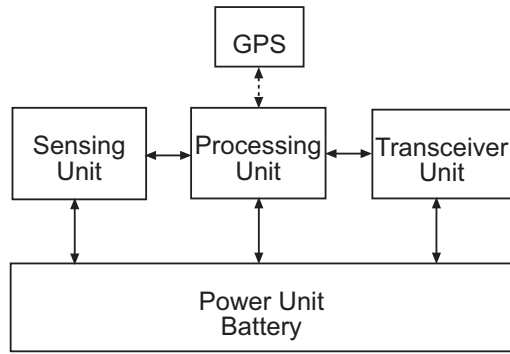
In this chapter, we first provide an overview of sensor networks in the context of this thesis by describing the general properties of sensor devices and enlisting potential applications that pertain to fast data collection. Next, we discuss the key challenges for designing efficient solutions, and present different data collection models that lead us to

the research questions addressed in this thesis. Finally, we describe our contributions and the organization for the rest of the chapters.

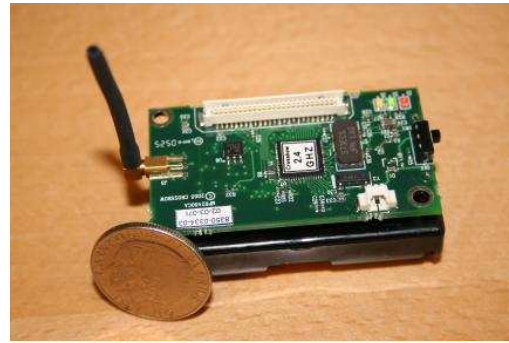
1.1 Wireless Sensor Networks

A sensor network typically comprises a large number of low-power, low-cost, tiny embedded devices with sensing capabilities, which are networked together to collect, process, and deliver information about a physical phenomenon of interest. The position of the nodes could be engineered or predetermined, such as in structural health monitoring, where nodes are placed at optimal locations to maximize the fidelity of measured vibrations for accurate and reliable diagnosis about the health of the structure. On other occasions, nodes could be placed randomly, allowing deployment of networks over inaccessible terrains or in disaster recovery operations. An individual sensor node consists of the following four basic components, as illustrated in Figure 1.1(a).

- **Sensing Unit:** It includes a variety of sensors, such as temperature, pressure, humidity, acoustic, light, vibration, etc; and optionally a few actuators, such as speakers, buzzers, and LEDs.
- **Processing Unit:** Typically, the processing unit is a 8-bit or 16-bit microprocessor that is responsible for processing raw measurements, such as carrying out simple



(a)



(b)

Figure 1.1: (a) Components of a sensor node. (b) A Micaz node.

computations, or fusing data from neighboring nodes into more meaningful quantities desired by the underlying application. Some examples of microprocessors used in current hardware are the following:

- 8 MHz Texas Instruments MSP430 microcontroller [73] with 10 KBytes RAM and 48 KBytes Flash memory used on Tmote Sky [131] platforms.
- MPR400 and MPR2400, which are based on Atmel ATmega128L and run MoteWorks from its internal flash memory. These are used on Mica2 and Micaz [30] platforms, and can be configured to run the sensor application/processing and the network/radio communication stack simultaneously. The size of the EEPROM and measurement flash memory on both these microcontrollers are 4 KBytes and 512 KBytes, respectively.
- Atmel AT91RM9200 [7] system on a chip (SOC) integrated circuit comprising ARM920T ARM Thumb processor used on the more advanced SunSPOT platforms [106]. These usually have a larger memory size consisting of 4

MBytes NOR Flash and 512 KBytes pseudo-static random access memory (pSRAM).

- **Transceiver Unit:** It comprises a low data-rate, low-power, short-range radio that usually operates on unlicensed bands, such as the 868–915 MHz band, or the 2.4 GHz industrial, scientific, and medical (ISM) band. The transmit power could vary from -24 dBm to 10 dBm with receiver sensitivity in the range of -95 dBm to -100 dBm. Typical power consumption varies in the range of 10 mA to 40 mA, and data rates in a range of 50 Kbps to 250 Kbps. Some of the state-of-the-art transceivers can also support multiple frequency channels. Common examples include the IEEE 802.15.4 compliant TI CC1000 [71] and CC2400 [72] (formerly Chipcon) radios used on Tmote Sky and SunSPOTs, and the Nordic Nrf905 [130] radios.
- **Power Unit:** The sensor nodes typically run on rechargeable AA batteries supplying a voltage of 2.7 V – 3.3 V, such as on Mica2, Micaz, and Tmote Sky platforms, or could be charged using USB power like in SunSPOTs. Since these batteries are not replenishable once the network is deployed, energy efficiency is a big concern with protocols that run on these devices.

In addition to the above four basic components, a sensor node can optionally have a satellite-based Global Positioning System (GPS) attached to it. Since in many cases the measurements need to be location stamped, the easiest way to obtain such positioning information for applications deployed in the outdoors is via a GPS device. However, for

networks deployed in the indoors, nodes must obtain their locations indirectly through network localization algorithms.

Being equipped with a variety of sensors and wireless communication capabilities, the possible applications of sensor networks are numerous. We list a few areas in the following, acknowledging that it can certainly be extended with the growing interest in both academia and industry.

- **Environmental:** Some of the environmental applications of sensor networks include tracking the movements of birds (*Great Duck Island* [100]), small animals, and insects; monitoring temperature and soil conditions that affect crops and livestock [123]; precision agriculture [10]; marine networks [43] for monitoring coral reefs; pollution study; monitoring Alpine permafrost [63]; etc.
- **Military and Security Surveillance:** As with many other information technologies, sensor networks originated primarily in military-related research, where unattended networks are envisioned as the key ingredient toward network-centric warfare systems. They can be an integral part of military command, control, surveillance, reconnaissance, target enemy classification, and tracking [104].
- **Disaster Early Warning Systems:** Sensor networks can be useful in providing early warnings about imminent flood [33], wildfire [159], volcano eruption [151], as well as hazardous substance detection [120], such as chemical contamination, etc.

- **Structural and Seismic Monitoring:** A growing class of application of sensor networks pertains to monitoring the condition of civil infrastructures [156], [26], such as buildings, bridges, roads, aircrafts, etc. Traditional structural safety assessment methods are often dependent on visual inspection or using technologies such as X-rays and ultrasound, which are manual, expensive, and time consuming. Unattended networked sensing can examine the structural integrity in an automatic and efficient manner, thus proactively detecting and preventing future damages. A particularly compelling futuristic application is the development of controllable structures, which would contain actuators to react to real-time sensor information to perform “echo-cancellation” on seismic waves in order to protect it from external disturbances, such as earthquakes.
- **Industrial and Building Automation:** In industrial manufacturing facilities, such as multi-stage chemical processing plants, sensors and actuators can be used for process monitoring and control [113] [122]. The key advantage of using wireless networks in these environments is the reduced cost and improved flexibility associated with installing, maintaining, and upgrading wired systems. In the context building automation, sensor networks can be used for controlling heating, ventilation, and air conditioning (HVAC), lighting, refrigeration, etc.
- **Health Care:** Continuous and remote monitoring of physiological patient data and tracking doctors and patients inside hospitals [60] are some of the typical health

care applications of sensor networks. In addition, recent development of intelligent physiological sensors can be integrated into a wearable wireless body sensor network (BSN) [13], which can be used for computer assisted rehabilitation and even early detection of medical conditions.

1.2 Data Communication Patterns in Sensor Networks

Sensor networks are characterized by ad hoc multi-hop networks that are capable of self-organizing without the help of any external infrastructure. Once a network is deployed, the nodes collect data about a physical phenomenon of interest, process it locally, and send it toward a common sink node, which can perhaps fuse all the received data and make intelligent decisions. The sink node is typically high-powered, such as a laptop, with larger memory and processing power. Although in small-scale, single-hop networks, direct communication between the nodes and the sink is possible, the most common form of communication in large-scale, multi-hop networks is peer-to-peer, i.e., among neighboring nodes. This peer-to-peer communication over short distances is ideal for low-power, short-range radios, and allows nodes to cooperate and collectively work toward a common goal.

There are three basic data communication patterns in sensor networks: (i) convergecast, (ii) unicast or local broadcast, and (iii) multicast. We describe each of them in details below.

- **Convergecast:** It is a *many-to-one* communication pattern [46, 47], where data flows from a set of nodes toward a common sink over a tree-based routing topology. This is the most common form of data communication and constitutes a fundamental operational primitive in sensor networks. When the sensor readings are correlated due to spatial/temporal proximity of the nodes, or when the application requires summarized information, data is often combined or *aggregated* at each hop en route to the sink. *Data aggregation* [98] has been put forward as an essential paradigm for wireless routing in sensor networks, where the idea is to combine the data coming from different sources before transmitting to the upstream node toward the sink. It has been shown that aggregation can eliminate redundancy and minimize the number of transmissions, thus saving energy. This paradigm shifts the focus from traditional address-centric approaches for networking (finding short routes between pairs of addressable end-nodes) to a more data-centric approach, i.e., finding routes from multiple sources to a single destination that allows in-network consolidation of redundant data. We refer to the convergecast process under aggregation as *aggregated convergecast* [50, 51], and distinguish it from *raw-data convergecast* [69, 70] when there is no aggregation.

- **Unicast:** It is a form of local broadcast where a node exchanges data with its local neighbors, for instance, to perform collaborative data processing and fusion instead of transmitting raw sensor readings.

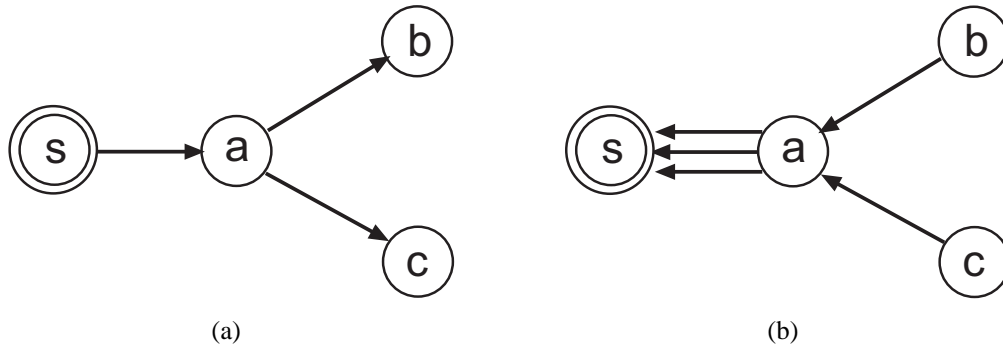


Figure 1.2: (a) Multicast - data flows from a single node s to a set of nodes a , b , and c . (b) Convergecast - data flows from nodes a , b , and c to a single node s .

- **Multicast:** It can be considered as opposite to convergecast, i.e., *one-to-many* communication pattern in which data is disseminated from the sink to a set of nodes. Multicast could be used, for instance, in reprogrammable sensor networks where the sink supplies automatic network-wide updates of system software or reconfiguration information to all the nodes.

Figure 1.2 shows a simple example that illustrates the characteristics of a typical multi-cast and convergecast. In a multi-cast, as shown in Figure 1.2(a), node s is the message source and nodes a , b , and c are its expected recipients. Node a hears the message directly from s and forwards a copy to nodes b and c using a single local broadcast. In case of a convergecast, as shown in Figure 1.2(b), nodes a , b , and c each has a message destined to sink s , and a serves as a relay for b and c . In this case, we assume there is no aggregation, and so node a transmits three messages as indicated by the three arrows.

1.3 Data Acquisition Models in Sensor Networks

Convergecast or many-to-one communication being the most fundamental form of data collection in sensor networks, it is natural to ask what triggers the data collection process; in other words, what are the data acquisition models in sensor networks. Here, we present a classification of the most common types of data acquisition models [143] and give examples of applications to which they are relevant.

- **Continuous/Periodic Delivery:** This form of data collection is most relevant for real-time, mission-critical applications where sensing and collection are performed synchronously, or for applications where periodic notification of events is required. Such data collection usually takes place over long durations of time ranging from low data-rate scenarios, such as in surveillance and habitat monitoring, to high data-rate scenarios, such as in structural health and permafrost monitoring. Since the nodes need to continuously sense and transmit data, energy efficiency is a big concern for continuous and periodic data collection.
- **Query-Driven Delivery:** In this model [21], the nodes send data only when triggered by an external query fed into the network by the sink node. Since the nodes could sleep most of the time and wake up to collect data only when triggered by the queries, this delivery model is efficient in terms of energy consumption. Typical application scenarios for query-driven data delivery includes getting a snapshot

view of the network, or sending back data/acknowledgments in response to software reconfigure/upgrade messages sent by the sink. Such data delivery usually spans over short intervals.

- **Event-Driven Delivery:** The sensor nodes could be programmed to deliver data whenever an event of interest occurs within the network. This mode of data acquisition is useful when the events are rare but critical. However, such events could also trigger huge bursts of data that require immediate delivery. Since the nodes need to sense the environment continuously for possible occurrence of the events, but transmit only when an actual event occurs, energy consumption is a lesser severe concern compared to the case of continuous/periodic delivery.

In this thesis, we focus on continuous/periodic data collection over long durations of time for high data-rate applications.

1.4 Link Scheduling in Wireless Networks

The nodes in a terrestrial wireless network communicate using radio frequencies (RF), commonly known as the *spectrum*. It is well understood that the strength of a radio signal attenuates according to the inverse square of distance between the transmitter and the receiver under a free-space propagation model when a clear unobstructed line-of-sight exists. However, in most cases, due to the presence of obstacles, such as buildings, mountains, etc, and other environmental parameters, a radio signal undergoes reflection,

absorption, scattering, and diffraction, which makes the signal strength attenuate and fade nonuniformly over distance. This makes wireless communications unreliable and error-prone. Moreover, multiple transmissions within close proximity of each other taking place at the same time and on the same frequency might interfere and may not be correctly decoded by their intended receivers. This calls for efficient mechanisms for scheduling links appropriately in the frequency, time, or code domain, so as to avoid mutual conflicts. We refer to any combination of time/frequency/code as a *channel*, and the problem of assigning channels to the links as the *link scheduling problem*.

With the rapid growth of wireless networking and embedded computing, leading to the proliferation of ubiquitous wireless devices into our everyday lives, it is likely that extremely high premiums are placed on the communications spectrum. The scarcity of spectrum necessitates efficient channel assignment mechanisms. Whether the channel sharing is based on Time Division Multiple Access (TDMA), Frequency Division Multiple Access (FDMA), Code Division Multiple Access (CDMA), or any combination thereof, there is a fundamental limit on the number of *nodes* or *links* sharing the same channel simultaneously. This has motivated the need for *spatial reuse* of the channels by having links that are sufficiently far apart to use the same frequency, time slot, or code.

Scheduling link transmissions so as to optimize one or more of the performance objective, such as throughput, delay, or energy, has spurred a lot of research interest in the last few decades. Unlike wireline networks, where all the links have fixed bandwidth, the effective bandwidth of a wireless link is influenced by channel variations due

to fading, transmission power, routing, changes in network topology, etc. Moreover, in resource constrained wireless networks, such as in sensor networks where the nodes are low-powered and equipped with a single half-duplex radio capable of either transmitting or receiving at any given time, designing efficient scheduling protocols in the face of varying channel conditions and network topology brings in additional challenges.

1.4.1 Medium Access Control Mechanisms

For ad hoc networks in general, and sensor networks in particular, there are two primary medium access control (MAC) mechanisms: (i) contention-based, and (ii) contention-free. In contention-based medium access, such as time-slotted Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), each node needs to contend for the medium before being able to transmit. These protocols are known to have lower delay and promising throughput at lower traffic loads, such as in query-driven data collection. However, when the network load is high, there is a higher chance of collisions and back-offs, thus wasting a lot of bandwidth. On the other hand, in contention-free medium access, such as Time Division Multiple Access (TDMA), the nodes do not need to contend for the medium; instead, time is slotted (fixed or variable durations), and each node is assigned a particular time slot to transmit. This eliminates collisions and minimizes retransmissions. Such contention-free protocols are better suited for higher traffic load that sustains over long durations, such as in the continuous/periodic data collection

model [102]. However, they might incur higher delay and lower throughput for low network traffic, because some time slots might go wasted if nodes do not have any packets to transmit.

In wireless sensor networks, since radio communication drains the maximum amount of energy compared to other computational processing required by the sensor nodes, reducing the number of retransmissions required to successfully send a packet by avoiding collisions saves a lot of energy. Due to low-cost, small form factor, and energy efficiency requirements, the radio capabilities on the sensor nodes are limited compared to other wireless devices. In addition, the nodes transmit at a low power making the effective bandwidth go down even further. Considering all these limitations, and the fact that we focus on continuous/periodic data collection for high data-rate applications, we propose to use contention-free TDMA scheduling protocols that will help in achieving improved throughput and delay.

1.4.2 Communication Using Multiple Frequency Channels

Until recently, most of the protocols developed for wireless sensor networks used only a single frequency channel for communication. However, this turns out to be not good enough to provide reliable and timely delivery for large amounts of data generated by high data-rate applications. This calls for the need of using multiple frequency channels. The basic idea behind communication using multiple frequencies is to tune neighboring

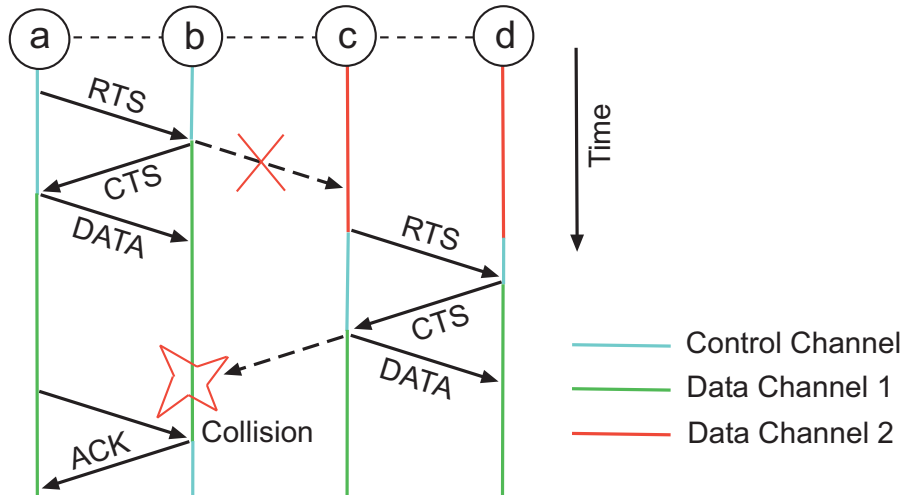


Figure 1.3: Multi-channel hidden terminal problem

transmitters and receivers on different orthogonal frequencies, such that concurrent transmissions do not interfere with each other, thus leading to increased network throughput. Newer generations of commercially available radios, such as the CC2420 and Nordic nRF905 radios, support multiple frequencies in order to comply with the IEEE 802.15.4 standard. The IEEE 802.15.4 standard is used as the basis for ZigBee [116], WirelessHART [137], and MiWi specifications, and provides a framework for low data-rate communication systems, such as wireless personal area networks (WPAN).

Along with its advantages, multiple frequencies also bring in two additional challenges in terms of synchronization that need special care. These are known as the *multi-channel hidden terminal problem* [135] and the *deafness problem* [99], which occur due to the fact that the nodes may be tuned to listen to different frequencies at different times. We briefly explain these two problems in the following.

- **Multi-Channel Hidden Terminal Problem:** Suppose there are K frequency channels available; one channel is used for exchanging control messages, known as the *control channel*, and all others are used for exchanging data and acknowledgments, known as the *data channels*. When a node is neither transmitting or receiving, it listens to the control channel. Suppose that the underlying MAC protocol is CSMA/CA. When node a wants to send a packet to node b , it sends out a ready-to-send (RTS) message to node b on the control channel, possibly with a list of data channels it is willing to use (cf. Figure 1.4.2). On receiving the RTS, node b chooses a data channel from the list, say channel 1, and sends out a clear-to-send (CTS) message on the control channel to node a . At this point, the handshake is complete and if there is no other node trying to transmit in the neighborhood, no collision occurs. However, suppose when node b sent the CTS to node a , there was another node c in the neighborhood that was busy receiving on another channel, say channel 2, from some other node d , and so it did not hear the CTS. Thus, not knowing that node b is receiving on channel 2, node c might initiate a communication with node d also on channel 1, resulting in a collision at node b .
- **Deafness Problem:** The deafness problem is also associated with CSMA/CA RTS/CTS based protocols. It occurs when a node sends an RTS message on the control channel to initiate a transmission when the destination is tuned to a different data channel. After trying multiple times, the transmitter might give up and conclude that the destination is not reachable anymore.

There are three primary channel assignment methods in multi-channel communication.

- **Fixed Channel Assignment:** In this method, the network is partitioned into different clusters, and nodes within each cluster are assigned the same channel, which remain unchanged throughout the operation of the network. An example of this type of channel assignment is presented in [158]. The clusters are assigned channels such that the interference among neighboring clusters is minimized. This static assignment strategy has the advantage of ease of implementation and incurring almost no overhead as any sender-receiver pair is always tuned to the right channel. However, it is not adaptive to network dynamics and can partition the network when certain links go down. For example, consider a network topology where a cluster of nodes is connected to the rest of the network using only a single link that is tuned a fixed channel. If that link goes down due to excessive neighborhood transmissions by other devices on the same channel, the network will get partitioned. Different from the idea of clustering nodes into different frequencies, Vedantham *et al.* [147] introduce the concept of *component-based channel assignment*. In this approach, all links in a connected component, induced by a flow graph between sources and destinations, operate on a single channel.
- **Semi-Dynamic Channel Assignment:** In this approach, the nodes are assigned fixed channels to start with, but can change channels during communication if they

experience interference, thus avoiding the network partitioning problem. Semi-dynamic assignment strategy, however, needs a coordinator node to make sure that a sender-receiver pair is tuned to the same frequency while transmitting in order to avoid the deafness problem. Moreover, in case of CSMA/CA RTS/CTS based protocols, such semi-dynamic channel assignment might run into *multi-channel hidden terminal problem*. Examples of semi-dynamic assignment are presented in [99] for wireless mesh networks, and in [132] for multi-hop packet radio networks where receivers are assigned fixed channels and the transmitters switch to those channels for communication.

- **Dynamic Channel Assignment:** In this method, every sender-receiver pair tunes on a particular channel on-the-fly before each packet transmission. Although such dynamic channel assignment strategy does not require time synchronization, it might incur overhead giving rise to unacceptable delays, especially for real-time data collection. In addition, when a dedicated control channel is used for data-ack channel negotiation, such as in the protocols presented in [74, 92], it runs into the control channel bottleneck problem due to frequent use. There also exist other methods of dynamic channel assignment, such as a *split phase*-based approach, used in [24, 135, 162], which eliminates the hidden terminal and deafness problems, and a frequency hopping-based approach used in [11, 109, 140, 145].

In this thesis, since we consider continuous/periodic data collection in high data-rate applications, we exploit the benefits of using multiple frequency channels in reducing interference and increasing the number of concurrent transmissions. This helps in achieving better network throughput. However, since the sensor nodes are resource constrained in terms of having a single half-duplex radio and limited energy source, frequent per-packet channel negotiation under a dynamic channel assignment strategy might cause unacceptable overheads, and so special care needs to be taken for optimal channel assignments.

1.5 Routing Topologies in Sensor Networks

Routing topologies play an important role in the performance of sensor networks in data collection [96] and data dissemination [66]. The Collection Tree Protocol (CTP) [55] is the defacto routing protocol standard for data collection. Since remote monitoring is one of the key drivers in sensor networks, data from individual nodes must be sent to a sink, often located far from the network, through the use of specific routing paths. Typically, these routing paths comprise a *spanning tree* rooted at the sink node [146]. During data collection, a node can send its data to its unique parent node, which in turn can send it to its parent and so on, until the data reaches the sink. The structure of the routing tree plays an important role in data collection. While one hand budgeting and technological constraints may require the placement of nodes at specific locations, thus sometimes

limiting network performance in terms of throughput and delay, one might, on the other hand, construct specific routing topologies to optimize performance metrics.

A routing topology can either be static or dynamic. A static routing topology has the advantage of having fixed routing paths that are a priori known to the network designer and can be leveraged for easy maintenance and optimizing certain performance metrics, such as energy consumption, throughput, delay, etc. However, since sensor nodes are subject to failure due to battery power depletion or environmental causes, static routing paths can at times be fatal in disconnecting the network. This might hamper network operations and requires additional resources for network maintenance. In addition, when new nodes join the network, recomputing some of the routing paths might be expensive. On the other hand, dynamic routing topologies, such as those used in Dynamic Source Routing (DSR) [76], routing paths are discovered only when a packet needs to be sent to a given destination. Computing such on the fly routing paths requires control messages to flow from the source to the destination, that could possibly include link quality estimation, congestion indicators, etc, which could help in establishing robust routing paths. Clearly, dynamic routing thus incurs overhead as compared to static routing.

In the context of data collection, which is the primary focus of this thesis, we consider routing topologies that are given a priori, as well as construct specific topologies that are optimized for enhancing the throughput-delay trade-off. In particular, given a deployment of nodes, our scheduling algorithms run on routing topologies that are optimized

for maximizing aggregated convergecast throughput and minimizing data collection delay. We propose algorithms to construct routing topologies that have low node degree as well as low depths. Having a low degree helps in reducing bottlenecks in the presence of multiple frequencies, and low depth helps in reducing delays.

1.6 Power Control Mechanisms in Sensor Networks

In recent years, power control has received intense attention in the domain of cellular networks [160] as well as in ad hoc networks [31]. The design of efficient power control mechanisms is crucial to the successful operation of a sensor network [136]. Power control not only saves energy consumption for these resource constrained devices, thus enhancing network lifetime, but also helps in reducing packet collision probability and wireless interference [35, 87, 136], thus allowing for higher throughput and lower delay. When combined with link reliability assessment algorithms, power control techniques can also be used to improve the reliability of links. For instance, upon detecting link reliability below a certain threshold, the MAC protocol can increase the transmission power, thus lowering the probability of receiving corrupted data. Power consumed in transmission and reception is responsible for up to 70% of the total energy consumption for a typical sensor node such as Tmote Sky and Micaz, and thus designing efficient power control protocols is a key research area.

There have been several studies in designing efficient power control mechanisms and analyzing their benefits. Instead of globally defining a transmission range that keeps a

network connected, wireless networks should adjust transmission ranges on each link. The average traffic capacity per node is constant even when more nodes are added to a fixed area network so long the network employs transmission power control. This is not true for fixed power levels, because more nodes will interfere with each other and lower the capacity.

A sensor node can perform calculations based on several readings and the network topology in order to identify the ideal transmission power. Some of these readings are: (i) Received Signal Strength Indicator (RSSI), (ii) Sensitivity, and (iii) Battery voltage. RSSI is the received signal strength measured by the transceiver and it depends on external factors, such as environmental conditions and presence of obstacles. Sensitivity is an indicator of the least power level at which a transceiver is able to detect and decode data correctly. RSSI values read from the radio are calculated with battery voltage as a reference. Thus, in order to convert any RSSI reading into an actual reception power, the battery voltage must be known.

In the context of this thesis, we use transmission power control for 3-dimensional networks to construct sparser topologies with the goal to maintain a connected network. Since the density of nodes is very high in 3-D for a network to be connected, adjusting the transmission power can provide significant benefits in reducing interference.

1.7 Contributions and Organization

The rest of the thesis is organized as follows. In Chapter 2, we provide a background on relevant topics comprising this thesis. In particular, we describe existing works on link scheduling for fast data collection in sensor networks, both with single channel and multi-channel scheduling. We also describe spanning trees and bicriteria network design problems that are suitable for fast data collection. Lastly, we describe existing works on the need for topology control and a few important topology control algorithms.

In Chapter 3, we propose a multi-channel scheduling algorithm for maximizing the aggregated convergecast throughput under a graph-based interference model. We show that by decoupling the joint frequency and time slot assignment problem into two sub-problems of frequency assignment and time slot assignment can achieve provably good worst-case performance bounds for two classes of random geometric graphs. For unit disk graphs, where nodes have a uniform transmission range, the scheduling algorithm gives a constant factor approximation, while for general disk graphs, where nodes can have different transmission ranges, it gives a logarithmic approximation. The proposed algorithms are combinatorial and use tools from approximation algorithms, graph theory, and linear programming. Along the way, we also prove several NP-hardness results on the the scheduling complexity for arbitrary networks. We evaluate the algorithms through simulations and show various trends of the scheduling performance with respect to network size and the number of frequencies.

In Chapter 5, we combine two metrics – throughput and delay – and construct efficient routing topologies that help in optimizing both in the context of aggregated convergecast. In particular, we formulate our joint throughput and delay optimization problem as a bicriteria optimization problem, and design an algorithm that minimizes the depth (radius) of a spanning tree given a certain bound on the node degree. We call such a tree a *bounded-degree minimum-radius spanning tree*, and show that our proposed algorithm gives a *constant factor* bicriteria approximation on both degree and depth of the tree. Once the routing tree is constructed, we ran the scheduling algorithms designed in Chapter 3 to evaluate the scheduling and delay performance. We show that scheduling on such a tree indeed achieves the best of both worlds in terms of maximizing the aggregated convergecast throughput as well as minimizing the maximum delay.

In Chapter 6, we consider 3-dimensional networks and designed distributed topology control algorithms from local neighborhood information with the goal to construct connected sparse topologies. We extended previous results on 2-dimensional topology control and proposed a heuristic based on orthographic projections that achieves very good performance in practice, although it does not theoretically guarantees network connectivity at all times. In addition, we also used tools from computational geometry, such as spherical Delaunay triangulation, to design a robust algorithm that achieves global network connectivity at all times, while incurring very low computational overhead as compared to existing techniques in 3-D. We evaluated our topology control algorithms

through simulations and showed various trends with average node degree, transmission power, and CPU execution time.

In Chapter 4, we consider multi-channel scheduling in the context of aggregated convergecast under a more realistic interference model, the so called *signal-to-interference-plus-noise-ratio* (SINR) model, and extended the algorithms designed in Chapter 3 to work under SINR. By using the notion of *link diversity* and appropriately dividing the network into grid cells of a certain size, we show that the worst-case scheduling complexity can be bounded by the number of non-empty length classes which, in practice, is a small constant.

In Chapter 7, we provide a summary of the contributions in this thesis and discuss various open problems and directions for future research.

Chapter 2

Background

As described in Chapter 1, this thesis focuses on throughput-delay performance for fast data collection in tree-based wireless sensor networks. In particular, we address the problem of jointly optimizing two mutually conflicting performance objectives – convergecast sink throughput and packet delays – by utilizing three techniques: (i) multi-channel scheduling, (ii) routing over optimal topologies, and (iii) transmission power control. This chapter introduces the existing literature in each of these categories, and describe their differences with our work.

2.1 Link Scheduling for Fast Data Collection

Wireless interference and bandwidth limitation are perhaps the key limiting factors to achieving high convergecast throughput in large-scale sensor networks that require periodic, real-time, and high-rate data collection. Interference restricts the number of concurrent transmissions that can take place within a given neighborhood, thus reducing

the effective bandwidth of the wireless channel and, in turn, network throughput. In addition, typical sensor nodes are equipped with a single radio that can transmit only in the order of few tens of Kbps to a few hundreds of Kbps, such as the older generation CC1000 [71] transceivers operating at 38.4 Kbps, to the newer state-of-the-art CC2420 [72] transceivers operating at 250 Kbps using spread spectrum capabilities and larger frequency bands of 5 MHz per channel. Furthermore, these radios operate on the 868/915 MHz and 2.4 GHz unlicensed ISM frequency bands, which limit their transmission power to a maximum of 10 dBm. This in turn results in a maximum transmission range of about 10 to 30 meters depending on environmental conditions. Given these limitations due to interference and bandwidth, techniques that are able to schedule a large number of spatially well separated links are required in order to achieve high throughput. To this end, utilizing multiple frequency channels gives a promising approach to overcome some of the interference-related limitations. In fact, most of the commercial radios available today support multiple frequency channels, such as the 16 orthogonal frequencies supported by CC2420 radios, or the 512 channels supported by Nordic Nrf905 radios. Thus, designing scheduling protocols by exploiting the benefits of multiple frequencies is imperative to the successful operation of large-scale sensor networks deployed for fast data collection.

In this section, we first describe the relevant scheduling literature that aim to maximize the data collection rate using a single frequency channel, and then describe the works that consider multiple frequency channels. We note that although multi-channel

communication is a well-studied research topic in wireless ad hoc networks, particularly in packet radio networks [78] and wireless mesh networks [29], such research is relatively new in the domain of sensor networks, caused partially due to the unique challenges brought forth by severe resource constraints and bandwidth limitations.

2.1.1 Single-Channel Scheduling

2.1.1.1 Maximizing Raw-Data Convergecast Throughput

We first consider the case of *raw-data convergecast*, i.e., when every packet generated by a source node needs to be delivered to a common sink without being aggregated at intermediate hops. This type of data collection is relevant for applications that require delivery of raw sensor readings, or when the redundancy in data is minimal. The problem of maximizing the convergecast sink throughput can be formulated as minimizing the number of time slots required per frame (referred to as the *schedule length*) under TDMA scheduling. Several variants of the problem exist depending on network topology, interference model, packet generation scheme, buffer constraints, antenna models, etc.

One of the early works in this category is by Florens *et al.* [40–42], who address the problem of scheduling for packet *distribution* in sensor networks, and show that it can be considered as an inverse operation of convergecast. Assuming protocol interference model, they propose optimal centralized algorithms for special network topologies, such as line, multi-line, and tree networks, for both omnidirectional and directional antennas.

For line networks where the sink sends $p(i) \geq 0$ packets to node i that is i hops away, the basic idea is to send packets destined to the furthest node first, then to the second furthest node, and so on, as quickly as possible respecting channel reuse constraints. Nodes between the sink and a packet's destination are required to forward a packet as soon as it arrives (i.e., in the next time slot following its arrival). This basic idea can also be extended to a multi-line and tree networks. The upper part of Figure 2.1 shows an example for packet distribution on a 10-node line network for directional antennas with $p(1) = 2$, $p(2) = 1$, $p(8) = 1$, and $p(9) = 1$. Once an optimal schedule is found for the distribution problem, a schedule for the convergecast problem, where node i sends $p(i)$ packets to the sink, is constructed by symmetry as shown in the bottom part of Figure 2.1. In particular, a transmission from node i to $i + 1$ occurring at time slot j for the distribution problem corresponds to a transmission from node $i + 1$ to i in time slot $T - j + 1$ for the convergecast problem, where N is the total number of nodes and T is the minimal schedule length.

Ergen *et al.* [37] prove that the problem of minimizing the schedule length is NP-hard by reducing it from the *Graph Coloring* problem. Under a graph-based interference model, they show that a conflict-free schedule can be found by coloring a conflict graph. A *conflict graph* is defined as one in which every node represents an edge in the original graph, and two nodes are connected if their corresponding edges interfere in the original graph, i.e., give rise to primary or secondary conflicts. Using such a graph coloring strategy, they propose a node-based and a level-based scheduling heuristic, and show that one

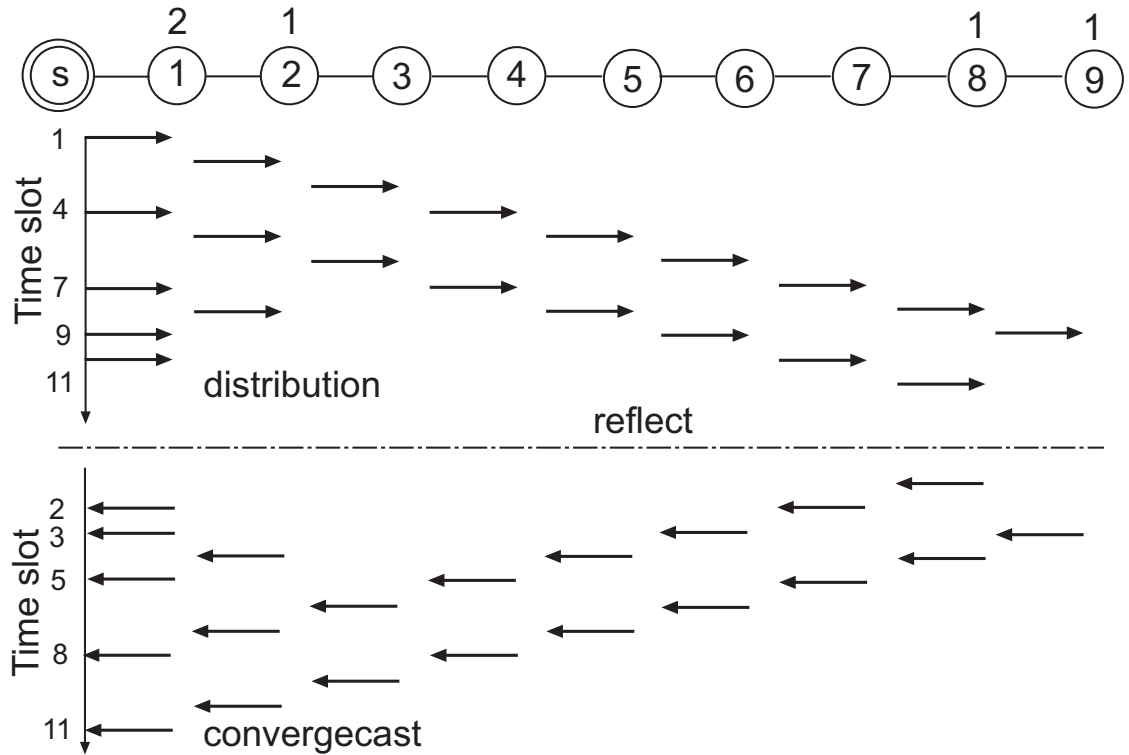


Figure 2.1: Optimal time scheduling for a 10-node line network with minimum schedule length 11. Upper part shows the schedule for packet distribution, bottom part shows the schedule for convergecast, which is obtained by symmetry, i.e., by reflecting the upper schedule with respect to the fictitious horizontal line. Note that nodes that are closer than or at 2 hops do not transmit concurrently to respect interference issue.

outperforms the other depending on the distribution of packet generation. In particular, node-based scheduling is better for topologies that have equal density of packets across the network or higher density of packets at low levels of the tree, whereas level-based scheduling is better for topologies when the packet density is higher at upper levels.

A *Virtual Node Expansion*-based approach that also uses graph coloring to find a minimum-length, conflict-free schedule, where every node generates a single packet in each frame is proposed by Lai *et al.* [90]. They first construct a conflict graph from the original graph and then expands it by creating, for each parent node, a number of virtual nodes equal to the size of the subtree rooted at that node in the original tree. This graph expansion is done to accommodate multiple transmissions by intermediate parent nodes which relay packets from nodes in its subtree. Since the virtual nodes also conflict with any node that has an edge to its original node, edges are added between the virtual nodes and the conflicting node. Similarly, an edge is added between each virtual node and its original node. Once the expanded conflict graph is constructed, an approximate coloring algorithm, originally due to Li *et al.* [94], is used to find a time slot assignment. The coloring algorithm works by finding a vertex with the least degree and removing it from all its adjacent edges. This is repeated until all the vertices are removed, after which it greedily assigns colors in the reverse order of removing the vertices. This results in a conflict-free schedule for each of the edges in the original graph.

The authors of [90] also discuss the effect of different routing structures on the schedule length and propose a *disjoint-strip* based approach to construct an efficient routing

topology. This results in uniform flow of data along different paths in the network and prevents certain nodes from being overloaded. The basic idea is to construct several disjoint, equally spaced *node strips*, all with the same number of nodes. Two distanced strips are likely to relay data simultaneously without interfering with each other. It is shown that this disjoint-strip routing, although increases the total number of transmissions, yields a shorter schedule length for unbalanced node deployments as compared to shortest-path routing, which is suitable for balanced deployments minimizing the total number of transmissions but not necessarily the schedule length.

Choi *et al.* in [28] formulate the scheduling problem as a *Minimum Information Gathering Time Problem*, where every node sends a single packet, and the goal is to find routing paths from the nodes to the sink as well as an optimal time slot assignment. By reducing it from the *Partition Problem* [105], they prove that the problem is NP-complete on general graphs and propose algorithms for line and tree topologies that take at most $3N - 3$ time slots to deliver all the packets to the sink. For general networks a heuristic is proposed, which starts with a minimum spanning tree and trims the edges such that transmissions on different branches of the tree do not interfere with each other and can be scheduled in parallel. This results in a backbone forest whose segments are then scheduled independently respecting adjacency and two-hop interfering constraints.

Following a strategy similar to [42], Gandham *et al.* in [46, 47] propose distributed scheduling algorithms for raw-data convergecast where every node generates a single packet in one data collection cycle. They give an *Integer Linear Programming* (ILP)

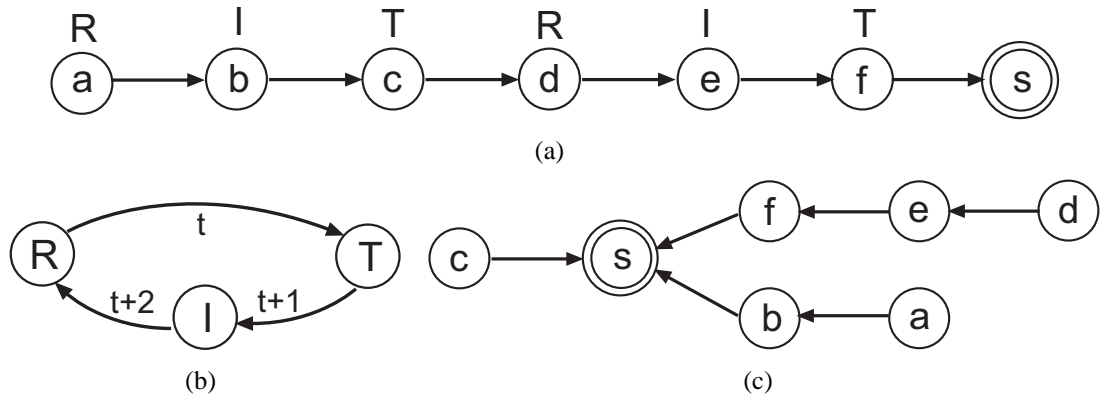


Figure 2.2: (a) A linear network with initial state assignment (R, I, T) depending on the hop distance from the sink s . (b) State transition of the nodes. (c) A multi-line network as a composition of multiple linear networks.

formulation of the problem and propose a distributed time slot assignment scheme that takes (i) at most $3N - 3$ time slots for linear networks, which is optimal, (ii) at most $\max(3n_k - 1, N)$ time slots for multi-line and tree networks, where the lower bound for multi-line networks is $\max(3n_k - 3, N)$, and (iii) at most $3N$ time slots for general networks. Here n_k represents the maximum number of nodes in any subtree of the routing structure. Similar results are also obtained by Tsai *et al.* [144].

In addition to minimizing the schedule length, the proposed algorithm in [47] also considers memory constraints on the sensor nodes and requires storage for at most two packets in each node buffer. Links are assumed to be symmetric and the interference model is assumed to be graph-based, with the interference range of a node equal to its transmission range. Their results also extend to the case where nodes generate multiple packets and when channel propagation characteristics are not ideal.

Starting from linear networks, the algorithm proposed by Gandham *et al.* [47] is generalized for multi-line networks, which are multiple linear networks intersecting with

the sink, and also to tree networks. The basic idea for linear networks is that each node is assigned an initial state depending on its hop count from the sink. As illustrated in Figure 2.2(a), a node at hop distance h is assigned a state of: *transmitting* (T) if $h \bmod 3$ is 1, *idle* (I) if $h \bmod 3$ is 2, and *receiving* (R) if $h \bmod 3$ is 0. A node comes back to this initial state after every 3 time slots and follows the state transition diagram as shown in Figure 2.2(b). Effectively, this implies that for every node in state R , there is only one node in the neighborhood which is in state T , and so packet transmission is always successful resulting in exactly one packet reception by the sink in every 3 time slots.

The above basic idea extends to more complex topologies, such as multi-line networks, as shown in Figure 2.2(c); tree networks where transmissions are scheduled in parallel along multiple branches; and general networks where packets are routed over a Breadth First Search (BFS) tree. However, for the distributed algorithm to work, each node must know the branch ID and the number of nodes in all other branches, but need not be aware of the entire network topology. The scheduling rule for multi-line networks is that the branch with the largest number of remaining packets and whose root has at least one packet gets priority to transmit (ties are broken based on the lowest branch ID). This results in a schedule length of $\max(3n_k - 1, N)$. For general networks, since there are interfering edges that are not part of the spanning tree, the goal is to first eliminate interference by constructing a BFS tree and then scheduling as before. However, in addition to knowing the number of nodes in all the branches and branch IDs, for general

networks, a node also has to know a conflict map at the initialization phase. This gives a schedule length of $3N$, although the simulations presented require only $1.5N$ time slots.

The use of orthogonal codes, such as Direct Sequence Spread Spectrum (DSSS) and Frequency Hopping Spread Spectrum (FHSS), to eliminate interference is studied by Annamalai *et al.* [5]. They propose a greedy, top-down, tree construction scheme that chooses the children of a node based on the nearest neighbor criterion starting from the sink and traversing the graph in BFS order. To reduce interference, nodes that fall within the transmission range of a parent other than their own, are assigned different codes if available; otherwise, the code that is least used by the interfering neighbors is used. Once the channel allocation is done, time slots are assigned in a greedy fashion such that a parent does not transmit before its children. Simulation results indicate that the schedule length on such a tree constructed specifically for convergecast is shorter than on a tree constructed for broadcast [27]. However, one limitation of this approach is that the miniature hardware design of sensor nodes may not permit employing complex radio transceivers required for spread spectrum codes or frequency bands systems.

2.1.1.2 Maximizing Aggregated Convergecast Throughput

Unlike raw-data convergecast where every packet generated by a node needs to be delivered to the sink, periodic data collection often requires delivery of only summarized information in the form of aggregated packets. In general, such aggregated convergecast requires less number of time slots than raw-data convergecast because of the reduced

traffic volume. Under this setting, it is assumed that every node generates a single packet at the beginning of every frame and perfect data aggregation is possible, i.e., each node is capable of aggregating all the packets received from its children as well as its own into a single packet before transmitting to its parent. It is also assumed that the size of an aggregated packet is constant and is independent of the size of the raw sensor readings.

Since the goal is to minimize the schedule length, each parent node ideally waits to receive all packets from its children and then aggregate those with its own before transmitting. Thus, in aggregated convergecast, a node transmits only once per frame and maintains an intrinsic order of transmission with respect to its children. When the routing tree is not specified as part of the problem, the algorithms also construct routing trees suitable for aggregation prior to scheduling.

One of the early works is by Chen *et al.* [25], where the problem is slightly generalized by considering only a subset $S \subseteq V$ of nodes generating data. Assuming uniform transmission range and a Unit Disk Graph (UDG) model, they formulate it as a *Minimum Data Aggregation Time* (MDAT) problem with the goal to find a collision-free schedule that routes data from the subset of nodes to the sink in minimum possible time. They prove that MDAT is NP-complete, even when restricted to UDGs, by reducing it from the restricted planar 3-SAT problem [48]. They design a centralized $(\Delta - 1)$ -approximation algorithm, where $\Delta + 1$ is the maximum number of nodes within the transmission range of any node. Their proposed approach does not assume that the routing tree is known a priori, instead the algorithm finds the data aggregation tree after the schedule is made. If

the height of the routing tree is h , then a trivial lower bound on the schedule length is $\max\{h, \log_2 |S|\}$.

The basic idea of the algorithm, called the *Shortest Data Aggregation (SDA)*, is to incrementally construct smaller and smaller shortest path trees (SPT) rooted at the sink that span nodes possessing all data, i.e., in the current iteration, the SPT rooted at the sink spans a set of nodes that possess all data aggregated from S till the previous iteration. The current iteration produces a collision-free schedule that comprises a set of simultaneously transmitting senders, which are selected from the leaves of the SPT based on the number of non-leaf neighbors in the graph, and a set of corresponding receivers.

Malhotra *et al.* [101] consider the joint routing and scheduling problem for aggregated convergecast with the goal to construct an optimal routing tree that will help minimizing the schedule length. The basic idea of the tree construction algorithm is to create an SPT and balance the number of children per node so that more parallel transmissions can take place without any single node causing a bottleneck. They show that for given a tree, a lower bound on the schedule length is $\max_{i \in V} (\xi_i + h_i)$, where ξ_i and h_i are the number of children and hop distance from the sink, respectively, for node i . To balance the number of children per node, an optimal *semi-matching* formulation on bipartite graphs, originally due to Harvey *et al.* [62], is used with the goal to assign nodes from level $h + 1$ to the parents at level h , such that every parent has an equal number of children. Once the balanced tree is constructed, a ranking-based heuristic is used for scheduling, which ranks all eligible nodes in decreasing order of their weights, taken as

the number of *non-leaf neighbors*. A higher weight gives a higher relative priority to a node to be scheduled in the current slot over other eligible nodes.

A variation of the aggregated convergecast problem where nodes can adjust their transmission ranges is studied by Shang *et al.* [133]. They propose an approximation algorithm that gives a constant factor guarantee on the optimal schedule length for UDG. It first constructs a BFS tree rooted at the sink, and then constructs a *maximal independent set* using the greedy First-Fit algorithm by choosing nodes in order of their increasing hop distances from the sink. This results in a *dominating set*, which contains the sink but is not necessarily connected. Then, a minimal number of connector nodes is added to construct a *connected dominating set*, V_{CDS} , and the transmission ranges of all the nodes in this set are set to one. Next, the scheduling phase runs in two stages. In the first stage, nodes in $V \setminus V_{CDS}$ are scheduled first so that all their data reach the nodes in V_{CDS} . In the second stage, data are sent from the nodes in V_{CDS} to the sink. It is shown that the first stage takes $15 \log_2 |V \setminus V_{CDS}|$ time slots, whereas the second stage takes $16d(T_{BFS}) - 12$ time slots, where $d(T_{BFS})$ is the depth of the BFS tree. Combining the two, it is shown that the schedule length is at most 31 times the optimal.

Zhang *et al.* extend their prior work on minimal time convergecast scheduling for raw data convergecast [46] to aggregated convergecast scheduling in [163]. When the size of data is much smaller than the size of the data frame, nodes aggregate the received packets instead of sending single packets. They use the same scheduling algorithm proposed in [46,47], and show that packet aggregation requires at most $N + 2$ time slots in a linear

network. According to the algorithm, the sink receives the first packet in the first time slot; then in every 3 time slots it receives an aggregated packet which contains data from 3 original unaggregated packets due to 2-hop scheduling to prevent interference. Hence, the total number of required time slots is $1 + 3 \lceil \frac{N-1}{3} \rceil \leq N + 2$. For multi-line networks, the algorithm achieves a schedule length of $\max \left(n_k + 3 \lceil \frac{N+2k}{3} \rceil \right)$, where k is the number of branches and n_k is the maximum number of nodes in a branch. Finally, for tree networks aggregation-enabled convergecast requires $\max \left(\hat{n}, \lceil \frac{N+2L+2(L-k)}{3} \rceil \right)$, where L is the number of leaf nodes, k is the number of one-hop subtrees, $\hat{n} = \max_i (n_i + 4l_i - 2)$, n_i is the number of nodes, and l_i is the number of leaf nodes in the i^{th} one-hop-subtree.

2.1.2 Multi-Channel Scheduling

The use of multi-channel communication is a well-studied research topic in wireless ad hoc networks and cellular networks. In cellular networks [79], base stations use different frequency domains within a cell, while clients share the time domain to access the wireless medium. However, this approach is either single-hop or infrastructure based, and thus is not suitable for multi-hop networks that are deployed over large geographical regions.

For multi-hop ad hoc networks, there exist several works that aim to increase the system throughput [74, 128, 135]. Most of these approaches are based on the IEEE 802.11 protocols. For instance, IEEE 802.11b allows 11 channels that are spaced 5 MHz apart.

However, the IEEE 802.11 protocols are very expensive in terms of energy consumption and do not meet the requirements of WSN. Kyasanur *et al.* [89] study the capacity of multi-channel wireless ad hoc networks by extending the analysis of Gupta and Kumar [59], which shows that the asymptotic achievable throughput per node for a randomly distributed set of N source-destination pairs with one-to-one communication is bounded by $\Theta\left(\frac{W}{\sqrt{N \log N}}\right)$, where W is the transmission capacity. They investigate the impact of multiple channels and the number of radio interfaces on the network capacity, and show that, even with smaller number of interfaces than that of available channels, multi-channel communication can enhance the networks capacity.

In the domain of sensor networks, research using multiple channels is relatively new due to their several differences with ad hoc networks, such as simpler radios, bandwidth and energy limitations, scalability, etc. Although the focus of this thesis is to utilize multiple channels to alleviate the impact of interference and improve network performance in terms of throughput and delay, we first briefly discuss a few important studies that use multiple channels to achieve other objectives.

The use of multi-channel communication against jamming is discussed in [4, 152, 157], where channel surfing mechanisms have been introduced such that the jammed nodes dynamically change their operating frequency. Multi-channel clustering is discussed in [58] where nodes that hold correlated data are clustered together and communicate on the same frequency, which is different from the communication frequency of other clusters. Cluster heads are assumed to be the aggregation points to process raw

data before relaying toward the sink node. The key objective in this work is to minimize energy consumption by reducing the effects of collisions that may occur if the clusters operate on the same frequency. Multi-channel communication is also used in reliable data dissemination [95, 155]. The Typhoon protocol [95] uses channel switching to reduce contention in the broadcast medium which, in turn, reduces the completion time of data dissemination. Multi-channel communication can also be used to overcome the congestion that can occur due to contention and interference in the network. Examples of joint channel assignment and congestion control do exist in wireless ad hoc networks [54, 110]. However, in WSN, congestion avoidance with multiple channels is very briefly addressed in [154].

Just as there exist different channel assignment strategies for ad hoc networks, such as fixed, semi-dynamic, and dynamic, as discussed in Chapter 1, similar assignments also exist in sensor networks. Fixed channel assignment, where nodes are clustered into different frequencies, are presented in [18, 58, 154]. The IEEE 802.15.4 standard also uses fixed channel assignment, but it is possible for the beacon node to change the operating frequency if other nodes report excessive interference. In [154], it is argued that frequent channel switching may cause potential packet losses, however, once channel switching is done synchronously, this overhead can be eliminated. Examples of semi-dynamic assignment, where nodes are assigned fixed channels but can switch in order to communicate with other nodes, are presented in [99, 132]. Y-mac [81] is the first example that uses dynamic channel assignment in WSN, where a combination of a dedicated control

channel and a frequency hopping method is used. In [148], Voigt extends the D-MAC protocol [97] and proposes to use multi-channel communication to reduce interference.

2.1.2.1 Maximizing Raw-Data Convergecast Throughput

Multiple channels to eliminate interference and increase network throughput has been studied by Incel *et al.* in [69]. They explore and evaluate a number of different techniques using realistic simulation models to study the data collection rate for raw-data convergecast. First, a simple spatial-reuse TDMA scheme is employed to minimize the schedule length, which is then combined with multiple frequency channels and transmission power control to achieve further improvement. A receiver-based channel assignment (RBCA) scheme is proposed where the receivers of the tree are statically assigned a channel, and the children of a common receiver transmit on that channel. This avoids pair-wise, per-packet channel negotiation overheads. Once multiple frequencies are used to completely eliminate interference (i.e., secondary conflicts), it is shown that the lower bound on the schedule length is $\max(2n_k - 1, N)$, and a time slot assignment scheme is proposed that achieves this bound with no nodes requiring to buffer more than one packet at any time. Here, as in [47], n_k is the maximum number of nodes in any branch of the tree. Next, the authors also show that once interference is eliminated, the data collection rate often becomes limited by the routing topology. To overcome this, trees with specific properties are constructed, which help in further enhancing the data collection rate. In particular, *capacitated minimum spanning trees* [117], which aim to have an equal

Algorithm 1 LOCAL-TIMESLOTASSIGNMENT

1. $node.buffer = full$
 2. **if** $node$ is sink **then**
 3. Among all eligible top-subtrees, choose one with the largest number of remaining packets, say top-subtree i
 4. Schedule link $(root(i), s)$ respecting interfering constraint
 5. **else**
 6. **if** $node.buffer == empty$ **then**
 7. Choose a random child c of $node$ whose buffer is $full$
 8. Schedule link $(c, node)$ respecting interfering constraint
 9. $c.buffer = empty$
 10. $node.buffer = full$
 11. **end if**
 12. **end if**
-

number of nodes on each branch, are shown to achieve a factor of two improvement as compared to single-channel TDMA scheduling on minimum-hop SPTs.

The key idea behind the algorithm in [69], which is formally presented in Algorithm 1 as LOCAL-TIMESLOTASSIGNMENT, is to: (i) schedule transmissions in parallel along multiple branches of the tree, and (ii) keep the sink busy in receiving packets for as many time slots as possible. Each node maintains a buffer and its associated state, which can either be *full* or *empty* depending on whether it contains a packet or not. Initially, all the buffers are full because every node has a packet to send.

The first block of the algorithm in lines 2-4 gives the scheduling rules between the sink and the roots of the top-subtrees. A *top-subtree* $TS(r)$ is defined as one whose root r is a child of the sink, and it is said to be *eligible* if r has at least one packet to send. For instance, in Figure 2.3(a), the top-subtrees are $\{1, 4\}$, $\{2, 5, 6\}$, and $\{3, 7\}$. For a given time slot, the root of an eligible top-subtree which has the *largest* number of total remaining packets is scheduled. If none of the top-subtrees are eligible, the sink does not

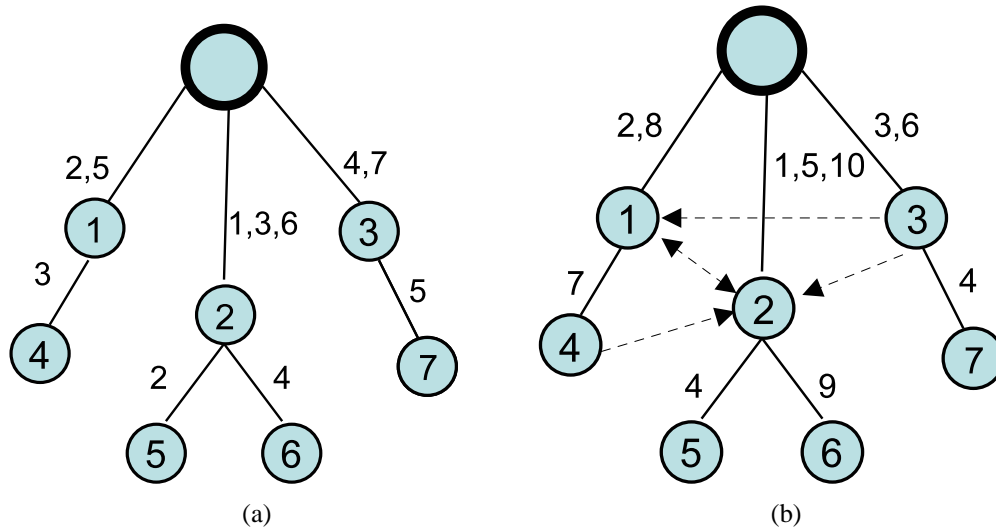


Figure 2.3: Raw-data convergecast using algorithm LOCAL-TIMESLOTASSIGNMENT: (a) Schedule length 7 when secondary conflicts are eliminated. (b) Schedule length 10 when secondary conflicts are present.

receive any packet during that time slot. Inside each top-subtree, nodes are scheduled according to the rules in lines 5-12. A subtree is defined to be *active* if there are still packets left in it (excluding its root) to be relayed. If a node's buffer is empty and the subtree rooted at this node is active, one of its children is scheduled at random whose buffer is not empty. The algorithm guarantees that in an active subtree there will always be at least one child whose buffer is not empty, and so whenever a node empties its buffer, it will receive a packet in the next time slot, thus emptying buffers from the bottom of the subtree to the top.

Figure 2.3(a) shows an illustration of the working of the algorithm. In slot 1, since the eligible top-subtree containing the largest number of remaining packets is $\{2, 5, 6\}$, link $(2, s)$ is scheduled and the sink receives a packet from node 2. In slot 2, the eligible top-subtrees are $\{1, 4\}$ and $\{3, 7\}$, both of which have 2 remaining packets. We choose

one of them at random, say $\{1, 4\}$, and schedule the link $(1, s)$. Also, in the same time slot since node 2's buffer is empty, it chooses one of its children at random, say node 5, and schedule the link $(5, 2)$. In slot 3, the eligible top-subtrees are $\{2, 5, 6\}$ and $\{3, 7\}$, both of which have 2 remaining packets. We choose the first one at random and schedule the link $(2, s)$, and so the sink receives a packet from node 5 (relayed by node 2). We also schedule the link $(4, 1)$ in slot 3 because node 1's buffer is empty at this point. This process continues until all the packets are delivered to the sink, yielding an assignment that requires 7 time slots. Note that, in this example, $2n_k - 1 = 5$, and so $\max(2n_k - 1, N) = 7$. In Figure 2.3(b), an assignment is shown when all the interfering links are present, yielding a schedule length of 10.

A similar result of $\max(2n_k - 1, N)$ is obtained by Song *et al.* [138] where they also extended it to the case when nodes have different number of packets to send. Multiple channels that minimize the schedule length for raw-data convergecast in WirelessHART networks [137] is studied by Zhang *et al.* [161]. One significant difference between WirelessHART networks and sensor networks is that the former performs channel hopping on a per-packet basis, while most existing TDMA convergecast schemes do not support this feature. Thus, parallel transmissions scheduled in the same time slot must use different channels, whereas most of the existing TDMA-based multi-channel protocols first statically assign channels to eliminate potential interference and then perform time slot scheduling. Like in [69] and [138], they also consider buffer requirements at each node, and show that when nodes can store at most one packet, the minimum schedule length

for line topologies is $2N - 1$ using at most $\lceil N/2 \rceil$ channels. However, when the nodes can buffer multiple packets, the optimal convergecast time remains the same while the number of channels required can be reduced to $\lceil N - \sqrt{N(N-1)/2} \rceil$. The basic idea of their proposed approach, which is similar to [69] and [138], is to schedule as many transmissions as possible in each time slot in order to maximize the use of available channels, and to make sure that a node which does not have a packet at the beginning of a time slot receives one packet at the beginning of the next time slot.

2.2 Spanning Trees for Fast Data Collection

Several optimization problems arising in the design of communication networks can be modeled as constructing optimal network topologies [164], in particular, spanning trees. Spanning trees are widely used in communication networks as a means to disseminate information from one node to all other nodes, as in broadcast or multi-cast, and/or to collect information from a set of sources to a single designated node, as in convergecast. Depending on the constraints, the problem of computing spanning trees has been characterized by different complexity classes. Perhaps the simplest among all these is the problem of constructing a minimum cost spanning tree in an edge weighted graph, which is efficiently solvable in polynomial time for both sequential and parallel cases. However, when simple constraints are imposed on the structure of the resulting tree, such as bounded node degree or diameter, the problem frequently becomes NP-hard.

In this thesis, we focus on computing spanning trees that help in improving the throughput-delay trade-off for fast data collection in large-scale sensor networks. In [70], Incel *et al.* show that, in addition to spatial-reuse TDMA scheduling using multiple frequency channels, which helps to eliminate interference and enable more concurrent transmissions, thereby enhancing the data collection rate, the structure of the routing tree also plays an important role in the performance of convergecast throughput. In particular, routing trees with high node degree are likely to create bottlenecks within the network, because the children of a common parent need to be scheduled at different time slots due to half-duplex transceivers. On the other hand, trees with low node degree might allow for more concurrent transmissions in the presence of multiple frequencies. Thus, if we aim to maximize the convergecast sink throughput and have the flexibility to construct a routing tree, computing one with minimum degree might be a good choice. We note that, sometimes, however, the routing tree on which a given network should operate might be fixed and specified a priori due to socio-economic constraints, in which case we do not have the flexibility to construct one that is best suited for the objectives we want to optimize.

It is also true that for a given deployment of nodes, a spanning tree with low node degree also has a large number of hops to the sink. Thus, if packet delays are measured purely in terms of hop counts, a tree with low node degree is likely to incur high delays as opposed to one with high node degree. These two opposing tree properties – *node degree* and *hop distance* – therefore, underscore the role of the routing topology in

maximizing the data collection rate and minimizing packet delays. In the following, we describe some of the existing works that aim to construct spanning trees with different structural constraints so as to optimize multiple conflicting objective functions. These works fall under the general theme of *multi-criteria network design problems* [125], and in particular, bicriteria problems when the number objectives is two.

2.2.1 Bicriteria Network Design Problems

A good example of a network design problem that calls for a multi-criteria optimization framework is multicast. Designing networks that are capable of accommodating multimedia (both audio and video) traffic in a multicast (simultaneous transmission of data to multiple destinations) environment has gained considerable interest in recent years. One of the popular solutions to multicast routing, as discussed by Kompella *et al.* [83], involves tree construction. Two optimization criteria – (i) worst-case transmission delay, and (ii) total cost – are typically sought to be minimized in constructing these trees. In addition, it is often the case that these multiple optimization criteria also have different cost functions and budget associated with each measure. For example, as pointed out in [83], in the problem of finding good multicast trees, each edge has two costs associated with it: a construction cost and a delay cost. The construction cost is typically a measure of the amount of buffer or channel bandwidth used, and the delay cost is a combination of propagation, transmission, and queuing delays. Clearly, these costs have associated budgets under which they should operate. Such multi-criteria network design

problems with separate cost functions for each optimization criterion also occur naturally in information retrieval and VLSI design where one aims to find minimum cost spanning or Steiner trees given delay bound constraints on source-sink connections.

Network design problems where even one cost measure needs to be minimized are often NP-hard, with the exception of a few, such as the *Minimum Diameter Spanning Tree* problem. Here the goal is to construct a spanning tree such that the tree diameter, defined as the longest hop distance between any pair of nodes, is minimized. On Euclidean graphs, this problem is solved in polynomial time $\Theta(n^3)$, and the result extends to any complete graph whose edge weights satisfy a distance metric [67]. The most recent result on general graphs is obtained by Hassin and Tamir [65] by showing its equivalence with the *Absolute 1-Center* (A1CP) problem that runs in $O(mn + n^2 \log n)$ time, where m is the number of edges. Note that, this reduces to the $\Theta(n^3)$ bound for complete graphs because $m = n^2$ for complete graphs.

Most notably among the NP-hard unicriterion optimization problems is the *Minimum Degree Spanning Tree* (MDST) problem, where the goal is to construct a spanning tree of a graph $G = (V, E)$ with n vertices, such that the maximal degree is smallest among all spanning trees of G . This problem is a generalization of the Hamiltonian Path Problem and is NP-hard, thus leaving us hope only in the space of approximation algorithms. Suppose T^* be an optimal spanning tree whose maximal degree is Δ^* . Furer and Raghavachari [44] propose an iterative polynomial time approximation algorithm that computes a spanning tree of maximal degree at most $O(\Delta^* + \log n)$. This result is then

refined by the same authors to produce a spanning tree of degree at most $\Delta^* + 1$, which is the best bound achievable in polynomial time, unless $P = NP$. In the Steiner version of the problem, along with the input graph, we are also given a distinguished set of vertices $D \subseteq V$, and the goal is to compute a tree of minimum degree spanning all the nodes in D . The *Minimum Degree Steiner Tree* problem [45] is more general and the MDST problem is its special case when $D = V$. Agrawal *et al.* [82] have shown that the Steiner version can be approximated to within a $\log n$ factor using multicommodity flow.

Taking a step further from unicriterion optimization problems, such as the MDST, we now move on to describe some of the existing works that consider bicriteria network design problems. As one could expect, such problems are also NP-hard and render us in the domain of approximation algorithms.

Definition 1. *A generic bicriteria optimization problem is defined as $\mathcal{P} = (\mathcal{X}, \mathcal{Y}, \mathcal{H})$, where \mathcal{X} and \mathcal{Y} are the two objectives, and \mathcal{H} represents a membership requirement in the class of subgraphs. The problem \mathcal{P} specifies a given budget on the first objective \mathcal{X} under a cost function $c_{\mathcal{X}}$, and the goal is to find a network topology that minimizes the second objective \mathcal{Y} under possibly a different cost function $c_{\mathcal{Y}}$, such that the network is within the budget of the first objective.*

An example of a bicriteria optimization problem is that of finding a spanning tree that yields low-cost and low-transmission-delay in multimedia networks for multicast, which can be modeled as *(Diameter, Total Cost, Spanning Tree)*-bicriteria problem as follows. Given an undirected graph $G = (V, E)$ with two cost functions c and d for each edge $e \in$

E modeling construction and delay costs, respectively, and a bound \mathcal{D} on the total delay, find a minimum c -cost spanning tree such that the diameter of the tree under the d -cost is at most \mathcal{D} . It is easy to see that the notion of bicriteria problems can easily be extended to the more general multi-criteria optimization problems. We note that such optimization problems that involve minimizing two cost measures have also been addressed in the literature by attempting to minimize a functional combination of the two, thus converting them into unicriterion problems, however, this approach fails when the two criteria are very disparate. It is shown by Marathe *et al.* [103] that bicriteria formulation is quite generic and robust because the quality of approximation is independent of which of the two criteria the budget is imposed on, and it subsumes the case where one wishes to optimize a functional combination of the two objectives.

Recall that an approximation algorithm for a unicriterion optimization problem provides a performance guarantee ρ if for every instance, the value of the solution returned by the algorithm is within a factor ρ of the optimal value for that instance. This notion of performance guarantee can be extended to bicriteria optimization problems as follows. An (α, β) approximation algorithm for a bicriteria minimization problem $(\mathcal{X}, \mathcal{Y}, \mathcal{H})$ is defined as a polynomial time algorithm that produces a solution in which the value of the first objective \mathcal{X} is at most α times the budget, and the value of the second objective \mathcal{Y} is at most β times the minimum for any solution that is within the budget of \mathcal{X} . The solution produced must also belong to the subgraph-class \mathcal{H} . Analogous definitions can be given when \mathcal{X} and/or \mathcal{Y} are maximization objectives. In [103], Marathe *et al.* show that

an (α, β) approximation algorithm for a problem $(\mathcal{X}, \mathcal{Y}, \mathcal{H})$ can be easily transformed in polynomial time into a (β, α) approximation algorithm for problem $(\mathcal{Y}, \mathcal{X}, \mathcal{H})$, and so the definition of a bicriteria approximation is independent of the choice of the criterion that is budgeted in the formulation.

The *(Diameter, Total Cost, Spanning Tree)* problem, also known as the *Bounded Diameter Minimum Spanning Tree* problem is NP-hard even in the special case when the two cost functions are identical [67]. Awerbuch *et al.* [9] gave an approximation algorithm with $(O(1), O(1))$ performance guarantee for finding a spanning tree that has small diameter as well as small cost, both under the same cost function. Khuller *et al.* [80] studied an extension called *Light, Approximate Shortest-path Tree (LAST)* and gave an approximation algorithm with $(O(1), O(1))$ performance guarantee. Kadaba and Jaffe in [16] and Kompella *et al.* in [83] considered the *(Diameter, Total Cost, Steiner Tree)* problem with two edge costs and presented heuristics without any guarantees. A closely related problem is the *(Diameter, Total Cost, s-t Path)* problem, which requires finding a diameter-constrained shortest path between two pre-specified vertices s and t . This problem, termed in the literature as the *Multi-Objective Shortest Path (MOSP)* problem, is NP-complete, and Warburton [149] presented the first fully polynomial time approximation scheme (FPTAS) for it. Hassin [64] also provided a strongly polynomial FPTAS for this problem which improved the running time of Warburton.

More general than the MDST problem is the *Bounded Degree Minimum Spanning Tree* problem *(Degree, Total Cost, Spanning Tree)* where, in addition to a degree bound Δ

on each node, the total cost of the tree also needs to be minimized. Ravi *et al.* [125] propose an approximation algorithm using a matching-based augmentation technique that gives a $(O(\Delta \log \frac{n}{\Delta}), O(\log \frac{n}{\Delta}))$ guarantee. Konemann and Ravi in [85] use a Lagrangian-relaxation based approach to propose an improved $(O(\Delta + \log n), O(1))$ approximation algorithm. Chaudhuri *et al.* presented an improved $(1, O(\Delta + \log n))$ approximation algorithm in [23], and a further improved version that gives $(1, O(\Delta))$ performance guarantee in [22] based on push relabel framework. Their algorithm is the first one that approximates both degree and cost within a constant factor. The best possible result is proposed recently by Singh and Lau in [134] where the approximation guarantee is $(1, \Delta + 1)$. In addition to the upper bound Δ on the node degree, when there is also a lower bound Δ' , their algorithm returns a spanning tree where the degree of each node v satisfies the constraint $\Delta' - 1 < \deg(v) < \Delta + 1$ with at most the optimal total cost. This result is also true for general cost functions (even negative).

Most related to our work is the *Bounded Degree Minimum Diameter Spanning Tree* problem (*Degree, Diameter, Spanning Tree*), where the goal is to find a spanning tree such that the diameter of the resulting tree is minimized subject to the constraint that the degree of each node is no more than a given integer Δ . This problem is NP-hard for general graphs. Ravi first proposed a $(O(\log^2 n), O(\log n))$ approximation algorithm in [124] that runs in $O(nm \log n)$ time and finds a spanning tree of degree $O(\Delta^* \log n + \log^2 n)$ and diameter $O(\mathcal{D} \log n)$, where Δ^* is the optimal degree of any spanning tree of diameter at most \mathcal{D} , whose value is greater than the diameter of the graph. They use

the notion of *poise* of a tree, defined as the sum of the maximum degree and diameter, and use multicommodity flow results, in particular, a result on the L-bounded minimum congestion problem, to prove the approximation guarantee. A further improvement is obtained by Konemann *et al.* [84] who propose a $O(\sqrt{\log_{\Delta^*} n})$ approximation algorithm for complete graphs under a metric cost function. Their algorithm uses a combination of filtering and divide and conquer techniques to find a spanning tree of maximum node degree Δ and diameter $O(\sqrt{\log_{\Delta} n} \cdot \mathcal{D}^*)$, where \mathcal{D}^* is the minimum diameter of all spanning trees with degree at most Δ .

2.3 Topology Control in Wireless Networks

2.3.1 Why Topology Control?

The *topology* of a packet radio network (PRN), and in particular, wireless multihop network is a set of communication links between node pairs that describe the connectivity information of the network, and is used explicitly or implicitly by a routing mechanism. Unlike the links in a wired network, which are well defined by modems and hard wires, links in a wireless network are the result of some link determination protocol subject to a transmission power limitation. A basic requirement for a link to exist between any two nodes in a topology is that the nodes are within each other's transmission range. In most of the early works, the topologies of a PRN is defined by connecting all neighbors within a fixed transmission range, which is uniform for all nodes.

In order to maintain good connectivity and reliability of the network, this fixed, uniform transmission range must be adjusted to be large enough, especially in sparse regions. However, setting a large, uniform transmission range will result in high node degrees in denser regions of a network, which will prevent spatial reuse of the wireless channel, thus causing congestion and reducing network throughput. Additionally, it might also cause long packet delays. Thus, reliability and throughput are conflicting requirements. For instance, the most reliable topology is a complete graph where every node is connected to every other node, but it is the worst case for a shared radio channel in terms of throughput because no reuse is possible. On the other hand, setting a low transmission range might partition the network.

One solution to the above problem associated with dense networks is *topology control*, in which every node can carefully select a set of neighbors to establish logical data links, and then dynamically adjust its transmission range for different links by adjusting its transmission power. In establishing these logical links, it is desirable to choose only those local connections that will guarantee overall global network connectivity while satisfying different and often conflicting performance metrics, such as overall throughput, network utilization, and power dissipation. In a shared radio channel, a higher transmit power results in more forward progress (i.e., the distance by which a packet has moved in the direction toward its destination) per transmission, while a lower transmit power causes less interference to distant nodes. Theoretically, selecting a proper transmit power per node can result in optimal performance. In addition, controlling the

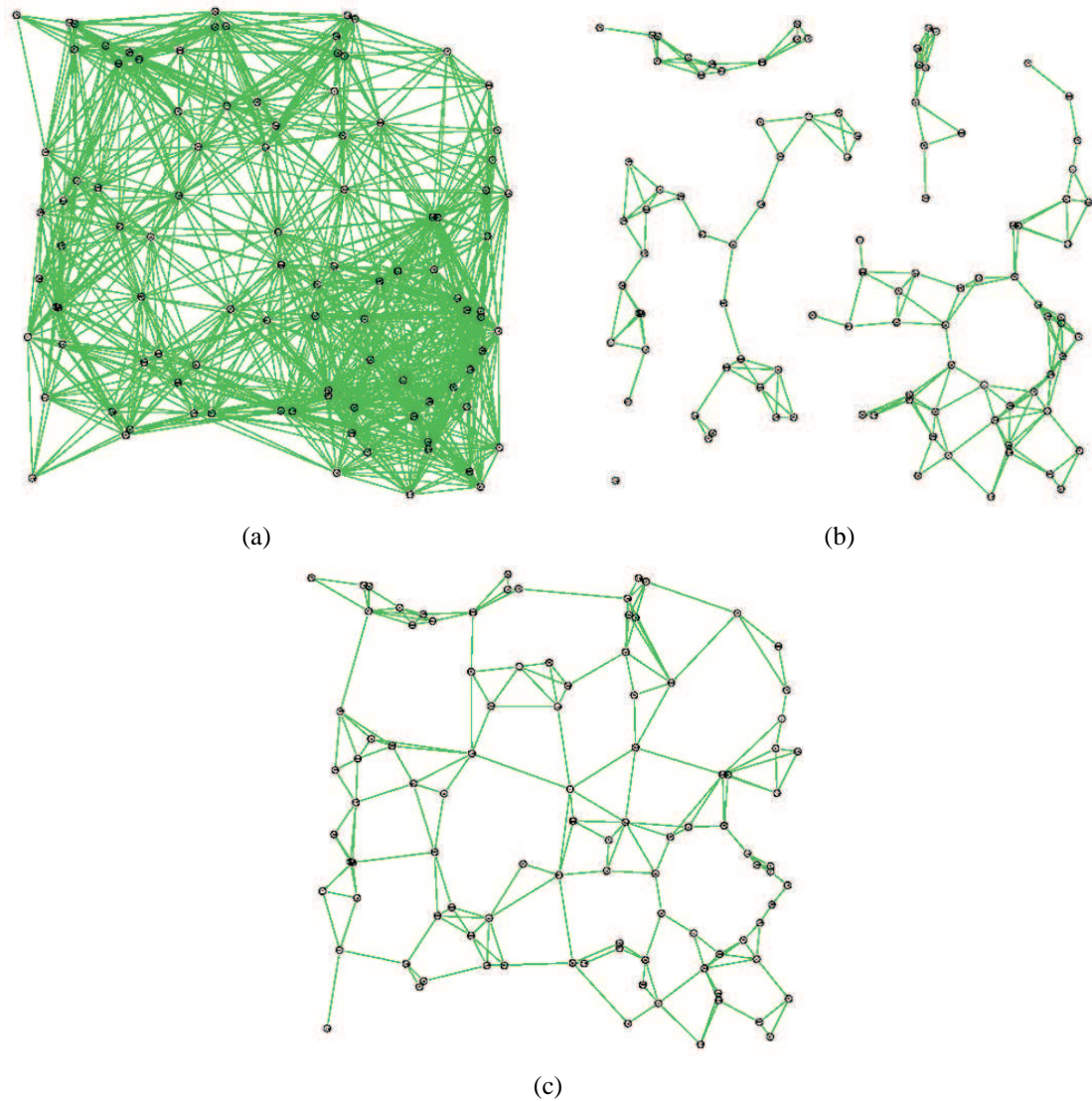


Figure 2.4: Benefits of topology control: (a) Network without topology control in which nodes are configured at maximum transmission range leading to high node degree. (b) Network without topology control in which nodes are configured at minimum transmission range leading to network partition. (c) Network with topology control in effect in which nodes adjust their transmission range leading to low average node degree and yet a connected network.

transmission power helps in extending network lifetime, which is a crucial requirement for wireless multihop networks, and in particular, resource constrained sensor networks. Figure 2.4(a) and 2.4(b) show examples of two networks without topology control in which nodes are configured at maximum and minimum transmission range, respectively, leading to high node degree and network partition. Figure 2.4(c) shows the same network with topology control in effect where nodes could adjust their transmission range leading to low average node degree and yet a connected network. A list of good properties for topology control in order to maximize network throughput is the following: (i) each node should have a similar number of neighbors, and the average node degree should be small, (ii) nodes that are geographically closer should have higher priority of being logical neighbors to a given node, (iii) regular and uniform structures are usually preferred, and (iv) it should be difficult to partition the network by removing only a few links.

In this thesis, we consider topology control for arbitrarily deployed networks with the goal to establish *global* connectivity by choosing only a minimal set of neighbors using *local* information. In particular, we consider 3-dimensional (3-D) networks where the density of nodes required to maintain connectivity in typical deployments is very high compared to 2-dimensional (2-D) networks. Although topology control in the domain of 2-D networks is a well researched topic, the number of such works in 3-D networks is only a few and they require high computational overhead. To this end, our contribution in this thesis is to develop an efficient distributed topology control protocol for 3-D

networks that has low computational complexity. In the following, we describe some relevant literature for 2-D as well as 3-D networks.

One of the prominent works that consider 2-D networks is by Wattenhofer *et al.* [93, 150], in which the authors propose a novel distributed *Cone-Based Topology Control* (CBTC) algorithm that increases network lifetime while maintaining global connectivity with reasonable throughput in multihop wireless ad hoc networks. In contrast to earlier approaches that rely on knowing and sharing global positioning information of all nodes, CBTC is a distributed algorithm that relies solely on local directional information of incoming signals from neighboring nodes.

Chapter 3

Maximizing Convergecast Throughput in Tree-Based Sensor Networks

3.1 Motivation

Convergecast, namely the *many-to-one* flow of data from a set of sources to a common sink over a tree-based routing topology, is a fundamental communication primitive in sensor networks . Depending on the specific application, such data collection can be triggered by external events, such as user queries, to periodically get a snapshot view of the network, or can be automated internally sustaining over long durations. For real-time, mission-critical, and high data-rate applications, such as security-surveillance, health-care, structural health monitoring [156], as well as some environmental networks, such as wildfire [141] and permafrost monitoring [63] [15], it is often critical to maximize the data collection rate at the sink node. In addition, when the sensor readings are correlated

Parts of this chapter are based on [50], [69], and [68].

due to spatial/temporal proximity of the nodes, or when summarized information is required, it is beneficial to aggregate data en route to the sink. Data aggregation helps in reducing redundancy and the number of transmissions, thus saving energy [98]. In such cases, we refer to the data collection process as *aggregated convergecast* [70].

In this chapter, we consider the problem of maximizing the aggregated convergecast throughput at the sink node by utilizing multiple frequencies using TDMA scheduling [102]. We consider two cases with respect to our knowledge about the routing topology: (i) when the topology is known a priori, and (ii) when the topology is unknown and can be arbitrary. We show that multiple frequencies can help in eliminating interfering links and enable more concurrent transmissions, thereby increasing the aggregated data collection rate. We prove NP-completeness results on the multi-channel scheduling problem, and design efficient link scheduling algorithms that have provably good, worst-case performance guarantee for *arbitrarily* deployed networks in 2-D. In particular, when the routing topology is known a priori, we design a constant factor approximation algorithm for networks modeled as Unit Disk Graphs (UDG) where every node has a uniform transmission range, and a $O(\Delta(T) \log N)$ -approximation for general disk graphs where nodes could have different transmission ranges. Here N is the total number of nodes in the network, and $\Delta(T)$ is the maximum node degree on a given routing tree T . We also prove that a constant factor approximation is achievable on UDG even when the routing topology is unknown, so long as the maximum in-degree of any node in the tree is

bounded by a constant. We evaluate our proposed algorithms using MATLAB simulation and show trends in the scheduling performance for different network parameters.

The rest of the chapter is organized as follows. In Section 3.2, we describe our models, assumptions, and problem formulation. In Section 3.3, we consider the multi-channel scheduling problem for *general graphs* (where links could exist between any pair of nodes), and prove two NP-completeness results. In the same section, we also design a polynomial-time algorithm to schedule the whole network in minimum time when there are enough frequencies available to remove *all* the secondary conflicts. In Section 3.4, we design approximation algorithms for the scheduling problem on a UDG model under a given limited number of frequencies for the two cases of known a priori and unknown routing topologies. In Section 3.5, we present an Integer Linear Programming (ILP) formulation of the scheduling problem for the general disk graph model, and propose an approximation algorithm. In Section 3.6, we present our evaluation results, and finally in Section 3.7, we summarize the chapter.

3.2 Preliminaries

3.2.1 Model and Assumptions

We model the network as an undirected graph $G = (V, E)$, where V is the set of nodes arbitrarily deployed in the 2-D plane, and E is the set of edges. An edge $e = (u, v) \in E$ exists between any two nodes u and v if their Euclidean distance $d(u, v)$ is no more than

the transmission range R . We assume G to be connected, i.e., there is no isolated node . We are also given a distinguished node $s \in V$ that represents the sink. We denote a spanning tree on G rooted at s by $T = (V, E_T)$, where $E_T \subseteq E$ is the set of tree edges that needs to be scheduled.

Each node is equipped with a half-duplex transceiver using which it can either transmit or receive a single packet at any given time slot. Consecutive slots are grouped into equal sized frames that are repeated for periodic scheduling. We assume that every node generates a single packet in the beginning of every frame, and has the ability to aggregate all the packets from its children as well as its own into a single packet before transmitting to its parent. The class of aggregation functions that falls in this category include *distributive* and *algebraic* functions [98], where the size of an aggregated packet is constant regardless of the size of the raw sensor readings. Examples of such functions are MIN, MAX, MEDIAN, COUNT, SUM, AVERAGE, etc.

We assume that transmissions on different frequencies are orthogonal and non-interfering with each other. This assumption may sometimes fail in practice depending on transceiver-specific adjacent channel rejection values, however, experimental results presented by Incel *et al.* [70] show that the scheduling performance remains similar for CC2420 and Nordic nrf905 radios. We consider a *receiver-based* frequency assignment strategy, in which we assign a frequency statically to each of the receivers (parents) in the tree and

If there are isolated nodes, or more than one connected components in the network, then we consider only those nodes that have a path to the sink.

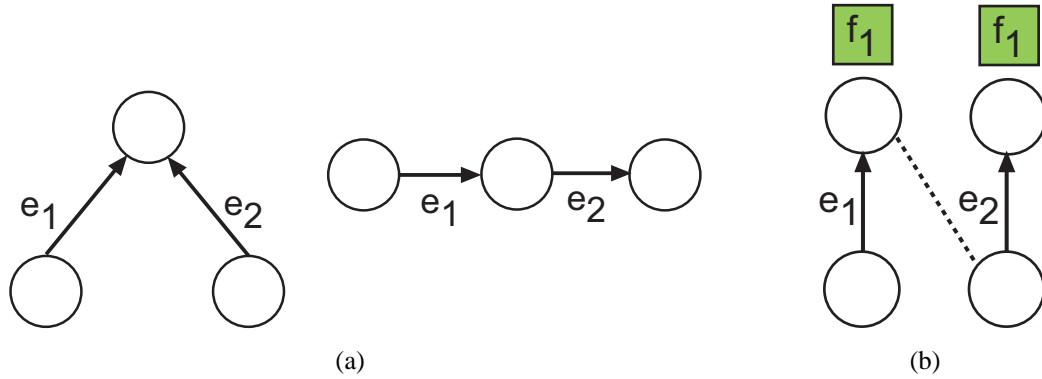
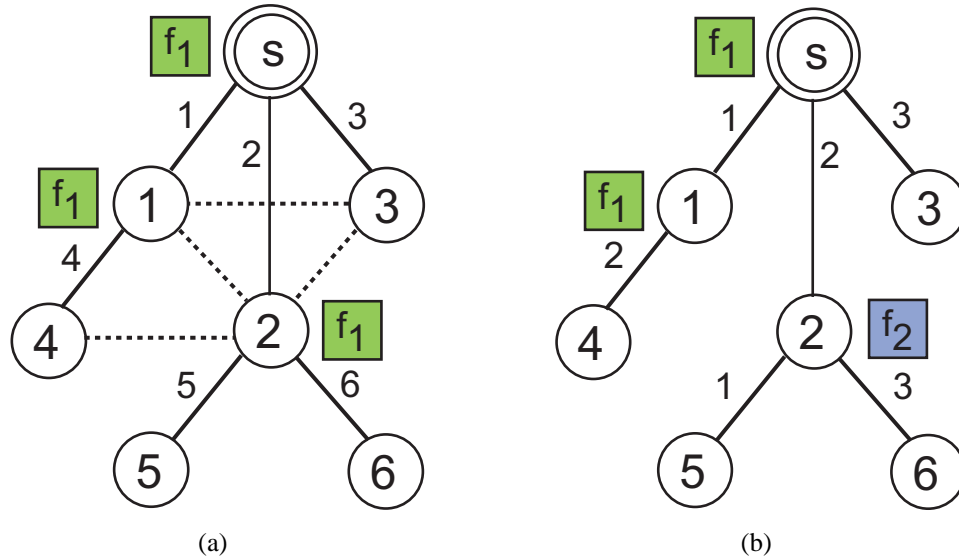


Figure 3.1: (a) Concurrent transmissions on adjacent edges e_1 and e_2 cause primary conflict. (b) Concurrent transmissions on edges e_1 and e_2 cause secondary conflict if their receivers are on the same frequency, say f_1 , and if either of the receivers is within the range of the other transmitter.

have the children transmit on the same frequency assigned to their parent. Due to this particular static assignment scheme, each node operates on at most two frequencies, which results in less overhead compared to dynamic frequency assignment schemes, such as pair-wise negotiation of frequencies on a per-packet basis.

We consider a graph-based interference model (also known as the *protocol model*), in which (a) the interference range of a node is equal to its transmission range R , and (b) concurrent transmissions on two tree edges $e_1, e_2 \in E_T$ interfere with each other if and only if either (i) they are adjacent, or (ii) both their receivers are on the same frequency and at least one of the receivers is within the range of the other transmitter. The first type of interference is known as *primary conflict*, and the second type as *secondary conflict*, as illustrated in Figure 3.1(a) and 3.1(b), respectively.



	Frame 1						Frame 2					
	1	2	3	4	5	6	1	2	3	4	5	6
s	1	2	3	-	-	-	(1,4)	(2,5,6)	3	-	-	-
1	-	-	-	4	-	-	-	-	-	4	-	-
2	-	-	-	-	5	6	-	-	-	-	5	6

(c)

	Frame 1			Frame 2		
	1	2	3	1	2	3
s	1	(2,5)	3	(1,4)	(2,5,6)	3
1	-	4	-	-	4	-
2	5	-	6	5	-	6

(d)

Figure 3.2: Aggregated convergecast: (a) Schedule length of 6 time slots using one frequency. Dotted lines represent secondary conflicts. (b) Schedule length of 3 time slots using two frequencies. Note that, all the secondary conflicts are eliminated. (c), (d) Nodes from which aggregated data is received by their corresponding parents in each time slot over 2 consecutive frames for (a) and (b), respectively.

3.2.2 Problem Formulation

We first explain the process of aggregated convergecast, and then define the multi-channel scheduling problem. Figure 3.2(a) shows a network of 6 source nodes and a routing tree whose edges are marked by solid lines; dotted lines represent interfering links causing secondary conflicts. We also show a possible time slot assignment (optimal in this case) represented by the numbers beside each of the edges in which they are scheduled to transmit. The frequencies assigned to the receiver nodes s , 1, and 2, are shown within the boxes. The left-most column in the table of Figure 3.2(c) lists the receivers, and the entries in each row list the nodes from which packets are received by their corresponding receivers in each time slot, represented by the columns.

We note that at the end of frame 1, the sink has not yet received packets from nodes 4, 5, and 6; however, as the same schedule is repeated, aggregated packets from nodes 1 and 4, and nodes 2, 5, and 6 reach the sink starting from slot 1 and slot 2, respectively, in frame 2. The entries (1, 4) and (2, 5, 6) represent single packets comprising aggregated data. Therefore, starting from frame 2 the sink continues to receive aggregated data from *all* the nodes in the network at the rate of once in every 6 time slots. This is when the network reaches a steady state and a *pipeline* gets established. We measure the data collection *rate* by the number of time slots required to schedule all the tree edges exactly once per frame, and call it the *schedule length*. Thus, maximizing the aggregated convergecast throughput under this scenario is equivalent to minimizing the schedule length. In Figure 3.2(b), we illustrate the advantage of using multiple frequencies on

the same network. Here, the frequency assigned to node 2 is different from that of node 1 and sink s . As a result, the secondary conflicts are eliminated and more concurrent transmissions are possible, which reduce the schedule length to only 3 time slots. The corresponding table is shown in Figure 3.2(d). We note that multiple frequencies cannot eliminate primary conflicts, which is due the inherent property of the transceivers being half-duplex. We now formally define the multi-channel scheduling problem.

Multi-Channel Scheduling Problem: Given a spanning T on a general graph G , and K orthogonal frequencies, we want to assign a frequency to each of the receivers and a time slot to each of the edges in T , such that the schedule length is minimized.

3.3 Multi-Channel Scheduling on General Graphs

In Figure 3.2(b), we observed that multiple frequencies, when assigned appropriately to the receivers, can help in reducing the schedule length by eliminating the secondary conflicts. In this section, we first prove the hardness results of the Multi-Channel Scheduling Problem and the Frequency Assignment Problem (defined below) on *general graphs*. Then we design a polynomial time algorithm when sufficient number of frequencies is available that can remove all the secondary conflicts.

3.3.1 Complexity of Multi-Channel Scheduling and Frequency Assignment Problems

Multi-Channel Scheduling Problem (*decision version*): Given a routing tree T on a general graph G , and two positive integers p and q , is there an assignment of time slots to the edges of T using at most q frequencies to the receivers, such that the schedule length is no more than p ?

Theorem 1. *The Multi-Channel Scheduling Problem is NP-complete.*

We first define the concept of *distance-2-edge-coloring* on trees, also known as *strong edge coloring*, and state a known result in Lemma 1 before proving Theorem 1.

Definition 2. *Two edges $e, e' \in E$ in a graph $G = (V, E)$ are within distance 2 of each other if either they are adjacent, or both are incident on a common edge.*

A *distance-2-edge-coloring* of G requires that every two edges that are within distance 2 of each other have distinct colors. The minimum number of such colors needed is called the *strong chromatic index*, denoted by $s\chi'(G)$. Although, finding $s\chi'(G)$ for general graphs is known to be NP-hard [48], the following result of Lemma 1 due to is true for trees. We use this in the proof of Theorem 1. It is also easy to see that even when all the receivers in G are assigned the same frequency, the minimum schedule length is no more than the strong chromatic index.

Lemma 1. *The strong chromatic index $s\chi'(T)$ of a tree $T = (V, E_T)$ is given by [38]:*

$$s\chi'(T) = \max_{(u,v) \in E_T} \{deg(u) + deg(v) - 1\}. \quad (3.1)$$

Proof. (of Theorem 1) It is easy to show that the Multi-Channel Scheduling Problem is in NP. Given a particular assignment, one can verify in polynomial time that: (i) at most q frequencies and p time slots are used, (ii) either the receivers of every edge-pair that form secondary conflict are assigned different frequencies, or their edges are on different time slots, and (iii) all adjacent edges are on different time slots.

To show NP-hardness, we reduce an instance $G' = (V', E')$ of the well known *Vertex Color* problem to an instance $G = (V, E)$ of the Multi-Channel Scheduling Problem, as illustrated with an example in Figure 3.3. Our construction is as follows. Let $|V'| = n$. For every vertex $v_i \in V'$, create a set S_i of q pairs of nodes $\{(u_{is}, v_{is}) : s = 1, \dots, q\}$ in G , and join each pair with an edge e_{is} , treating u_{is} as the parent of v_{is} . Then, create $\binom{q}{2} = q(q-1)/2$ interfering links between all such pairs in each S_i as follows. Consider each u_{is} in turn, for $s = 1, \dots, q-1$, and create an interfering link from u_{is} to v_{il} , for all $l > s$. Thus, every two edges in S_i form an interfering edge structure.

Next, for every edge $e_{ij} = (v_i, v_j) \in E'$, create q^2 interfering links (and hence, q^2 interfering edge structures) in G by considering the two sets: $S_i = \{(u_{is}, v_{is}) : s = 1, \dots, q\}$ and $S_j = \{(u_{js}, v_{js}) : s = 1, \dots, q\}$, and creating an interfering link from each

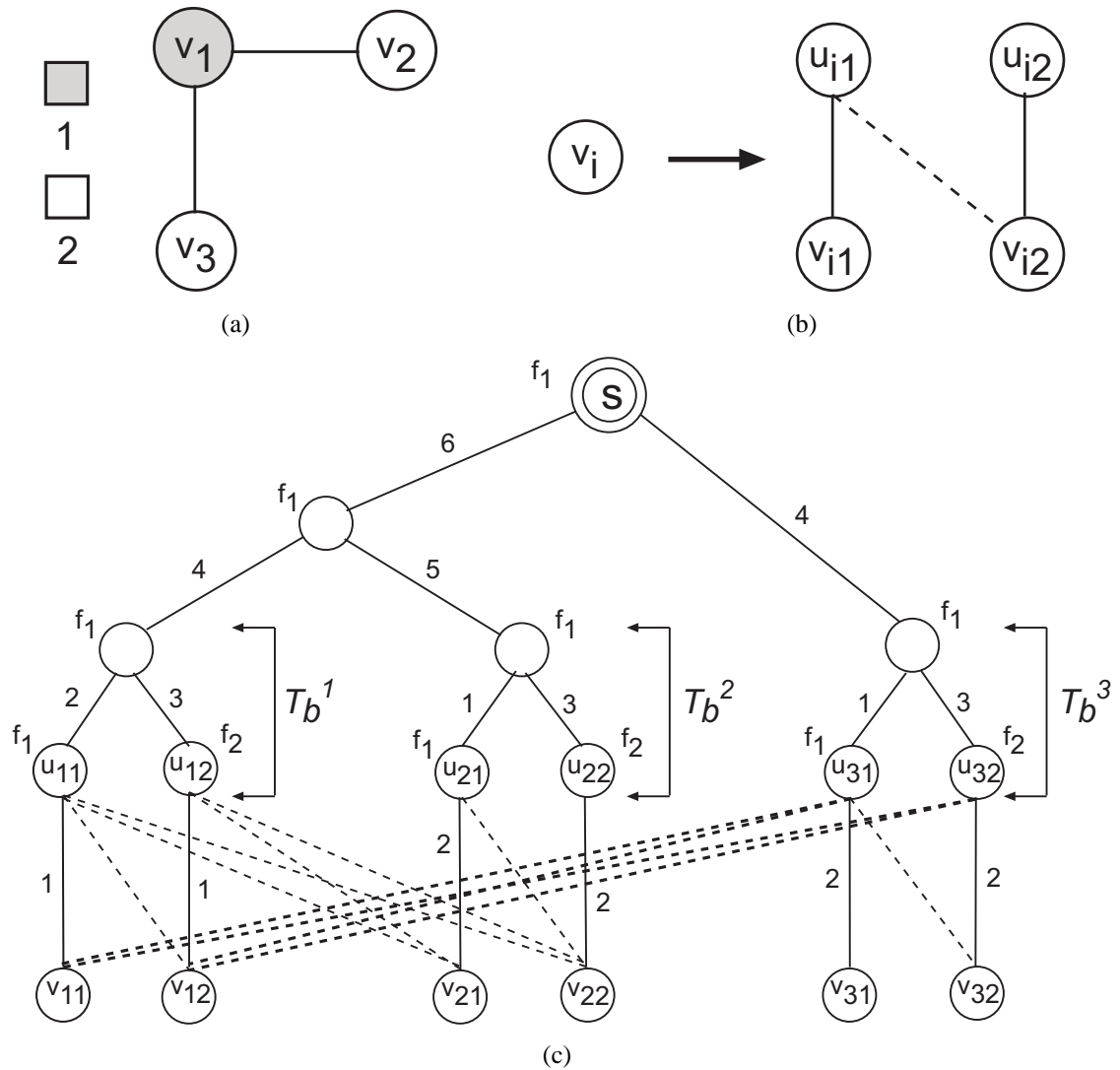


Figure 3.3: Reduction for the Multi-Channel Scheduling Problem: (a) Gadget for each v_i in G' when the number of frequencies q is 2. (b) Instance G' of the vertex color problem. (c) Instance G of the Multi-Channel Scheduling Problem as constructed from G' for $q = 2$.

u_{is} to each v_{js} . Then, for each S_i , construct a binary tree T_b^i creating additional nodes and edges, and treating the $\{u_{is}\}$ nodes as leaves, for $s = 1, \dots, q$.

Finally, treating the roots of T_b^i 's as leaves create a binary tree on top of it, and designate the root of it as the sink s . The reduction clearly runs in polynomial time and creates an instance of the Multi-Frequency Scheduling Problem. Next, we show that there exists a solution to the Vertex Color problem using at most p colors if and only if there exists an assignment in T using at most q frequencies and at most p plus a constant number of time slots.

Suppose G' is vertex colorable using at most p colors, and v_i is assigned color i . First, assign frequency f_s to u_{is} , for $s = 1, \dots, q$, in each S_i , and any one of the q frequencies, say f_1 , to all the parents in the rest of tree. Then, assign time slot i to all the q edges connecting the pairs (u_{is}, v_{is}) , for $s = 1, \dots, q$, in each S_i . Because all the receivers in S_i are assigned different frequencies, assigning the same time slot to all the edges in S_i does not violate the interfering link constraint within each S_i . Also, since only non-adjacent vertices in G' may have the same color, two sets of edges S_i and S_j that are on the same time slot cannot have interfering links between each other, because interfering links exist between S_i and S_j whenever v_i and v_j are adjacent in G' .

Next, the lowest level edges, which connect to the $\{u_{is}\}$ nodes, of all the binary trees T_b^i , $\forall i$, can be scheduled using at most 2 time slots, because all the edges in each S_i are assigned the same slot. Finally, all the remaining edges in the binary tree can be scheduled in polynomial time because a distance-2-edge-coloring on trees can be

computed in polynomial time [129], and within number of time slots no more than its strong chromatic index which, from Lemma 1, equals at most 5.

Conversely, suppose there exists a valid assignment in G that uses at most q frequencies and at most p plus a constant number of time slots. Assign colors to the vertices in G' as follows. For each frequency f_s , consider the set of edges $E_{ts} = \{(u_{ts}, v_{ts})\}$, which are assigned time slot t , for $t = 1, \dots, p$, in order. Since the edges in E_{ts} are on the same time slot and their receivers are on the same frequency, they cannot be part of an interfering edge structure, and so each one of them must lie in a different S_i . Therefore, each edge in E_{ts} has a corresponding vertex in G' , no two of which are adjacent. Select those edges in E_{ts} whose corresponding vertices are unassigned, and assign color t to all of them. Repeat the above assignment for all the frequencies f_s , for $s = 1, \dots, q$. Clearly this uses at most p colors and assigns different colors to adjacent vertices. Also, because we run the above procedure over all frequencies and over all time slots, and select an edge from E_{ts} only when its corresponding vertex is unassigned, exactly one edge gets picked from each S_i . Therefore, every node in G' gets a proper color, and the theorem follows. □

The NP-hardness of the Multi-Channel Scheduling Problem is due to the presence of interfering links that cause secondary conflicts, making scheduling inherently difficult. This is because many subsets of non-conflicting nodes are candidates for transmission in each time slot, and the subset chosen in one time slot affects the number of transmissions in the next time slot. A natural question to ask, therefore, is to find the *minimum* number

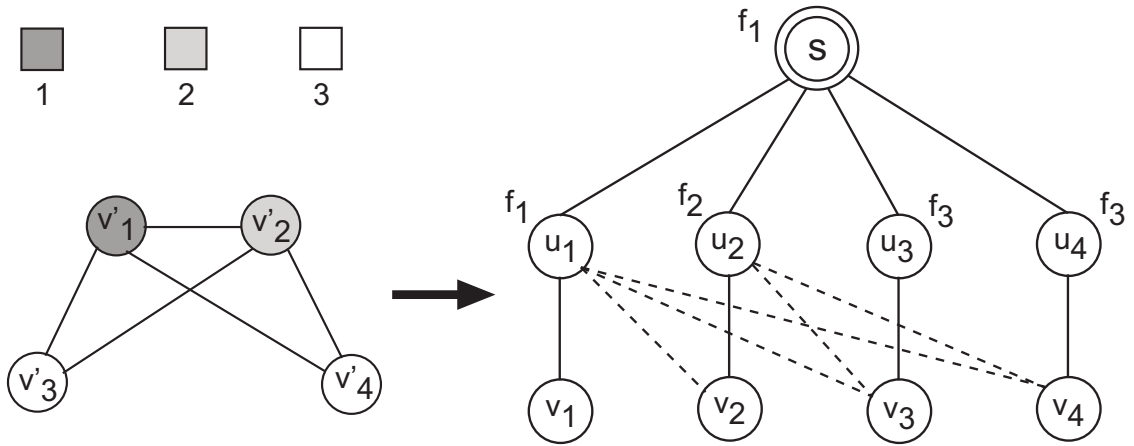


Figure 3.4: Reduction from the Vertex Color problem to the Frequency Assignment Problem. The left part of the figure shows as instance of the Vertex Color problem, which is colored using 3 colors. The right part of the figure shows the corresponding instance of the Frequency Assignment Problem, which also requires 3 frequencies to remove all the secondary conflicts.

of frequencies that is sufficient to eliminate *all* the secondary conflicts, which will then reduce the problem from being on a graph to being on a tree. We say that a secondary conflict is eliminated if the two receivers of an edge-pair are assigned different frequencies. We define this as the *Frequency Assignment Problem*, and prove its hardness result in the following.

Frequency Assignment Problem (decision version): Given a tree T on a general graph G and an integer q , is there a frequency assignment to the receivers in T using at most q frequencies such that all the secondary conflicts are removed?

Theorem 2. *The Frequency Assignment Problem is NP-complete.*

Proof. It is easy to show that the Frequency Assignment Problem is in NP. Given a particular assignment, one can verify in polynomial time (i) if at most q frequencies

are used, and (ii) if the receivers of every edge-pair that cause secondary conflicts are assigned different frequencies.

To show NP-hardness, we reduce an instance $G' = (V', E')$ of the *Vertex Color* problem to an instance $G = (V, E)$ of the Frequency Assignment Problem, as illustrated in Figure 3.4. For every vertex $v'_i \in V'$, create two nodes u_i and v_i in G , and join them with an edge $e_i = (u_i, v_i)$, treating u_i as the parent of v_i . For every edge $e'_{ij} = (v'_i, v'_j) \in E'$, create an interfering link in G between u_i and v_j . Finally, create a root node s , and add an edge $e_{is} = (u_i, s)$ from each u_i to s , treating s as the parent of u_i . This is an instance of the Frequency Assignment Problem, where the tree is given by $T = (V = \{u_i\} \cup \{v_i\} \cup \{s\}, E_T = \{e_i\} \cup \{e_{is}\})$. Clearly, the reduction runs in polynomial time. Next, we show that there exists a solution to the Vertex Color problem using at most q colors if and only if there exists a solution to the Frequency Assignment Problem using at most q frequencies.

Suppose G' is vertex colorable using at most q colors, and suppose v'_i is assigned color j . Assign frequency f_j to u_i in G , and any one of the frequencies, say f_1 , to s . Clearly, this needs at most q frequencies. Since no two adjacent vertices v'_i and v'_j in G' are assigned the same color, no two nodes u_i and u_j in G , which are the receivers of an interfering edge structure, are assigned the same frequency, because by construction a secondary conflict exists between u_i and v_j whenever v'_i and v'_j are adjacent in G' . Therefore, this frequency assignment removes all the secondary conflicts.

Conversely, suppose there exists a solution to the Frequency Assignment Problem using at most q frequencies. If u_i is assigned frequency f_j , assign color j to v'_i in G' . Clearly, this requires at most q colors, because the number of receivers in G is one more than the number of vertices in G' . Since all the interfering link constraints are removed by such a frequency assignment, every two nodes u_i and u_j , which are receivers of an edge-pair that cause secondary conflict, are assigned different frequencies. And since their corresponding vertices v'_i and v'_j are adjacent in G' , they will be assigned different colors, thus yielding a proper coloring of G' . Therefore, the theorem follows. \square

3.3.2 Scheduling Under Sufficient Frequencies

Since finding the minimum number of frequencies required to remove all the secondary conflicts in a general graph is NP-hard, in this section we give an upper bound and show that when *sufficient* number of frequencies is available, the scheduling problem can be solved optimally in polynomial time.

Lemma 2. *Create a constraint graph $G_C = (V_C, E_C)$ from the original graph G as follows (cf. Figure 3.5(a) and 3.5(b) for an illustration). For each receiver (parent) in G , create a node in G_C . Connect any two nodes in G_C if their corresponding receivers in G are incident on two edges that form a secondary conflict. Then, the number K_{max} of frequencies that will be sufficient to remove all the secondary conflicts is bounded by: $K_{max} \leq \Delta(G_C) + 1$, where $\Delta(G_C)$ is the maximum node degree in G_C .*

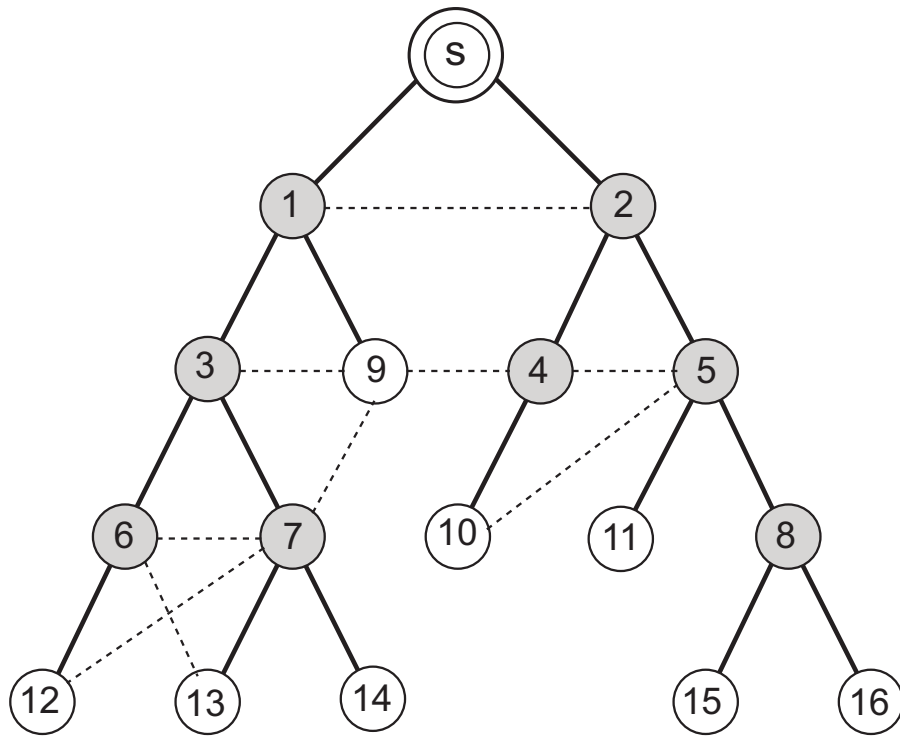
Algorithm 2 BFS-TIMESLOT-ASSIGNMENT

1. **Input:** $T = (V, E_T)$
 2. **while** $E_T \neq \phi$ **do**
 3. $e \leftarrow$ next edge from E_T in BFS order;
 4. Assign minimum time slot to e respecting adjacency constraint;
 5. $E_T \leftarrow E_T \setminus \{e\}$;
 6. **end while**
-

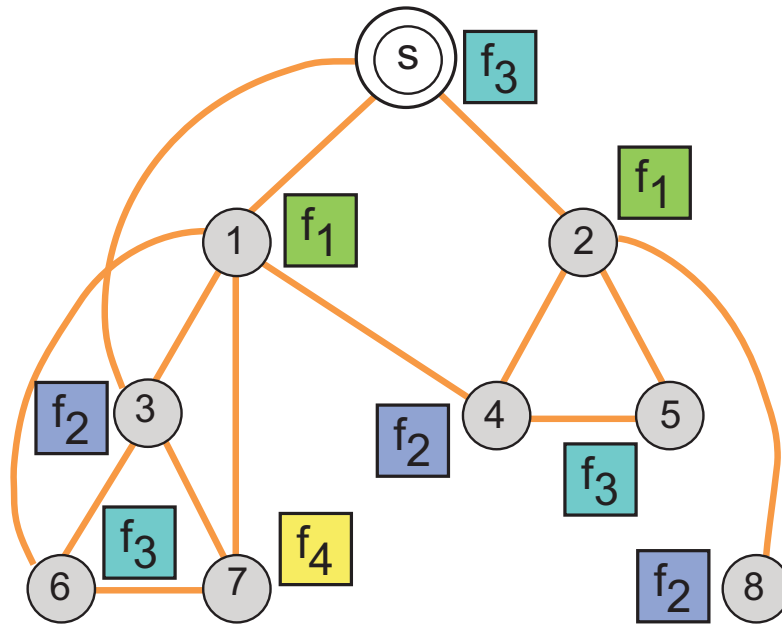
Proof. Since we create an edge between every two nodes in G_C whenever their corresponding receivers in G form a secondary conflict, assigning different frequencies to every such receiver-pair in G is equivalent to assigning different colors to the adjacent nodes in G_C . Thus, K_{max} is equal to the minimum number of colors needed to vertex color G_C , called its *chromatic number*, denoted by $\chi(G_C)$. Since $\chi(G) \leq \Delta(G) + 1$, for any arbitrary graph G , the lemma follows. \square

As illustrated in Figure 3.5(b), the frequencies assigned to the receivers in G_C are as follows: frequency f_1 to nodes 1 and 2; f_2 to nodes 3, 4, and 8; f_3 to nodes 5, 6, and 7; and f_4 to node 8. This particular frequency assignment is according to the heuristic called *Largest Degree First* [91], in which we consider the nodes in G_C in non-increasing order of their degrees and assign the first available frequency such that no two adjacent nodes have the same frequency.

Once all the secondary conflicts are eliminated by an appropriate frequency assignment to the receivers, the following time slot assignment scheme, called BFS-TIMESLOT-ASSIGNMENT (running in $O(|E_T|^2)$ time), presented in Algorithm 2 minimizes the schedule length. In each iteration (lines 2-6) of the algorithm, an edge e is chosen in



(a)



(b)

Figure 3.5: (a) Original graph G ; receiver nodes are shaded. (b) Constraint graph G_C and a frequency assignment to the receivers according to Largest Degree First. Here, 4 frequencies are sufficient to remove all the secondary conflicts, i.e., frequencies on adjacent nodes are different.

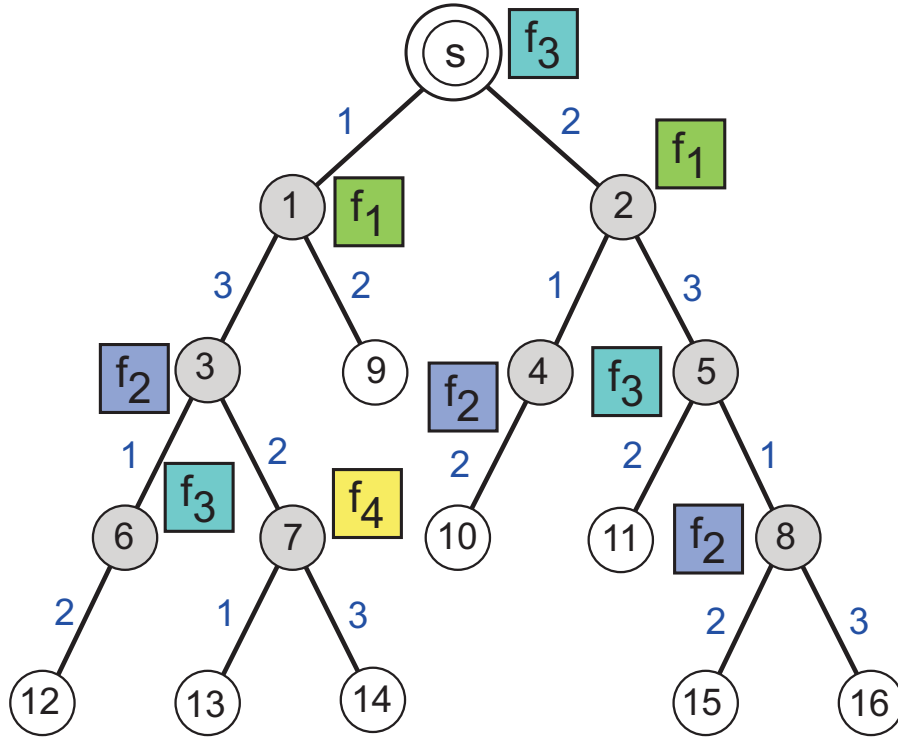


Figure 3.6: An optimal time slot assignment according to Algorithm BFS-TIMESLOT-ASSIGNMENT yielding a schedule length 3 after all the secondary conflicts are removed using 4 frequencies.

the *Breadth-First-Search* (BFS) order, starting from any node, and is assigned the minimum time slot that is different from all its adjacent edges. We prove in Theorem 3 that such an assignment gives a minimum schedule length equal to the maximum degree $\Delta(T)$ of T . An illustration of this time slot assignment scheme for the same network topology as in Figure 3.5(a) is shown in Figure 3.6.

Theorem 3. *After all the secondary conflicts are removed, Algorithm BFS-TIMESLOT-ASSIGNMENT gives a minimum schedule length equal to $\Delta(T)$.*

Proof. The proof is by induction on i . Let $T^i = (V^i, E_T^i)$ denote the subtree of T in the i^{th} iteration constructed in the BFS order, where E_T^i comprises all edges that are already

assigned a slot, and V^i comprises the set of nodes on which the edges in E_T^i are incident. Note that, $|E_T^i| = i$, because at every iteration exactly one edge is assigned a slot. For $i = 1$, clearly the number of slots used is 1, equal to $\Delta(T^1)$.

Suppose the number of time slots $N(i)$ needed to schedule the edges in T^i is $\Delta(T^i)$. In the $(i + 1)^{\text{th}}$ iteration, after assigning a slot to the next edge in BFS order, the number of slots needed in T^{i+1} can either remain the same as before, or increase by one. Thus,

$$N(i + 1) = \max \{N(i), N(i) + 1\}. \quad (3.2)$$

If it remains the same, $N(i + 1)$ is still the maximum degree of T^{i+1} at end of $(i + 1)^{\text{th}}$ iteration. Otherwise, if it increases by one, the new edge must be incident on a node v^* , common to both T^i and T^{i+1} , such that the number of incident edges on v^* that were already assigned a time slot at the end of i^{th} iteration was $\Delta(T^i)$. This is so because in the BFS traversal, *all* the edges incident on a node are assigned a slot first before moving on to the next node, and because the slot assigned to the new edge is the minimum possible that is different from all that already assigned to the edges incident on v^* until the i^{th} iteration. Thus, at the end of $(i + 1)^{\text{th}}$ iteration, the number of slots used $N(i) + 1$ is equal to the number of assigned edges incident on v^* which, in turn, equals $\Delta(T^{i+1})$. This proves the inductive step. Therefore, it holds at every iteration of the algorithm until the end when $i = |V| - 2$, yielding a schedule length equal to the maximum degree $\Delta(T) = \Delta(T^{|V|-1})$. Since assigning different time slots to the adjacent edges in T is

equivalent to edge coloring T , which requires at least $\Delta(T)$ colors, the schedule length is minimum. □

3.4 Multi-Channel Scheduling on Unit Disk Graphs

3.4.1 Known Routing Topologies

In the previous section, we showed that with sufficient number of frequencies available, all the secondary conflicts can be removed and a minimum length schedule can be found in polynomial time. However, typically there is a limitation on the number of frequencies over which a given transceiver can operate, and as shown in Theorem 1, the scheduling problem is NP-hard for a given (constant) number of frequencies. In this section, we take into account the limited number of frequencies available on current WSN hardware, and design an algorithm for the Multi-Channel Scheduling Problem that achieves a constant factor approximation on the optimal schedule length. We assume that the routing topology is known a priori.

We divide the 2-D deployment region into a set of square grid cells $\{c_i\}$, each of side length α . We define two cells to be *adjacent* to each other if they share a common edge or a common grid point. Thus, a cell can have either 3, 5, or 8 adjacent cells depending on whether it is a corner cell, an edge cell, or an interior cell, respectively. In our approach to design an approximation algorithm for minimizing the schedule length, we decouple the frequency assignment and the time slot assignment phases. We first

assign the frequencies to the receivers in T , such that the maximum number of nodes transmitting on the same frequency is minimized. Then, we employ a greedy time slot assignment scheme. We describe the two phases in detail below.

3.4.1.1 Frequency Assignment

Let $\mathcal{R}_i = \{v_1, \dots, v_i\}$ denote the set of receivers on a given routing tree T that lie in cell c_i , and let $m : \mathcal{R}_i \rightarrow \{f_1, \dots, f_K\}$ be a mapping that assigns a frequency to each of these receivers. Note that, $m(v_j) = f_k$ implies that all the children of v_j transmit on frequency f_k due to our receiver-based frequency assignment strategy.

Definition 3. We define a load-balanced frequency assignment in cell c_i as an assignment of the K frequencies to the receivers in \mathcal{R}_i , such that the maximum number of nodes transmitting on the same frequency is minimized.

To express this formally, we define the *load* on frequency f_k in cell c_i under mapping m as the total number of children of all the receivers in \mathcal{R}_i that are assigned frequency f_k , and denote it by $\ell_i^m(f_k)$. We call the number of children of node v_j its *in-degree*, and denote it by $deg^{in}(v_j)$. Thus,

$$\ell_i^m(f_k) = \sum_{v_j \in \mathcal{R}_i: m(v_j)=f_k} deg^{in}(v_j). \quad (3.3)$$

Then, a load-balanced frequency assignment m^* in c_i is defined as:

$$m^* = \arg \min_m \max_k \{\ell_i^m(f_k)\}. \quad (3.4)$$

Algorithm 3 FREQUENCY-GREEDY

1. **for all** non-empty cell c_i **do**
 2. Sort receivers in \mathcal{R}_i in non-increasing order of in-degrees;
 3. Suppose: $\text{deg}^{\text{in}}(v_1) \geq \text{deg}^{\text{in}}(v_2) \geq \dots \geq \text{deg}^{\text{in}}(v_i)$;
 4. **for** $j = 1$ to i **do**
 5. Find frequency f_k that is least loaded (breaking ties arbitrarily);
 6. Assign f_k to v_j ;
 7. **end for**
 8. **end for**
-

We denote the load on the maximally loaded frequency under mapping m^* in cell c_i by $L_i^{m^*}$. We show that finding a load-balanced frequency assignment is equivalent to scheduling jobs on identical machines with the goal to minimize the *makespan* (i.e., the last finishing time of the given jobs), and prove this equivalence in Lemma 3. Since minimizing the makespan is known to be NP-hard [48], so is a load-balanced frequency assignment. Therefore, we resort to designing an approximation algorithm.

In Algorithm 3, we describe a greedy assignment scheme called FREQUENCY-GREEDY that achieves a constant factor approximation on the optimal load. The basic idea of the algorithm is as follows: For each cell c_i , we sort the receivers in \mathcal{R}_i in non-increasing order of their in-degrees; let this order be: v_1, \dots, v_i . Then, starting from v_1 , we assign to each subsequent node v_j a frequency that has the least load on it so far, breaking ties arbitrarily. In Figure 3.7(a), we illustrate this scheme for two frequencies f_1 and f_2 , and three receivers v_1, v_2 , and v_3 , sorted in non-increasing order of their in-degrees. First, node v_1 is assigned frequency f_1 , incurring a load of 5, as it has 5 children. Then node v_2 is assigned frequency f_2 , giving a load of 3. Finally, node v_3 is also assigned frequency

f_2 , because f_2 is the least loaded so far. In this particular case, the assignment achieves an optimal load of 5 on both frequencies. In general, the following approximation holds.

Lemma 3. *Algorithm FREQUENCY-GREEDY in each cell c_i gives a $\left(\frac{4}{3} - \frac{1}{3K}\right)$ approximation on the maximum load $L_i^{m^*}$ achieved by an optimal load-balanced frequency assignment scheme m^* .*

Proof. Consider a job scheduling problem with K identical machines m_1, \dots, m_K , and i jobs $1, \dots, i$. Suppose, executing a job j on any machine takes time $t_j > 0$. Thus, if $\Psi(k)$ denotes the set of jobs assigned to machine m_k , then the total time m_k takes is given by $\sum_{j \in \Psi(k)} t_j$, and the makespan is defined as the maximum of this quantity over all the machines, i.e., $\max_{1 \leq k \leq K} \{\sum_{j \in \Psi(k)} t_j\}$. The objective of the job scheduling problem is to find an assignment of the jobs to the machines such that the makespan is minimized.

In the case of a load-balanced frequency assignment, we map each receiver $v_j \in R_i$ to job j , and its in-degree $deg^{in}(v_j)$ to time t_j . We also map each frequency f_k to machine m_k . The load on frequency f_k is therefore equal to the total time m_k takes. Thus, minimizing the maximum load over all the frequencies is equivalent to minimizing the makespan over all the machines. Under this mapping, Algorithm FREQUENCY-GREEDY is identical to Graham's list scheduling algorithm according to the *Longest Processing Time* (LPT) [57] first, which achieves a $\left(\frac{4}{3} - \frac{1}{3K}\right)$ approximation on the minimum makespan. Therefore, the lemma follows. \square

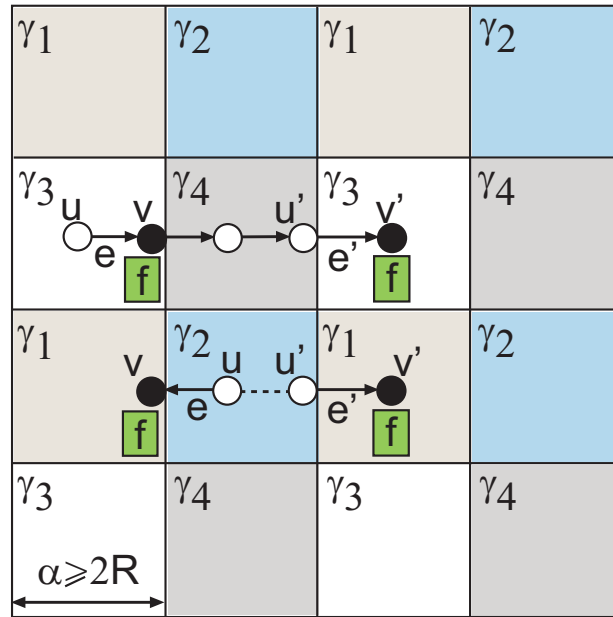
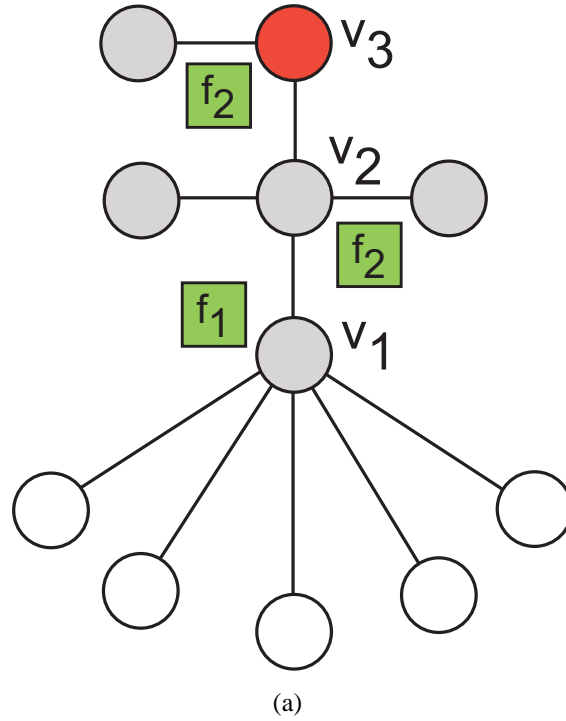


Figure 3.7: (a) Frequency assignment according to Algorithm FREQUENCY-GREEDY. Load on frequencies: $\ell(f_1) = 5$, $\ell(f_2) = 5$. White colored nodes transmit on frequency f_1 ; gray colored nodes transmit on frequency f_2 . (b) Four pair-wise disjoint sets of time slots $\gamma_1, \gamma_2, \gamma_3$, and γ_4 schedule the whole network. Each set γ_i maps to a unique color. Edges whose receivers lie in non-adjacent cells can be scheduled simultaneously, e.g., edges (u, v) and (u', v') .

3.4.1.2 Time Slot Assignment

Once the receivers in each cell c_i are assigned frequencies according to Algorithm FREQUENCY-GREEDY, we employ a greedy time slot assignment scheme for the whole network.

Lemma 4. *Let γ_i denote the set of time slots needed to schedule all the edges in cell c_i . Then, the minimum schedule length Γ for the whole network is bounded by: $\Gamma \leq 4 \cdot \max_i |\gamma_i|$, for all $\alpha \geq 2R$.*

Proof. Consider the grid cells shown in Figure 3.7(b). In our network model, there exists a link between any two nodes if and only if they are at most a distance R from each other. This implies that interference is spatially restricted and time slots can be reused across cells that are spatially well separated. In particular, as illustrated in Figure 3.7(b), for all $\alpha \geq 2R$, two edges $e = (u, v)$ and $e' = (u', v')$, whose respective receivers v and v' lie in non-adjacent cells must have their transmitters u and u' more than distance R away from the other receiver, and so can be scheduled on the same time slot regardless of the frequency assigned to their receivers.

If each set of time slots γ_i represents a unique color, then the whole network can be scheduled using at most four distinct colors such that no two adjacent cells have the same color, i.e., four pair-wise disjoint sets of time slots, as shown in Figure 3.7(b). Therefore, the total number of time slots required is 4 times the maximum number of slots in any set γ_i . □

Lemma 5. *If L_i^ϕ denote the load on the maximally loaded frequency in cell c_i under mapping $\phi : \mathcal{R}_i \rightarrow \{f_1, \dots, f_K\}$ achieved by Algorithm FREQUENCY-GREEDY, then*

any greedy time slot assignment scheme can schedule all the edges in cell c_i within $2 \cdot L_i^\phi$ time slots.

Proof. Consider a multi-graph $H = (\{f_1, \dots, f_K\}, E')$, where for each edge $e = (v_i, v_{i'})$, $v_i, v_{i'} \in \mathcal{R}_i$ with $\phi(v_i) \neq \phi(v_{i'})$, we have an edge $(\phi(v_i), \phi(v_{i'})) \in E'$. Note that these will be multi-edges; let $n(f_k, f_{k'})$ denote the number of edges between f_k and $f_{k'}$ in H . Then, $\deg(f_k) \leq l_i^\phi(f_k)$, where $l_i^\phi(f_k)$ is the load on f_k under ϕ in cell c_i . By Ore's theorem [115], which generalizes Vizing's theorem [107] for edge coloring on multi-graphs, it follows that the edges in H can be colored using $\max_k \{l_i^\phi(f_k)\}$ colors. Therefore, all edges of the form $e = (v_i, v_{i'})$ between two nodes in \mathcal{R}_i with different frequencies can be colored in $\max_k \{l_i^\phi(f_k)\} = L_i^\phi$ colors.

All the remaining edges either have only one end-point in \mathcal{R}_i , or have both end-points in \mathcal{R}_i , with the same frequency on their receivers; let $S(f_k)$ denote the set of such edges with their end-point in \mathcal{R}_i that are assigned frequency f_k . Note that $|S(f_k)| \leq l_i^\phi(f_k)$, and edges $e \in S(f_k), e' \in S(f_{k'})$ can be assigned the same time slot if $f_k \neq f_{k'}$. So all the remaining edges can be scheduled in $\max_k |S(f_k)| \leq \max_k \{l_i^\phi(f_k)\}$ time slots. Therefore, all edges in c_i can be scheduled within $2 \cdot \max_k \{l_i^\phi(f_k)\} = 2 \cdot L_i^\phi$ time slots, and the lemma follows. \square

We now prove the key approximation result on the optimal schedule length.

Theorem 4. *Given a routing tree T on an arbitrarily deployed network in 2-D, and K orthogonal frequencies, there exists a greedy algorithm \mathcal{A} that achieves a constant factor*

$8\mu_\alpha \cdot \left(\frac{4}{3} - \frac{1}{3K}\right)$ approximation on the optimal schedule length, where $\mu_\alpha > 0$ is a constant for any $\alpha \geq 2R$.

Proof. Algorithm \mathcal{A} consists of two phases. In Phase 1, we run algorithm FREQUENCY-GREEDY to assign the K frequencies to the receivers in each cell. In Phase 2, we greedily schedule a *maximal* number of edges in each time slot. Let the schedule length of algorithm \mathcal{A} be $\Gamma_{\mathcal{A}}$, and that of an optimal algorithm be OPT .

Due to the presence of interfering links, there exists a constant $\mu_\alpha > 0$ depending on cell size α and deployment distribution, such that at most μ_α edges in any cell, whose receivers are on the same frequency, can be scheduled in the same time slot by OPT . Now, regardless of the assignment chosen by an optimal strategy, it will take at least $\max_i \{L_i^{m^*} / \mu_\alpha\}$ time slots to schedule all the edges, because $L_i^{m^*}$ is the *minimum* of the *maximum* number of edges that are on the same frequency in cell c_i . Thus,

$$OPT \geq \frac{1}{\mu_\alpha} \cdot \max_i \{L_i^{m^*}\}. \quad (3.5)$$

By running FREQUENCY-GREEDY in cell c_i , Lemma 3 implies

$$L_i^\phi \leq \left(\frac{4}{3} - \frac{1}{3K}\right) \cdot L_i^{m^*}, \quad (3.6)$$

and by scheduling a maximal number of edges in each time slot, Lemma 5 implies $|\gamma_i| \leq 2 \cdot L_i^\phi$. Then, from Lemma 4

$$\begin{aligned}
\Gamma_{\mathcal{A}} &\leq 4 \cdot \max_i \{|\gamma_i|\} \\
&\leq 8 \cdot \max_i \{L_i^\phi\} \\
&\leq 8 \cdot \max_i \left\{ \left(\frac{4}{3} - \frac{1}{3K} \right) \cdot L_i^{m^*} \right\} \\
&\leq 8\mu_\alpha \cdot \left(\frac{4}{3} - \frac{1}{3K} \right) \cdot OPT.
\end{aligned} \tag{3.7}$$

Therefore, the theorem follows. \square

3.4.2 Unknown Routing Topologies

In this subsection, we consider the scenario when the routing topology is not known to the algorithm designer a priori. The significance of this situation is that an optimal algorithm can choose the best routing topology given a deployment of nodes, and as a designer we would want to find properties of a routing topology that could still guarantee a constant factor approximation on the optimal schedule length.

Theorem 5. *Given a network modeled as a UDG and K orthogonal frequencies, there exists an algorithm \mathcal{H} that achieves a constant factor $8\mu_\alpha \cdot \Delta_C$ approximation on the optimal schedule length, so long as the maximum in-degree of any node in the routing tree is bounded by a constant $\Delta_C > 0$, where $\mu_\alpha > 0$ is a constant for a given cell size $\alpha \geq 2R$.*

Proof. Let V_i denote the set of nodes in cell c_i . We note that the set of receivers on the tree depends on the routing topology, but the total number of nodes V_i (in each cell) depends only on the graph. Because an optimal algorithm can concurrently schedule at most a constant number $\mu_\alpha > 0$ of nodes (edges) in any cell c_i whose parents are on the same frequency, the best it could do with K frequencies is distribute the nodes in V_i evenly among all the frequencies, so that $\lceil |V_i|/K \rceil$ is the *minimum* of the *maximum* number of nodes transmitting on the same frequency. Thus,

$$\begin{aligned} OPT &\geq \max_i \frac{1}{\mu_\alpha} \left\{ \left\lceil \frac{|V_i|}{K} \right\rceil \right\} \\ \Rightarrow \max_i \left\{ \left\lceil \frac{|V_i|}{K} \right\rceil \right\} &\leq \mu_\alpha \cdot OPT \end{aligned} \quad (3.8)$$

Suppose $\mathcal{R}_i(T) = \{v_1, \dots, v_n\}$ denote the set of receivers in c_i on an arbitrary routing tree T , and suppose $\Delta^{in}(T)$ be the maximum in-degree of any node in T .

Define a *cyclic frequency assignment* under mapping $\psi : \mathcal{R}_i(T) \rightarrow \{f_1, \dots, f_K\}$ as follows:

$$\psi(v_i) = \begin{cases} i \bmod K, & \text{if } i \neq qK \\ K, & \text{if } i = qK \end{cases} \quad (3.9)$$

where $q \in \mathbb{N}^+$ is a positive integer. It is easy to see that the maximum number of receivers that are on the same frequency is $\lceil |\mathcal{R}_i(T)|/K \rceil$. Therefore, the load L_i^ψ on the maximally loaded frequency in cell c_i is bounded by:

$$\begin{aligned}
L_i^\psi &\leq \left\lceil \frac{|\mathcal{R}_i(T)|}{K} \right\rceil \cdot \max_{v_j \in \mathcal{R}_i(T)} \{deg^{in}(v_j)\} \\
&\leq \left\lceil \frac{|V_i|}{K} \right\rceil \cdot \Delta^{in}(T)
\end{aligned} \tag{3.10}$$

The load L_i^ϕ on the maximally loaded frequency produced by FREQUENCY-GREEDY cannot be more than L_i^ψ ; thus

$$L_i^\phi \leq L_i^\psi \leq \left\lceil \frac{|V_i|}{K} \right\rceil \cdot \Delta^{in}(T) \tag{3.11}$$

Then, scheduling a maximal number of edges in each time slot and using Lemma 4, Lemma 5 as before, and (3.11) it follows that:

$$\Gamma_{\mathcal{H}} \leq 8 \cdot \max_i \left\{ \left\lceil \frac{|V_i|}{K} \right\rceil \cdot \Delta^{in}(T) \right\} \tag{3.12}$$

Since $|V_i|$ and $\Delta^{in}(T)$ are independent of each other, we can take the maximum separately on the two terms as:

$$\begin{aligned}
\Gamma_{\mathcal{H}} &\leq 8 \cdot \max_i \left\{ \left\lceil \frac{|V_i|}{K} \right\rceil \right\} \cdot \max_i \{ \Delta^{in}(T) \} \\
&= 8 \cdot \max_i \left\{ \left\lceil \frac{|V_i|}{K} \right\rceil \right\} \cdot \Delta^{in}(T) \\
&\leq 8\mu_\alpha \cdot \Delta^{in}(T) \cdot OPT.
\end{aligned} \tag{3.13}$$

Thus, (3.13) implies that so long as the maximum in-degree of a node in T is bounded by a constant $\Delta_C > 0$, the theorem holds. Although finding a degree-bounded spanning tree on a general graph is known to be NP-hard [48], for any UDG it is always possible to find a spanning tree of degree at most 5 [153]. Therefore, the theorem follows. \square

3.5 Multi-Channel Scheduling on General Disk Graphs

In this section, we relax our assumption of the nodes having a uniform transmission range, and consider networks where nodes could transmit at different power levels resulting in different transmission ranges. Such networks could be modeled as *general disk graphs*, where a directed edge from node u to node v exists if $d(u, v) \leq R(u)$, where $R(u)$ is the transmission range of u . We consider only those edges that are bidirectional, i.e., both $R(u)$ and $R(v)$ are greater than or equal to their Euclidean distance.

For an edge $e = (u, v)$, we use the convention that u is the transmitter and v is the receiver. We denote the Euclidean length of e by $\ell(e)$. Define $I(e)$ as the set of edges that are either adjacent to e , or form a secondary conflict with e . Also, define $I_{\geq}(e) \subseteq I(e)$ as the subset of edges of $I(e)$ whose end points have *larger* disks than those of e , i.e.,

$$I_{\geq}(e) = \{e' = (u', v') : e' \in I(e) \text{ and } \max\{R(u'), R(v')\} \geq \max\{R(u), R(v)\}\}.$$

As before, we separate the two phases of frequency assignment and time slot assignment. Our goal in the frequency assignment phase is to assign the frequencies in such a

way that minimizes the maximum number of edges that interfere with any given edge. Once this is done, we devise a time slot assignment strategy that optimizes the schedule length.

3.5.1 An Integer Linear Programming Formulation

We formulate the frequency assignment subproblem as a 0-1 Integer Linear Program (ILP). Define indicator variables X_{vk} for edge $e = (u, v)$ as follows:

$$X_{vk} = \begin{cases} 1, & \text{if } v \text{ is assigned frequency } f_k \\ 0, & \text{otherwise} \end{cases} \quad (3.14)$$

A frequency assignment is therefore a 0-1 assignment to the variables X_{vk} , for all $e = (u, v) \in E_T$, and for all f_k . Furthermore, for an edge $e = (u, v)$ on frequency f_k , define Z_{ek} as the total number of edges in $I_{\geq}(e)$ that are also on frequency f_k . Thus,

$$Z_{ek} = \sum_{e'=(u',v') \in I_{\geq}(e)} X_{v'k} = \sum_{v'} n(e, v') X_{v'k}, \quad (3.15)$$

where $n(e, v') = |\{e' = (u', v') \in I_{\geq}(e) : \ell(e') \geq \ell(e)\}|$. If we are given a frequency assignment \vec{X} , the following lemma (based on [88], [61]) shows how the schedule length is related to the Z_{ek} 's.

Lemma 6. ([88], [61]) Let X_{vk} and Z_{ek} be as defined above. Then, $\Omega(\max_{e,k} \{Z_{ek}\})$ is a lower bound on the length of any schedule for the edges. Also, it is possible to schedule all the edges using $\max_{e,k} \{Z_{ek}\}$ time slots.

Proof. (sketch) The lower bound directly follows from [88], [61]. We briefly sketch a greedy scheduling algorithm, which implies the upper bound.

Let $E_T = \{e_1, e_2, \dots, e_{n-1}\}$, with the edges numbered so that $\ell(e_1) \geq \ell(e_2) \dots$. Our scheduling algorithm assigns a time slot $t(e_i)$ for each edge e_i in the following manner: Consider the edges in the order e_1, \dots, e_{n-1} , and for edge e_i , assign the smallest available time slot $t = t(e_i)$ so that that following two conditions satisfy:

- for each edge e_j with $j < i$ having the same receiver as e_i , we have $t(e_i) \neq t(e_j)$,
- for each e_j such that $e_i, e_j \in E_T^k$ (i.e., they are assigned the same frequency), we have $t(e_i) \neq t(e_j)$.

It can now be shown that the number of slots needed is at most $\max_{e,k} \{Z_{ek}\}$. □

The above lemma implies that we should find a frequency assignment that minimizes $\max_{e,k} \{Z_{ek}\}$. We formulate this by the following ILP.

Minimize λ

subject to :

$$\forall e = (u, v), \forall f_k : \sum_{v'} n(e, v') X_{vk} \leq \lambda \quad (3.16)$$

$$\forall e = (u, v) : \sum_k X_{vk} = 1, \quad (3.17)$$

$$\forall e = (u, v), \forall f_k : X_{vk} \in \{0, 1\} \quad (3.18)$$

The second constraint guarantees that each of the receivers is assigned a single frequency. Since solving this ILP is NP-hard, we first find the solution to the linear programming (LP) relaxation of it, which is obtained by modifying the third constraint to include fractional values for the indicator variables as $X_{vk} \in [0, 1]$. Let the optimum (fractional) values for the indicator variables X_{vk} obtained by solving the LP relaxation be X_{vk}^* , and let λ^* be the corresponding objective value. We now construct integral random variables Y_{vk} by rounding the fractional values X_{vk}^* in the following manner: For each v and f_k , we choose $Y_{vk} = 1$ with probability X_{vk}^* . This rounding is done in a mutually exclusive manner, so that for each v , there is exactly one frequency f_k with $Y_{vk} = 1$; this can be done since $\sum_k X_{vk}^* = 1$.

Lemma 7. Let Y_{vk} be the rounded solution, as described above. Then,

$$\max_{e=(u,v),k} \left\{ \sum_{e'=(u',v') \in I_{\geq}(e)} Y_{v'k} \right\} = O(\Delta(T) \log n \cdot \lambda^*), \quad (3.19)$$

with probability at least $1 - \frac{1}{n}$.

Proof. Because of our randomized rounding strategy

$$E[Y_{ik}] = Pr[Y_{ik} = 1] = X_{ik}^* \quad (3.20)$$

Let

$$\widehat{Z}_{ek} = \sum_{e'=(u',v') \in I_{\geq}(e)} Y_{v'k} = \sum_{v'} n(e, v') Y_{v'k}. \quad (3.21)$$

Therefore, by linearity of expectation,

$$\begin{aligned} E[\widehat{Z}_{ek}] &= \sum_{v'} n(e, v') E[Y_{v'k}] \\ &= \sum_{v'} n(e, v') X_{v'k}^* \\ &\leq \lambda^*. \end{aligned} \quad (3.22)$$

Next, note that $n(e, v') \leq \Delta(T)$ for any e, v' . Therefore, by the weighted version of the Chernoff bound [108], it follows that for any edge e and frequency f_k , we have

$$Pr\left[\widehat{Z}_{ek} \geq \Delta(T) \log n \cdot \lambda^*\right] \leq \frac{1}{n^3}. \quad (3.23)$$

Since the number of edges in T and the number of frequencies are both $O(n)$, we have

$$\begin{aligned} Pr \left[\max_{e,k} \left\{ \widehat{Z}_{ek} \right\} \geq \Delta(T) \log n \cdot \lambda^* \right] &\leq \sum_{e,k} Pr \left[\widehat{Z}_{ek} \geq \Delta(T) \log n \cdot \lambda^* \right] \\ &\leq \frac{1}{n}, \end{aligned} \tag{3.24}$$

where the first inequality follows from the union bound. □

The above rounded solution, along with the scheduling algorithm from Lemma 6, leads to the following approximation guarantee.

Theorem 6. *The schedule constructed by Lemma 6, along with the frequency assignment using the above randomized rounding procedure results in a schedule of length $O(\Delta(T) \log n)$ times the optimum.*

3.6 Evaluation

In this section, we evaluate the performance of our scheduling algorithm using MATLAB simulations on networks modeled as *random geometric graphs*. We generate connected networks by uniformly and randomly placing nodes in a square region of maximum size 200×200 , and connecting any two nodes that are at most distance $R = 25$ apart.

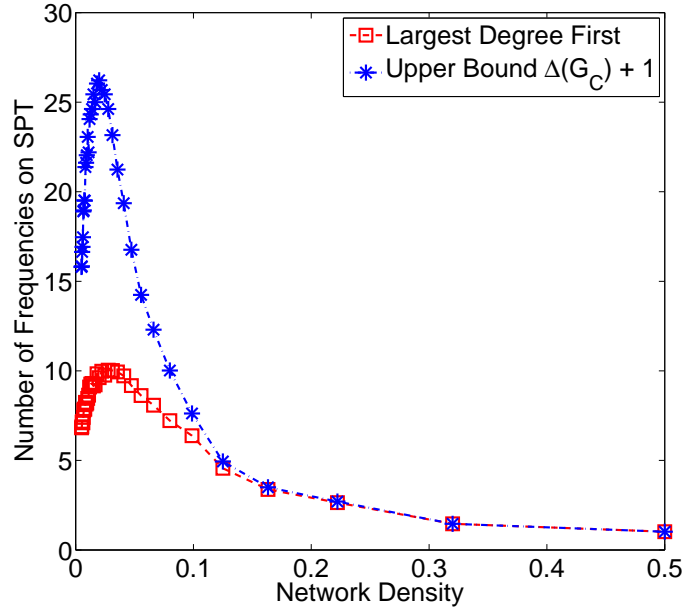


Figure 3.8: Number of frequencies required to remove all the secondary conflicts as a function of network density on shortest path trees.

3.6.1 Frequency Bounds

In Figure 3.8, we compare the number of frequencies needed as a function of network density to remove all the secondary conflicts on shortest path trees, as calculated from the upper bound $\Delta(G_C) + 1$, and that from *Largest Degree First* (LDF) assignment. Here, the number of nodes N is fixed at 200, and the length l of the square region is varied from 200 to 20; so the density, $d = N/l^2$, varies from 0.005 to 0.5.

The plot shows that the number of frequencies initially increases with density, reaching a peak at around 0.025, and then steadily going down to one. This happens because of two opposing factors. As the density goes up, the parents link up with more and more new nodes, increasing the number of secondary conflicts; however, at the same time, the number of parents on the SPT gradually decreases because the deployment region gets

smaller in size. As we keep on increasing the density further, the latter effect starts dominating, and since the number of frequencies required depends on the number of parents in the constraint graph G_C , this number goes down as well, until the network finally turns into a single hop network with the sink as the only parent. We also observe that for sparser networks there is a significant gap between the upper bound and the LDF scheme, as opposed to that in denser networks. This is because in sparser settings there are many parents, resulting in higher a $\Delta(G_C)$ value, and assigning a distinct frequency to the largest degree parent according to the LDF scheme removes more secondary conflicts at every step than it does for denser settings when the parents are fewer and have comparable degrees.

3.6.2 Multiple Frequencies on Schedule Length

Since multiple frequencies can eliminate interfering links and reduce the schedule length, we now evaluate their effects on SPT for our proposed multi-channel scheduling algorithm. Figure 3.9 shows the schedule length with increasing network size for one, three, and five frequencies. We observe that the gains of utilizing multiple frequencies increase as the network gets larger in size. However, the schedule lengths with three and five frequencies are almost the same. This is due to very high node degrees on an SPT resulting in many primary conflicts in the network than secondary conflicts. Recall that primary conflicts are not removable using multiple frequencies.

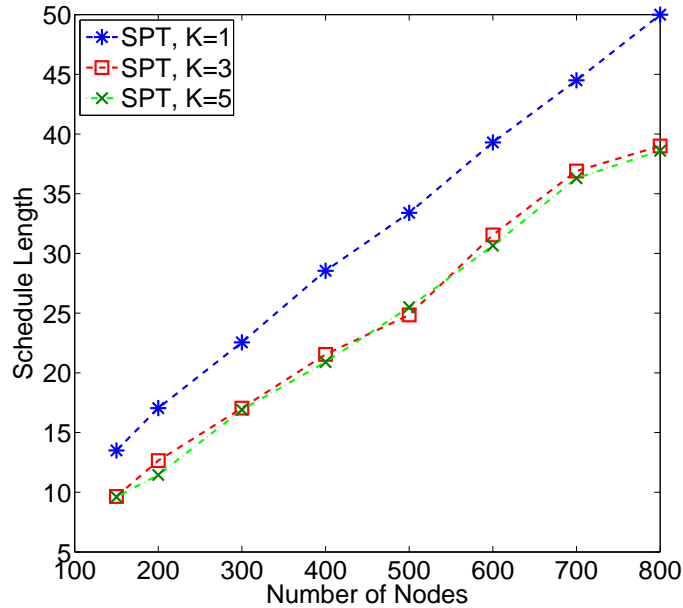


Figure 3.9: Effect of multiple frequencies: Schedule Lengths for SPT with network size for $K = 1, 3,$ and 5 frequencies.

3.6.3 Scheduling Under SINR Model

In our evaluation so far, we have considered the graph-based interference model and evaluated the proposed scheduling algorithm. However, since the protocol model deviates from the more realistic *Signal-to-Interference-plus-Noise-Ratio* (SINR) model in capturing cumulative interference from distant transmitters, thus sometimes under/over estimating interference, the schedules generated from the protocol model might conflict under the SINR model. To measure the amount of conflict, we plot in Figure 3.10 the percentage of nodes on an SPT whose schedules calculated according to the protocol model conflict under the SINR model. The parameters for the SINR model are chosen according to the CC2420 radio parameters with receiver sensitivity -95 dB, path-loss

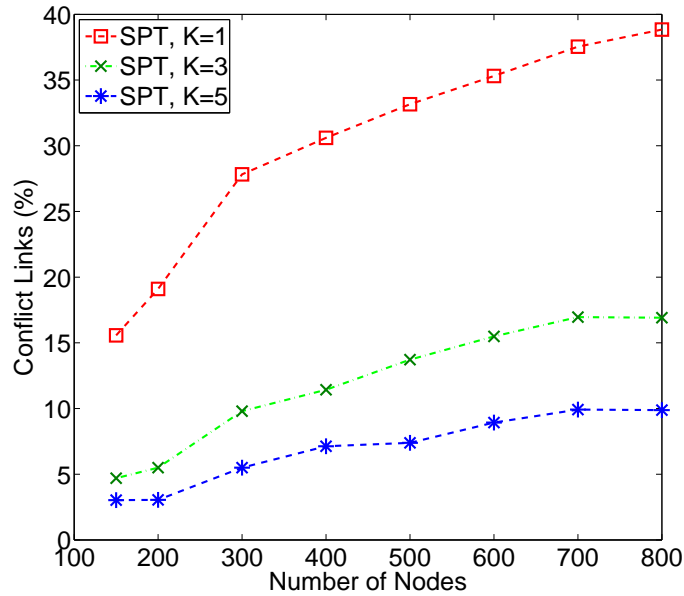


Figure 3.10: Percentage of nodes whose schedules conflict in the SINR model for different network sizes and three different number of frequencies ($K = 1, 3, 5$) on an SPT.

exponent 3.5, and transmit power -6 dB. We note that for a given number of frequencies, as the network gets denser the amount of conflict increases, reaching almost 40% for the densest deployment with one frequency. However, with multiple frequencies, the amount of conflict is much lesser. This indicates that although the protocol model performs reasonably well under multiple frequencies, more sophisticated SINR based scheduling algorithms are needed when there is only one frequency.

3.7 Summary

In this chapter, we considered the aggregated convergecast problem in tree-based sensor networks, and proposed multi-channel scheduling algorithms to maximize the sink throughput. We proved two NP-completeness results related to scheduling for general

graphs – one on finding an optimal time slot assignment that would minimize the schedule length, and the other on finding the minimum number of frequencies that would remove all the secondary conflicts. We showed that once all the secondary conflicts are removed using multiple frequencies, the scheduling problem reduces to one on a tree from being on a graph, and can be solved optimally in polynomial time.

When the number of frequencies is limited, as in most current sensor network hardware, we considered the scheduling problem on random geometric graphs for a given routing tree, and designed approximation algorithms that have provably good, worst-case performance bounds. In particular, for UDG our proposed algorithm gives a constant factor approximation, and for general disks graph it gives an $O(\Delta(T) \log n)$ approximation. When the routing tree is not known a priori, we showed that a constant factor approximation on the optimal schedule length is still achievable so long as the maximum in-degree of any node in the tree is bounded by a constant. We evaluated our algorithms through simulations and showed various trends in performance for different network parameters. We observed that multiple frequencies can reduce the schedule length up to a certain extent, beyond which the structure of the routing tree plays an important role in the scheduling performance. Lastly, evaluating our algorithms on the SINR model revealed some of the weaknesses of the graph-based interference model, and motivated the need for further SINR-based scheduling algorithms.

Chapter 4

Multi-Channel SINR Scheduling for Fast Convergecast

4.1 Motivation

In Chapter 3, we considered the aggregated convergecast problem assuming a simple, graph-based interference model and designed efficient algorithms that have provably good worst-case performance bounds. However, in our evaluation, we observed in Figure 3.10 that the schedule computed by such a graph-based model might be conflicting with respect to a more realistic fading channel model, such as the SINR model, i.e., not all concurrent transmissions might be successful under SINR. Although the percentage of conflicts reduces with multiple frequencies, such a conflicting schedule might need for more retransmissions and undesirable control overhead. In this chapter, we consider the multi-channel scheduling problem under the SINR model and extend our previous algorithms.

The scheduling conflict occurs because of the following reason. In a graph-based model, the network consists of two graphs: a *connectivity graph* and an *interference*

graph, where the vertices represent nodes in both the graphs. An edge between any two nodes exists in the connectivity graph if they are within the transmission range R of each other. Likewise, an edge in the interference graphs exists if two nodes are within a certain distance from each other, for example, within twice the transmission range, as in the *2-hop interference model*. The set of edges in the interference graph is therefore typically a superset of the set of edges in the connectivity graph. Under such a setting, a transmission from a sender s_i to a receiver r_i is always successful so long as there is no concurrent transmission from any other node s_j that has an edge to node r_i in the interference graph, i.e., there is no secondary conflict (assuming primary conflicts have already been taken care of). As a result, solving the scheduling problem often boils down to finding *independent sets* and *coloring* in the graphs.

Although the concept of an *edge* in the interference graph, which is defined based on a fixed distance between the nodes, lends to elegant algorithmic analysis by assuming interference to be a binary and local phenomenon, it is an oversimplification of the physical laws underlying wireless communications. An electromagnetic signal while propagating in the wireless medium does not stop suddenly at a particular distance boundary; it continues to propagate infinitely in space until its intensity gradually fades away with distance and due to the presence of obstacles. Thus, while a single transmission originated outside the interference range of a particular receiver r_i may not disrupt the signal, the effect of many, possibly far-away located, concurrent transmitters might accumulate

and prevent successful reception at node r_i . This illustrates that a set of concurrent transmissions forming an independent set in the interference graph might be conflicting in the underlying SINR model, and might therefore represent an unfeasible schedule.

The rest of the chapter is organized as follows. In Section 4.2, we describe the SINR model and our problem formulation. In Section 4.3, we present an approximation algorithm for multi-channel scheduling under the SINR model...

4.2 Preliminaries

4.2.1 The SINR Model

The SINR model, often known as the *physical interference model*, offers a more realistic representation of wireless communications. A successful transmission in this model accounts for the cumulative interference generated by *all* concurrent transmissions in the whole network, as opposed to the binary and local interference dictated by the presence of interfering edges in a graph-based model. Since the SINR depends on which nodes are being scheduled simultaneously, it is not possible to build an interference graph a priori to finding a scheduling solution.

In the SINR model, the received signal strength decays proportionally to the inverse of the sender-receiver distance raised to the power of the so called *path-loss exponent* α , whose value is a constant and depends on external conditions (such as obstacles, humidity, temperature), as well as on the exact sender-receiver distance. Typically, it is

assumed that α lies between 2 and 4. Thus, if all the nodes transmit at a uniform power P on frequency f , the received signal power $P_{r_i}(s_i)$ from sender s_i to its intended receiver r_i is given by

$$P_{r_i}(s_i) = \frac{P}{d(s_i, r_i)^\alpha}. \quad (4.1)$$

Similarly, the received signal power at r_i from any other node $s_j, j \neq i$ transmitting simultaneously with s_i on the same frequency f , referred to as *interference* and denoted by $I_{r_i}(s_j)$, is given by

$$I_{r_i}(s_j) = \frac{P}{d(s_j, r_i)^\alpha}, \quad j \neq i \quad (4.2)$$

The cumulative interference at r_i is defined as the *sum* of the interferences caused by all the nodes that are transmitting on the same frequency f simultaneously with s_i , and is denoted by $I_{r_i} = \sum_{j \neq i} I_{r_i}(s_j)$. In the SINR model, a transmission from sender s_i to its intended receiver r_i is defined to be successful if and only if the ratio, $SINR_{r_i}$, of the received signal power to the cumulative interference plus an ambient noise power \mathcal{N} at receiver r_i is more than a certain hardware-dependent threshold β , i.e., if and only if

$$\begin{aligned} SINR_{r_i} &= \frac{P_{r_i}(s_i)}{I_{r_i} + \mathcal{N}} \\ &= \frac{P_{r_i}(s_i)}{\sum_{j \neq i} I_{r_i}(s_j) + \mathcal{N}} \\ &= \frac{\frac{P}{d(s_i, r_i)^\alpha}}{\sum_{j \neq i} \frac{P}{d(s_j, r_i)^\alpha} + \mathcal{N}} \geq \beta \end{aligned} \quad (4.3)$$

We refer to the above condition as the *SINR constraint*, and assume that $\beta \geq 1$.

Due to orthogonality, transmissions on different frequencies do not cause interference

4.2.2 Problem Formulation

Our problem formulation is similar to the one for maximizing aggregated convergecast throughput described in Chapter 3. As before, we are given the following: (i) a set V of nodes arbitrarily distributed in the 2-D plane, (ii) a distinguished node $s \in V$ that represents the sink, (iii) a routing tree $T = (V, E_T)$ rooted at the sink, where E_T is the set of edges, which define a parent-child relationship among the nodes in V and need to be scheduled, and (iv) a set of K orthogonal non-interfering frequencies. We define the *length* ℓ_i of an edge $e_i = (s_i, r_i) \in E_T$ between a sender s_i and its intended receiver r_i in the tree as the Euclidean distance between them, i.e., $\ell_i = d(s_i, r_i)$. Note that the parents in the tree (except the sink) are both senders and receivers.

Each node is equipped with a single half-duplex transceiver using which it can either transmit or receive at any given time slot. We assume that every node generates only one packet in the beginning of every TDMA frame, and that it can aggregate packets from its children before transmitting to its parent. Our goal is to find a *minimum length* schedule using multiple frequencies under the *SINR model* so that aggregated packets can reach the sink once per frame after a *pipeline* is established. In other words, we want to assign a frequency to each of the receivers in T following a *receiver-based* frequency assignment strategy, and a time slot to each of the edges in E_T satisfying the SINR constraint of (4.3).

The concept of pipeline is defined in Chapter 3

4.3 Multi-Channel Scheduling Under SINR

4.3.1 Overall Approach

In this section, we extend the multi-channel scheduling algorithm presented in Chapter 3 so that it computes a non-conflicting schedule under the SINR model. As before, we decouple the joint frequency and time slot assignment problem into two separate phases of frequency assignment and time slot assignment, and show that the resulting algorithm still guarantees a provably good worst-case performance bound.

4.3.1.1 Link Diversity

We first classify the edges in E_T based on their lengths, and use the notion of *link diversity* $g(E_T)$, originally proposed in [56], to represent the number of magnitudes of different lengths. Formally,

$$g(E_T) = |\{m \mid \exists e_i, e_j \in E_T : \lfloor \log(\ell_i/\ell_j) \rfloor = m\}| \quad (4.4)$$

In realistic scenarios, $g(E_T)$ is usually a small constant, although in theory it could be as large as the number of nodes. Without loss of generality, we normalize the minimum edge length to one, and let $E = \{E_0, \dots, E_{\log(e_{max})}\}$ denote the set of non-empty length classes, where E_k is the set of edges of lengths lying in the interval $[2^k, 2^{k+1})$, and e_{max} is the longest edge. First, the problem instance is partitioned into disjoint length

classes; then the edges in each class are processed separately for frequency and time slot assignment.

For length class E_k , we divide the 2-D deployment region into a set C_k of square grid cells of side length $\eta_k = \delta \cdot 2^k$ (we will choose δ later). Our greedy frequency assignment strategy is exactly the same as before, i.e., for each cell in C_k , we consider the receivers of those edges in length class E_k that lie within that cell, and assign the K frequencies in a *load-balanced* manner such that the maximum number of nodes transmitting on the same frequency is minimized in that cell. Recall from Lemma 3 in Chapter 3, that such a frequency assignment will give a constant factor approximation on the maximum load on any frequency for each cell in C_k .

4.3.1.2 Reusing Time Slots Under SINR

To assign time slots for each of the edges in length class E_k , we first 4-color the grid cells in C_k such that no two adjacent cells have the same color, as illustrated in Figure 4.1 (cf. Figure 3.7(b)). Note that, unlike in the previous case of a graph-based interference model, now it is no longer clear whether it is possible to reuse time slots in non-adjacent cells in order to schedule edges that are on the *same frequency* under the SINR model. This is because we do not have the concept of an interfering edge anymore whose effect is limited upto only a certain distance, and therefore we need to consider the cumulative interference caused by concurrent transmissions from *all* non-adjacent cells. In the following theorem (similar to Theorem 5.1 in [56]), we prove that such a reuse of time slots

γ_1	γ_2	γ_1	γ_2	γ_1
γ_3	γ_4	γ_3	γ_4	γ_3
γ_1	γ_2	γ_1	γ_2	γ_1
γ_3	γ_4	γ_3	γ_4	γ_3
γ_1	γ_2	γ_1	γ_2	γ_1

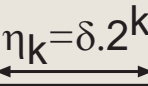
$\eta_k = \delta \cdot 2^k$


Figure 4.1: Coloring the grid cells in C_k corresponding to the length class E_k using four distinct colors $\gamma_1, \gamma_2, \gamma_3,$ and γ_4 . Size of each grid cell is $\eta_k = \delta \cdot 2^k$.

is still possible so long as the size η_k of the grid cells for each length class E_k is chosen appropriately.

Theorem 7. *For cells of the same color in C_k corresponding to a given length class E_k , we can simultaneously schedule at most one edge in E_k from every non-adjacent cell that lies on the same frequency, so long as the cell size η_k satisfies the following:*

$$\eta_k = \delta \cdot 2^k, \quad \text{for } \delta = 4 \left[8\beta \cdot \frac{\alpha - 1}{\alpha - 2} \right]^{\frac{1}{\alpha}}, \quad (4.5)$$

where $\beta \geq 1$ is the SINR threshold, and $\alpha > 2$ is the path-loss exponent.

Proof. For any given frequency, we prove that all the transmissions (at most one per cell of the same color) scheduled on the same time slot satisfy the SINR constraint of (4.3). In particular, without loss of generality, we consider one specific edge $e_i = (s_i, r_i) \in E_k$ in a given cell $c \in C_k$, and prove that the cumulative interference caused by concurrent transmissions from all non-adjacent cells is still within the SINR threshold β .

Since the length of edge e_i satisfies $\ell_i < 2^{k+1}$, the received power at r_i is

$$P_{r_i}(s_i) = \frac{P}{\ell_i^\alpha} \geq \frac{P}{2^{\alpha(k+1)}}. \quad (4.6)$$

The number of non-adjacent cells of the same color that are closest (we call this layer 1) to cell c is 8. For instance, consider the center cell marked as γ_1 and its 8 non-adjacent cells of the same color also marked as γ_1 in Figure. 4.1. Every sender s_j of an edge e_j whose corresponding receiver r_j lies in one of those layer 1 non-adjacent cells must be at least a distance $d(r_i, s_j) \geq \delta \cdot 2^k - 2^{k+1} = 2^k(\delta - 2)$ away from the receiver r_i , as illustrated in Figure. 4.2. Therefore, the interference caused at r_i by such a concurrent sender s_j is

$$I_{r_i}(s_j) \leq \frac{P}{[2^k(\delta - 2)]^\alpha}, \quad (4.7)$$

and consequently, the cumulative interference caused by all the 8 senders from layer 1 non-adjacent cells is

$$\sum_{j=1}^8 I_{r_i}(s_j) \leq \frac{8P}{[2^k(\delta - 2)]^\alpha}. \quad (4.8)$$

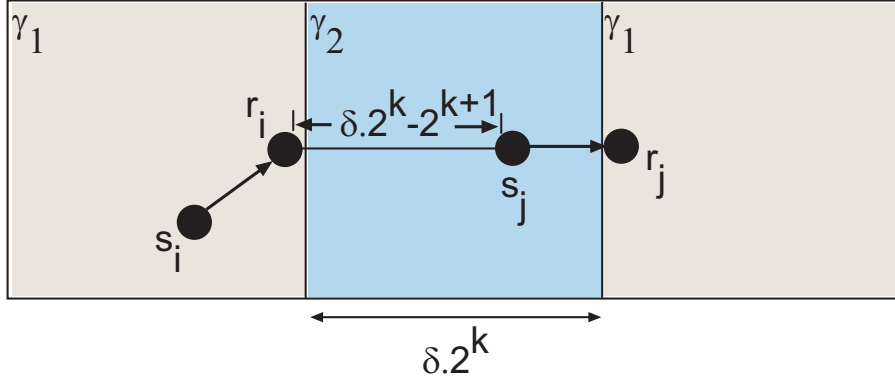


Figure 4.2: For an edge $e_i = (s_i, r_i) \in E_k$, the distance between its receiver r_i and the sender s_j of another edge $e_j = (s_j, r_j) \in E_k$ whose corresponding receiver r_j lies in one of the non-adjacent layer 1 cells of the same color, is at least $\delta \cdot 2^k - 2^{k+1}$.

Next, we consider at most 16 senders whose corresponding receivers lie in layer 2 non-adjacent cells of the same color to c . Each of these senders is at least a distance $3\delta \cdot 2^k - 2^{k+1} = 2^k(3\delta - 2)$ away from the receiver r_i , and thus contributes to a total interference of

$$\sum_{j=9}^{25} I_{r_i}(s_j) \leq \frac{16P}{[2^k(3\delta - 2)]^\alpha}. \quad (4.9)$$

We continue to aggregate interferences from *all* non-adjacent cells of the same color in every layer. In general, the number of such cells in layer q is $8q$, and a sender whose receiver lies in one of those cells is at least a distance $(2q - 1)\delta \cdot 2^k - 2^{k+1} =$

$2^k \{(2q - 1)\delta - 2\}$ away from the receiver r_i . Therefore, the cumulative interference caused by all concurrent senders is bounded by

$$\begin{aligned}
I_{r_i} &\leq \sum_{q=1}^{\infty} \frac{8qP}{[2^k \{(2q - 1)\delta - 2\}]^\alpha} \\
&= \frac{8P}{2^{k\alpha}} \sum_{q=1}^{\infty} \frac{q}{[(2q - 1)\delta - 2]^\alpha} \\
&\leq \frac{8P}{2^{k\alpha}} \sum_{q=1}^{\infty} \frac{q}{[(2q - 1)\delta/2]^\alpha}, \quad \text{for all } \delta \geq 4 \\
&= \frac{8P}{2^{(k-1)\alpha}\delta^\alpha} \sum_{q=1}^{\infty} \frac{q}{(2q - 1)^\alpha}
\end{aligned} \tag{4.10}$$

Substituting $2q - 1 = z$, the infinite sum in (4.10) can be written in the following form:

$$\begin{aligned}
\sum_{q=1}^{\infty} \frac{q}{(2q - 1)^\alpha} &= \sum_{z=1}^{\infty} \frac{z + 1}{2z^\alpha} \\
&\leq \sum_{z=1}^{\infty} \frac{1}{z^{\alpha-1}} \\
&\leq \text{Bound on Zeta function} \\
&\leq \frac{\alpha - 1}{\alpha - 2}.
\end{aligned} \tag{4.11}$$

Thus, using (4.6), (4.11), and (4.10), the SINR at receiver r_i is (ignoring noise \mathcal{N})

$$\begin{aligned}
SINR_{r_i} &= \frac{P_{r_i}(s_i)}{I_{r_i}} \\
&> \frac{\frac{P}{2^{\alpha(k+1)}}}{\frac{8P}{2^{(k-1)\alpha}\delta^\alpha} \cdot \frac{\alpha-1}{\alpha-2}} \\
&= \frac{\delta^\alpha}{4^\alpha \cdot 8 \cdot \frac{\alpha-1}{\alpha-2}} \\
&= \beta,
\end{aligned}$$

substituting $\delta = 4 \left[8\beta \cdot \frac{\alpha-1}{\alpha-2}\right]^{\frac{1}{\alpha}}$. Therefore the theorem follows. \square

4.3.2 $O(g(E_T))$ Approximation Algorithm

The overall process of frequency and time slot assignment is presented in Algorithm 4. We start by forming sets of edges E_k whose lengths lie in $[2^k, 2^{k+1})$, and consider each length class in turn. For a given length class, we partition the 2-D deployment region into a set C_k of square grid cells of lengths $\delta \cdot 2^k$. Then, for each cell in C_k , we assign the K frequencies to the receivers according to algorithm FREQUENCY-GREEDY. In the time slot assignment phase, we first 4-color all the cells in C_k , and consider cells of the same color in turn. For cells of the same color, we simultaneously schedule *at most* one edge in E_k (respecting adjacency constraint) from every non-adjacent cell with receivers on the same frequency. Note that, Lemma 5 from Chapter 3 holds even in the SINR case, because no two edges on the same frequency from any cell is scheduled simultaneously. We next prove the complexity of the algorithm in Theorem 8.

Algorithm 4 APPROXIMATION ALGORITHM UNDER SINR

1. **Input:** Routing tree $T = (V, E_T)$; K orthogonal frequencies
 2. Let $E = \{E_0, \dots, E_{\log(e_{max})}\}$, where E_k is the set of edges with lengths in $[2^k, 2^{k+1})$;
 3. $t = 1$;
 4. **for all** $E_k \neq \phi$ **do**
 5. Partition the 2-D region into a set C_k of square grid cells of length $\eta_k = \delta \cdot 2^k$;
 6. For each cell $c \in C_k$, consider the edges in E_k whose receivers lie in c , and run FREQUENCY-GREEDY;
 7. 4-color the cells in C_k such that no two adjacent cells have the same color;
 8. **for** color $j = 1$ to 4 **do**
 9. **repeat**
 10. **for** frequency $f = 1$ to K **do**
 11. Pick *at most* one edge in E_k (respecting adjacency constraint) from every cell of color j whose receiver is on frequency f ;
 12. Assign time slot t to all those edges;
 13. **end for**
 14. $t = t + 1$;
 15. **until** All edges in E_k whose receivers lie in cells of color j are scheduled.
 16. **end for**
 17. **end for**
-

Theorem 8. *Given a routing tree T on an arbitrarily deployed network in 2-D, and K orthogonal frequencies, Algorithm 4 gives a $O(g(E_T))$ approximation on the optimal schedule length.*

Proof. Let the schedule length of an optimal algorithm be OPT and that produced by Algorithm 4 be Γ . The proof depends on the choice of a critical grid cell \hat{c} for which the number of edges on the same frequency, as assigned by an optimal *load-balanced* frequency assignment strategy m^* , is maximum across all length classes. This corresponds to the maximum load on any frequency over all length classes, i.e.,

$$L_{\hat{c}}^{m^*} = \max_{k,c} \{L_{c \in C_k}^{m^*}\} \quad (4.12)$$

We first prove that an optimal algorithm can schedule at most μ edges simultaneously that are on the same frequency in any cell of any given length class, where μ is given by

$$\mu = \frac{[2(\sqrt{2} \cdot \delta + 1)]^\alpha}{\beta}. \quad (4.13)$$

and δ as defined in (4.5). This will imply OPT will take at least $L_c^{m^*}/\mu$ time slots to schedule the whole network.

We prove this by contradiction. Consider an edge $e_i = (s_i, r_i) \in E_k$ whose receiver lies in cell $c \in C_k$, and suppose μ more edges (numbered $1, 2, \dots, \mu$) of the same length class whose receivers also lie in cell c are scheduled simultaneously; thus, there are $\mu + 1$ edges scheduled simultaneously in c . Ignoring noise, the $SINR$ at receiver r_i is

$$SINR_{r_i} = \frac{P/d(s_i, r_i)^\alpha}{\sum_{j=1}^{\mu} P/d(s_j, r_i)^\alpha} \quad (4.14)$$

The length of each edge in E_k is at least 2^k , i.e., $2^k \leq d(s_i, r_i) < 2^{k+1}$, and the length of each grid cell in C_k is $\delta \cdot 2^k$. Therefore, the maximum distance between two receivers r_i and r_j that lie in cell c is at most $2\sqrt{2} \cdot \delta \cdot 2^k$, and so $d(s_j, r_i) < (2\sqrt{2} \cdot \delta \cdot 2^k + 2^{k+1})$. Substituting these inequalities and μ from (4.13) in (4.14), we get

$$\begin{aligned} SINR_{r_i} &< \frac{P/2^{k\alpha}}{\mu P / [2\sqrt{2} \cdot \delta \cdot 2^k + 2^{k+1}]^\alpha} \\ &= \frac{[2(\sqrt{2} \cdot \delta + 1)]^\alpha}{\mu} \\ &= \beta \end{aligned} \quad (4.15)$$

Thus,

$$OPT \geq L_c^{m^*} / \mu \quad (4.16)$$

Now, since Lemma 3 from Chapter 3 still holds, and each length class is processed separately in Algorithm 4, the total number of time slots required is

$$\begin{aligned} \Gamma &\leq 4 \cdot \max_{k,c} \{|\gamma_{c \in C_k}|\} \cdot g(E_T) \\ &\leq 8 \cdot \max_{k,c} \{L_{c \in C_k}^\phi\} \cdot g(E_T) \\ &\leq 8 \cdot \max_{k,c} \left\{ \left(\frac{4}{3} - \frac{1}{3K} \right) \cdot L_{c \in C_k}^{m^*} \right\} \cdot g(E_T) \\ &\leq 8\mu \cdot \left(\frac{4}{3} - \frac{1}{3K} \right) \cdot g(E_T) \cdot OPT \\ &= O(g(E_T)), \end{aligned} \quad (4.17)$$

where we have used similar notations from Chapter 3, that $\gamma_{c \in C_k}$ denotes the set of time slots required to schedule all the edges from cell c in length class E_k , and ϕ denotes a frequency assignment strategy according to FREQUENCY-GREEDY. Therefore, the theorem follows. \square

4.4 Summary

In this chapter, we have considered a realistic interference model, the so called SINR model, and extended the multi-channel scheduling algorithms presented in Chapter 3. By using the notion of *link diversity*, which measures the number of non-empty link

classes in the network, and by appropriately choosing grid sizes, which is a function of the path-loss exponent α and the hardware-dependent SINR threshold β , we have shown that slight modification to the algorithms presented earlier retains a good performance ratio even under SINR. In particular, the extended algorithms scale in proportional to $O(g(E_T))$, where $g(E_T)$ is defined as the number of non-empty length classes, which is typically a small constant.

Chapter 5

Optimal Spanning Trees for Maximizing Throughput and Minimizing Packet Delays

5.1 Motivation

In the previous chapter, we showed that multiple frequencies can reduce the number of interfering links and enable more concurrent transmissions, thereby enhancing the data collection date for aggregated convergecast in tree-based sensor networks . Evaluation results of our proposed multi-channel scheduling algorithms indicate that increasing the number of frequencies beyond a certain point has diminishing returns. For instance, on shortest path trees (SPT), the schedule lengths remain almost similar when utilizing 3 or 5 frequencies. The reason behind this trend is that, on shortest path trees, nodes choose their parents based on minimum hop counts to the sink, and so in a densely populated network the number of children for each node is enormous, resulting in high

Parts of this chapter are based on [52] and [51].

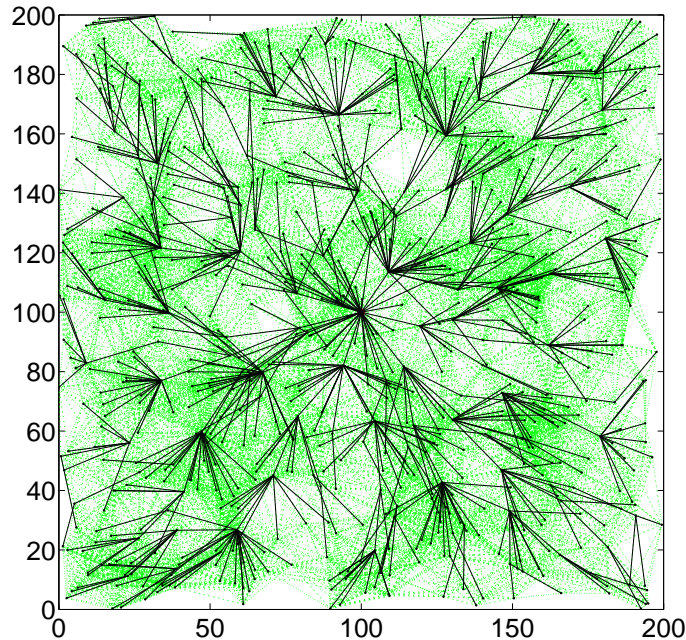


Figure 5.1: Shortest path tree (SPT): High node degrees and minimum hop distances to the sink. Dark lines represent tree edges and dotted lines represent interfering links.

node degrees on the routing topology. This leads to a large number of primary conflicts, because those children need to be scheduled at different time slots due to the single half-duplex transceiver on the nodes. Recall that multiple frequencies cannot eliminate primary conflicts. Figure 5.1 shows a sample routing topology based on shortest path trees on a network of 800 nodes randomly deployed in a region of size 200×200 . The sink is located at the center, and a link between any two nodes exists if they are within distance 25 of each other. We observe that the tree has high node degrees but minimum hop distances to the sink.

In order to further verify that high node degrees indeed create bottlenecks and yield diminishing returns with increasing number of frequencies, and that routing topologies

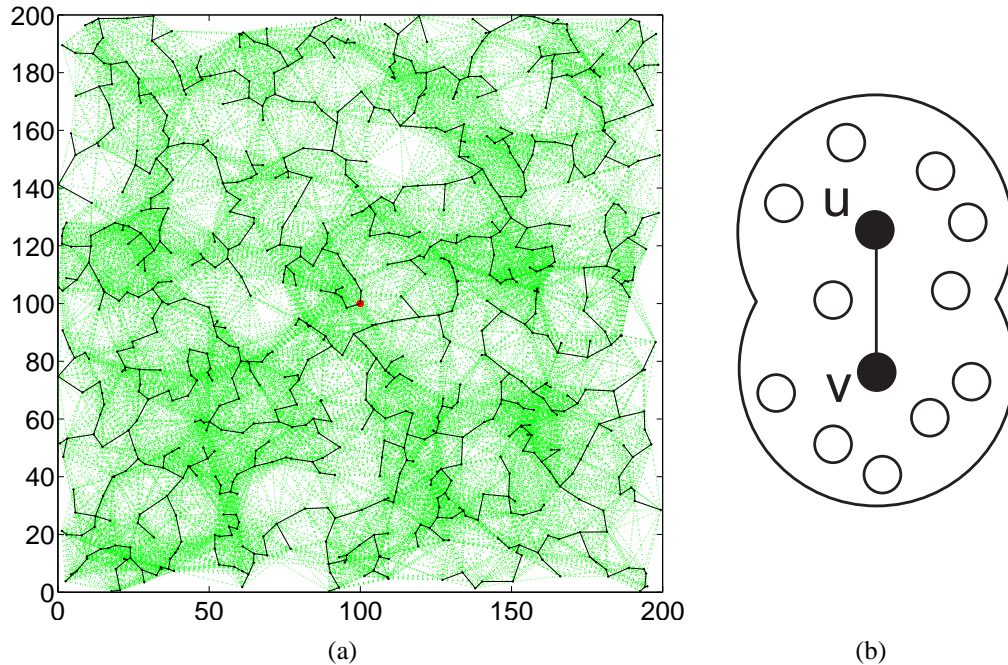


Figure 5.2: (a) Minimum interference tree (MIT): Low node degrees but large number of hops to the sink. Dark lines represent tree edges, dotted lines represent interfering links. (b) Cost of an edge (u, v) is equal to the number of nodes lying within the union of the two disks centered at nodes u and v , each of radius $d(u, v)$; here cost is 11.

with low node degrees might further enhance the data collection rate, we consider a particular type of spanning tree, known as the *minimum interference tree* (MIT) [17], which has very low node degrees and perhaps could achieve a better scheduling performance with multiple frequencies.

An MIT is basically a minimum spanning tree (MST) with a particularly defined edge cost. It is constructed as follows: Given a graph $G = (V, E)$, define the cost of an edge $e = (u, v) \in E$ as the number of nodes covered by the union of the two disks centered at nodes u and v , each of radius equal to their Euclidean distance $d(u, v)$. This particular cost function (cf. Figure 5.2(b)), therefore, gives a measure of interference by counting the number of nodes affected by u and v communicating with just enough transmission

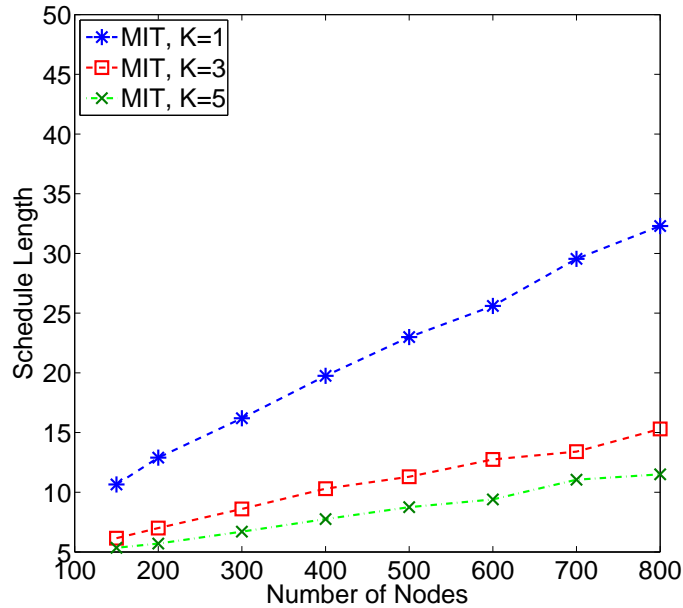


Figure 5.3: Schedule length on minimum interference trees with different network sizes for $K = 1, 3,$ and 5 frequencies.

power to exactly reach each other. Figure 5.2(a) shows an MIT on the same deployment of nodes as in Figure 5.1. Clearly, it has low node degrees but large number of hops to the sink.

In Figure 5.3, we plot the schedule length with increasing network sizes for 1, 3, and 5 frequencies on minimum interference trees by running the same multi-channel scheduling algorithm proposed in the previous chapter. To minimize statistical variation, we construct MITs on the same network topologies that were also used to construct SPTs (cf. Figure 3.9). We observe a significant reduction in the schedule length as compared that on shortest path trees. This is because an MIT has small node degrees and large number of hops to the sink, which gives rise to a lot of secondary conflicts but only a very few primary conflicts. A typical path on an MIT from any node to the sink looks *almost*

like a *linear network*, where every non-adjacent edge can be scheduled simultaneously with only two frequencies, and this is the best one could achieve on linear networks. Note that, with one frequency, only *distance-2 edges*, i.e., edges which are neither adjacent nor whose end points are incident on a common edge, can be scheduled simultaneously.

The discussion so far points out to the fact that routing topologies with low node degrees are perhaps best suitable for maximizing the network throughput in aggregated convergecast. However, many large-scale sensor networks, particularly those intended for mission critical applications, such as disaster early warning systems, or tracking and surveillance systems, also require *timeliness* of data collection in addition to maximizing the throughput. In other words, such applications are not delay-tolerant and need close to real-time data delivery, failure of which might lead to catastrophes.

One of the network properties that contributes to packet delays is the *hop count* on the routing topology from any node to the sink. The further away a nodes is from the sink, the longer it takes for its packets to reach the sink. Thus, a minimum interference tree, which is usually characterized by large number of hops, will incur unacceptable delays for such mission-critical applications. On the other hand, shortest path trees, although have minimum hop counts to the sink, are characterized by high node degrees that tend to lower the data collection rate. Therefore, if shortest path trees are more suitable for minimizing delays, minimum interference trees are better fit for maximizing throughput, but neither of them can yield the best throughput-delay guarantee, for instance, in terms of minimizing the maximum delay while guaranteeing a given lower bound on the

throughput. This calls for the need of constructing special type of routing topologies that have low node degrees as well as small hop counts to the sink.

In addition to minimizing delay, lower hop counts are also useful to achieve better network reliability [84]. For example, consider a network where each edge e fails with probability p_e , and that all failures occur independently. Then the probability that a path $e_1 \rightarrow e_2 \rightarrow \dots \rightarrow e_k$ is operational is $(1 - p_{e_1}) \times (1 - p_{e_2}) \times \dots \times (1 - p_{e_k})$. Thus, given a certain threshold value for the desired reliability, there is a corresponding parameter γ such that the *radius* of the network defined as the maximum of the number of edges on any path to the sink is required to be at most γ . Therefore, reliability constraint is transformed into a radius constraint.

In a similar fashion, besides enhancing convergecast throughput, node degree constraints also appear naturally in many graph-theoretic abstractions of communication network design problems [164]. As an example, consider the IP multicast problem where we would want to disseminate centrally stored information from a server node to a set of client hosts. The standard solution is to construct a tree rooted at the server node and spanning all client nodes. Data packets are then sent from the root along each of its incident edges in the tree. An internal node forwards incoming packets to its children in the tree. The number of children of a node has is therefore proportional to the amount of work done by the node, and hence it is natural to compute spanning trees of low maximum node degree.

In this chapter, we design an algorithm to construct such routing topologies, known as the *Bounded-Degree Minimum-Radius Spanning Trees* (BDMRST) [51], and show that multi-channel scheduling on these trees can help in balancing mutually conflicting performance objectives, in particular, an optimal throughput-delay trade-off.

The rest of the chapter is organized as follows. In Section 5.2, we describe our model and problem formulation. In Section 5.3, we present an approximation algorithm and its analysis to construct a BDMRST. Section 5.4, we evaluate the multi-channel scheduling algorithm on different types of spanning trees.

5.2 Bicriteria Problem Formulation

Given a network modeled as a graph $G = (V, E)$, where V is the set of nodes and E is the set of communication links, we define the *radius* $R(T)$ of a spanning tree T of G rooted at a distinguished vertex $s \in V$ as the maximum number of hops from any node to s . An edge $e = (u, v) \in E$ exists between any two nodes u and v if their Euclidean distance $d(u, v)$ is no more than the transmission range R . We formulate the problem of constructing routing topologies as a *bicriteria optimization problem* [125] where, given an upper bound on the maximum node degree, our goal is to minimize the tree radius. We call such a tree a *Bounded-Degree Minimum-Radius Spanning Tree* (BDMRST), and formally define the routing tree construction problem as follows.

Routing Tree Construction Problem: Given a graph G and a constant parameter $\Delta^* \geq 2$, we want to construct a *Bounded-Degree Minimum-Radius Spanning Tree* T on

G rooted at sink s such that the radius of T is minimized, subject to the constraint that the degree of any node in T is at most Δ^* .

We note that such bicriteria optimization formulations [103] are quite generic and robust, as the quality of approximation is independent of which of the two criteria the budget is imposed on, and it subsumes the case where one wishes to optimize a functional combination of the two objectives, such as, maximizing the sum or product of the maximum node degree and the tree radius. Bicriteria optimization problems on spanning trees are often NP-hard on general graphs, and sometimes even on geometric graphs. Extending the notion of performance guarantee of approximation algorithms for unicriterion optimization problems, here we want to design an (α, β) bicriteria approximation algorithm to the routing tree construction problem, such that the maximum node degree is at most $\Delta^* + \alpha$, and the radius is at most β times the minimum possible radius subject to the degree constraint, where α and β are positive constants.

5.3 Routing Tree Construction

In this section we present an approximation algorithm for constructing a BDMRST, and show that it gives a constant factor approximation on both maximum node degree and tree radius.

5.3.1 An Approximation Algorithm

We tessellate the 2-D deployment region into a set of hexagonal grid cells $\{c_j\}$, each of side length $R/2$, as shown in Figure 5.4. We associate each node to a unique cell whose center is closest to the node, breaking ties arbitrarily. We define a cell to be *non-empty* if it has at least one node, and two cells to be *neighbors* of each other if they share at least one common side. We also define a cell to be *adjacent* to a node u if a circle of radius R centered at u intersects the cell. The basic idea of the spanning tree construction algorithm is as follows.

The algorithm runs in two phases.

- *Phase 1:* In Phase 1, we construct a *backbone tree* $T_B = (V_B \subseteq V, E_B \subseteq E)$ from the original graph G by choosing one *representative node* arbitrarily from each non-empty cell and connecting them in a Breadth-First-Search (BFS) order starting from the sink, while ensuring that the hop distances along T_B is not too long compared to a shortest path tree in G . This backbone tree determines the global structure of our solution. We call the representative nodes the *local roots*.
- *Phase 2:* In Phase 2, we construct a *local spanning tree* of minimum radius within each cell from the remaining nodes in $V \setminus V_B$ lying in that cell while respecting the degree constraint Δ^* . This is always possible because the nodes within each cell form a *complete graph*, as the diameter of the circumcircle for each hexagonal cell is R . Finally, we construct the overall spanning tree T by taking the union of the backbone tree and all the local spanning trees.

During the execution of the algorithm, we *mark* a cell if its local root has been included in T_B ; otherwise the cell is unmarked. We now describe the two phases in detail below.

Phase 1 - Backbone Tree Construction:

- In the beginning, all the cells are unmarked. We initialize the backbone tree with sink s , and mark the cell to which s belongs. Choose one local root arbitrarily from each non-empty cell; let this set of nodes be $\mathcal{R} = \{r_1, \dots, r_n\}$.
- Consider all the unmarked adjacent cells of s in some arbitrary order. In Figure 5.4, these are the shaded cells.
- Let r_j be the local root in the unmarked adjacent cell c_j of s . One of the three possibilities could occur:
 - if r_j lies within the transmission range of s , connect it directly to s and mark c_j . Update the backbone tree by adding r_j to V_B , and the edge (r_j, s) to E_B . Nodes r_1, r_2 , and r_6 in Figure 5.4 are such nodes.
 - if r_j lies outside the transmission range of s , but there is some other node w_k lying within one of the adjacent cells of s , and is within the range of both r_j and s , connect r_j to s via w_k . We call such a node w_k a *helper node*. Mark c_j and update the backbone tree by adding both r_j and w_k to V_B , and the two edges (r_j, w_k) and (w_k, s) to E_B . In Figure 5.4, nodes r_3 and r_5 are connected to s via w_1 , and node r_4 via w_2 .

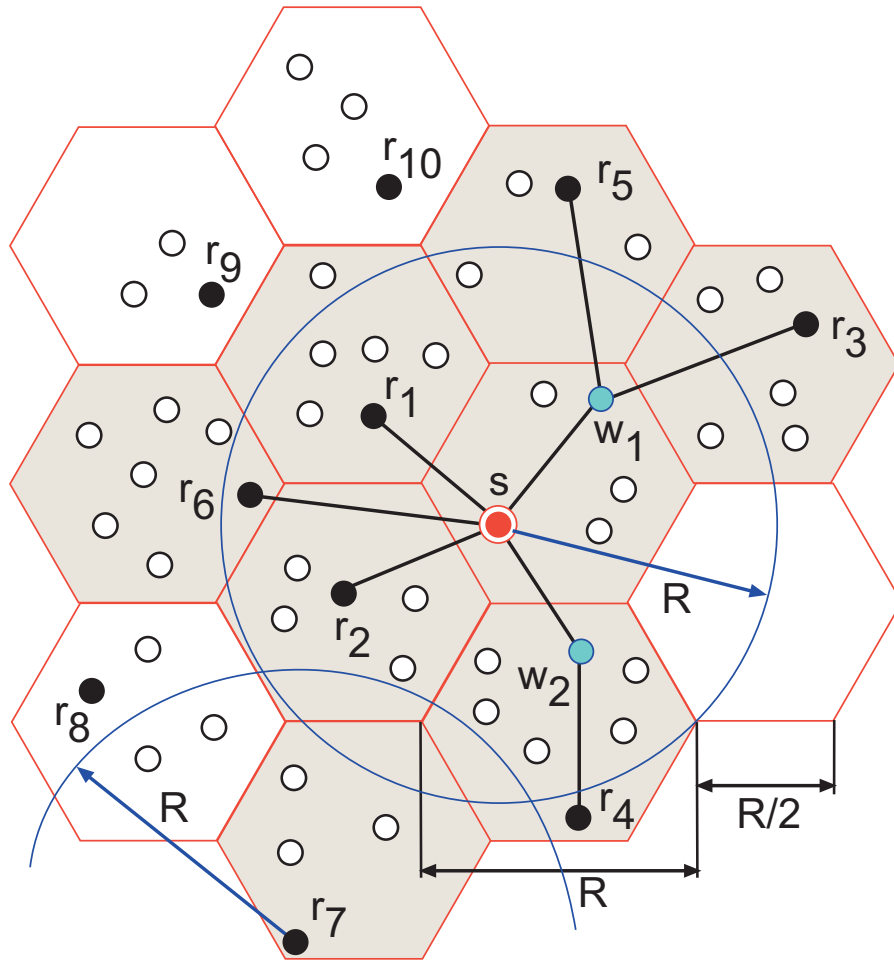


Figure 5.4: Backbone tree construction: Filled black circles represent local roots (chosen arbitrarily from each non-empty cell), and shaded cells are non-empty adjacent cells of s . Iteration 1: Local roots r_1, r_2, r_3, r_4, r_5 , and r_6 from non-empty adjacent cells of s are connected. Nodes r_3 and r_5 are connected to sink s via helper node w_1 , and node r_4 via helper node w_2 . Node r_7 is out of range of any helper node, and is not connected in the first iteration.

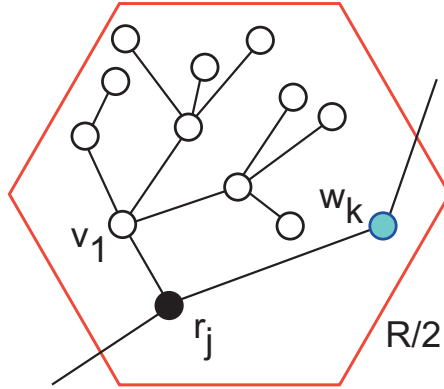


Figure 5.5: Local tree construction on an induced complete graph within each cell for maximum node degree $\Delta^* = 4$; filled black circle represents the local root.

- if r_j lies outside the range of s , and there is no helper node, leave r_j as it is.

Node r_7 in Figure 5.4 is such a node.

- Consider the local roots whose cells have been marked in a BFS order, and repeat the second and third steps. Continue until all the local roots in \mathcal{R} have been connected. We implement the BFS processing of the local roots in a queue data structure.

Phase 2 - Local Spanning Tree Construction:

Since the transmission range of every node is R , nodes lying within each cell induce a *complete graph*, i.e., every node has an edge to every other node. Consider the local root r_j in cell c_j . Let the set of nodes in c_j that are not yet connected to the tree be $V_j = \{v_1, \dots, v_{n_j}\} \subset V \setminus V_B$. Connect v_1 to r_j treating r_j as its parent. Then, connect at most $\Delta^* - 1$ nodes (if those many exist) from V_j to v_1 ; these constitute the direct neighbors of v_1 . Next, treating these direct neighbors as parents, connect at most $\Delta^* - 1$

nodes to each one of them, if those many exist. Figure 5.5 shows an illustration of this phase. Continue this until there is no isolated node left in V_j , and repeat the procedure for each of the non-empty cells. At the end of this phase, each c_j contains a local spanning tree T_j rooted at r_j , with each node (except the leaves and the last parent) having degree Δ^* . The overall spanning tree T is the union of the backbone tree T_B and all the local spanning trees T_j .

A formal description of the algorithm is given in Algorithm 5.

5.3.2 Algorithm Analysis

Theorem 9. *Algorithm 5 gives a constant factor (α, β) bicriteria approximation to the Bounded-Degree-Minimum-Radius Spanning Tree, where $\alpha = 10$ and $\beta = 5$.*

The proof unfolds in the following lemmas.

Lemma 8. *Let r_i and r_j be any two local roots on the backbone tree T_B . Let $P_G(r_i, r_j)$ be the shortest path on the original graph G between r_i and r_j consisting of m hops. Then, the length of the unique simple path $P_{T_B}(r_i, r_j)$ between r_i and r_j on T_B is at most $4m$.*

Proof. Let $P_G(r_i, r_j) = \{r_i = u_0, u_1, \dots, u_{m-1}, u_m = r_j\}$ be the shortest path on graph G , and $P_{T_B}(r_i, r_j) = \{r_i = v_0, v_1, \dots, v_{h-1}, v_h = r_j\}$ be the unique simple path on the backbone tree T_B . Note that each v_k is either a local root or a helper node. We will show that for every edge (u_k, u_{k+1}) in P_G we add at most a constant number of (v_k, v_{k+1}) edges in P_{T_B} .

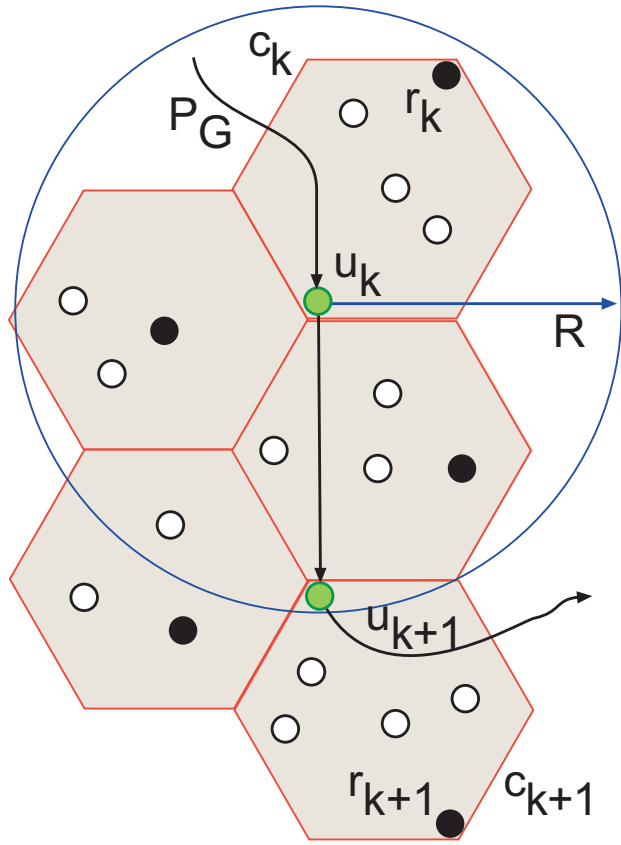


Figure 5.6: Traversing the edge (u_k, u_{k+1}) along the shortest path P_G in graph G . Local roots r_k and r_{k+1} are at most distance $3R$ away from each other.

Consider the nodes u_1, \dots, u_{m-1} , and traverse them in the order as they appear in P_G . At the same time, traverse the path in P_{T_B} by *tracking* the progress in P_G , i.e., for every edge traversed in P_G , we traverse a certain number of edges in P_{T_B} . Both traversals start from the same cell (and the same node r_i). Suppose at any give point, we are about to traverse the edge (u_k, u_{k+1}) . One of the two possibilities could occur: (i) u_k and u_{k+1} lie in the same cell, or (ii) u_k and u_{k+1} lie in different cells, say c_k and c_{k+1} , respectively, as shown in Figure 5.6. In the first case, we do not traverse any edge in P_{T_B} . In the second case, we traverse along the local roots and helper nodes in P_{T_B} , such that the end point of the last edge traversed in P_{T_B} is a local root r_{k+1} that lies in the same cell as u_{k+1} . This is always possible because each non-empty cell contains a local root.

Since the Euclidean length of the edge (u_k, u_{k+1}) is at most R , u_{k+1} must lie in one of the adjacent cells of u_k . Also, since the side length of each cell is $R/2$, the distance between the local roots r_k and r_{k+1} lying in cells c_k and c_{k+1} , respectively, is at most $3R$. Now, during the backbone tree construction phase in Algorithm 5, we first connect the local roots from *all* the non-empty adjacent cells of an already connected local root before exploring other cells. Since connecting a local root to the backbone tree takes at most 2 edges (when helper nodes are needed), the number of edges traversed on P_{T_B} for case (ii) is at most 4. Therefore, the length of the path P_{T_B} is at most 4 times the length of the path P_G . □

Lemma 9. *The degree of any local root or helper node in the backbone tree is at most a constant 12 in the worst case.*

Proof. Recall that during the backbone tree construction, a new local root from each non-empty cell adjacent to an already chosen local root r_j is connected either directly or via a helper node to r_j . This increases the degree of r_j by at most one. Since the side length of each cell is $R/2$, the maximum number of cells that are adjacent to r_j is at most 11; this will happen when r_j lies near one of the corners within its cell c_j . There will be 6 cells that are neighbors to c_j (i.e., share exactly one side with c_j) and 5 other cells that are not neighbors to c_j . Also, recall that during constructing the local spanning trees within each cell, at most one node is connected to the local root of that cell. Therefore, the degree of any local root in the backbone tree is at most 12 in the worst case. A similar argument also holds for the degree of any helper node. \square

Proof. (Theorem 9) (*Bound on the Radius*): Let the radius of an optimal spanning tree whose maximum node degree is Δ^* on graph G be OPT , and let that produced by Algorithm 5 be $R(T)$. Suppose v^* be the node farthest from sink s , and m be the hop distance on a shortest path tree (no degree bound) from s to v^* on graph G . Then, $OPT \geq m$, because a degree constraint can only increase the radius of a tree.

Now the node v^* can either be a local root, or a helper node, or a node in a local spanning tree.

- if v^* is a local root, then by Lemma 8, $R(T) \leq 4m \leq 4 \cdot OPT$.
- if v^* is a helper node, then the shortest path on the tree comprises a path from s to the local root of the cell containing v^* , plus an additional hop from the local root to v^* . Thus, $R(T) \leq 4m + 1 \leq 4 \cdot OPT + 1$.

- if v^* is a node in a local spanning tree T_j , then the shortest path on the tree comprises a path from s to the local root of the cell containing v^* , plus at most $R(T_j)$ hops from the local root to v^* . Thus,

$$\begin{aligned}
R(T) &\leq 4m + R(T_j) \\
&\leq 4 \cdot OPT + OPT \\
&= 5 \cdot OPT
\end{aligned}$$

The second inequality follows because the radius of each local spanning tree T_j constructed on the complete graph within each cell is minimum, respecting degree constraint Δ^* , and so $OPT \geq R(T_j)$.

(*Bound on the Degree*): From Lemma 9, the maximum node degree of any node in the backbone tree is at most 12. Also, the degree of any node in a local spanning tree is at most Δ^* . Thus, the degree of any node in the overall spanning tree is bounded by $\max(\Delta^*, 12) \leq \Delta^* + 10$, for any $\Delta^* \geq 2$.

Therefore, the theorem follows with $\alpha = 10$ and $\beta = 5$. □

5.4 Evaluation

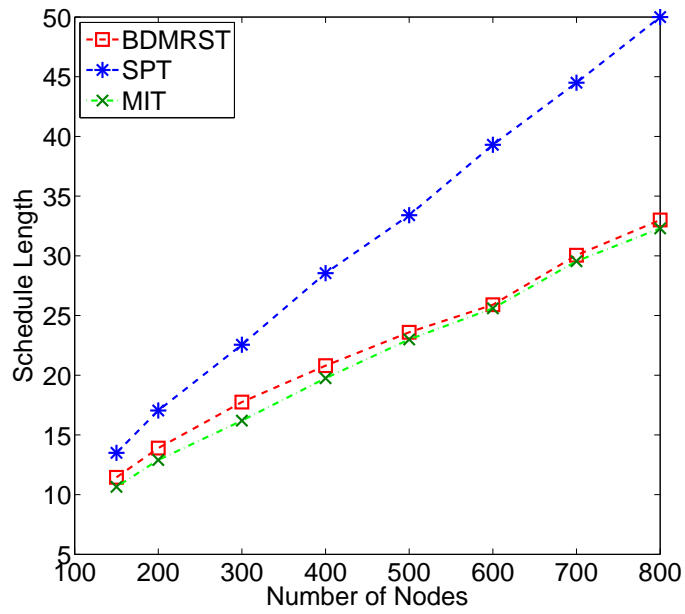
In this section, we evaluate the performance of the multi-channel scheduling algorithm presented in the previous chapter, and the routing tree construction algorithm presented in this chapter using MATLAB simulations on networks modeled as *random geometric*

graphs. We generate connected networks by uniformly and randomly placing nodes in a square region of size 200×200 , and connecting any two nodes that are at most distance $R = 25$ apart.

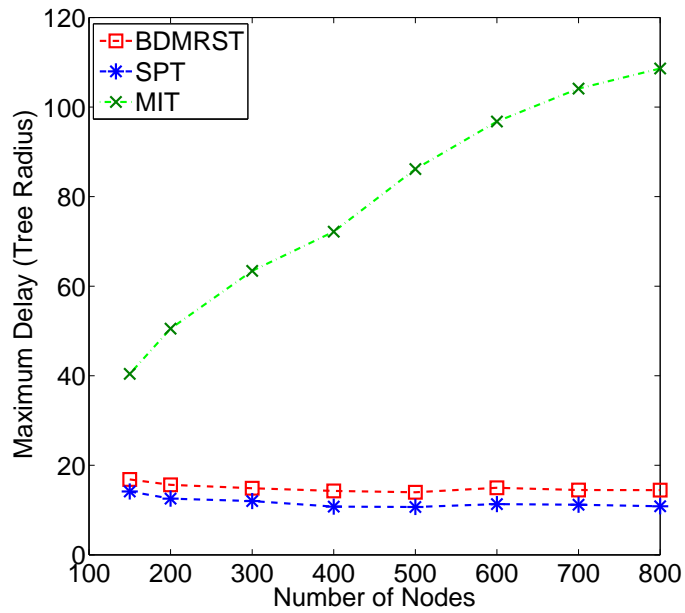
5.4.1 Schedule Length and Maximum Delay

We run the multi-channel scheduling algorithm on three different kinds of spanning trees – BDMRST, SPT, and MIT. We first present the results for a single frequency, and then discuss the case with multiple frequencies.

Figure 5.7(a) and 5.7(b) show the schedule length and the maximum packet delay, respectively, with increasing network size on three different types of trees for single frequency scheduling. Each point in the plot is averaged over 20 iterations. In each iteration, we deploy the nodes uniformly and randomly, construct the spanning trees, and then run the scheduling algorithm. In other words, we keep the node deployment fixed in a given iteration for all the three types of trees in order to preserve the underlying communication graph and minimize any statistical variation. The maximum degree bound on the BDMRST is taken as $\Delta^* = 4$. We observe that the schedule lengths on a BDMRST and MIT are very close to each other, and are much lower than that on an SPT. This difference becomes more predominant with increasing network size, because the maximum node degrees on an SPT go up rapidly, as shown in Figure 5.8. On the other hand, the maximum packet delays on an SPT are minimum, and those on a BDMRST are very close. However, the delays on an MIT are much higher due to its very small and almost



(a)



(b)

Figure 5.7: (a) Schedule Length, and (b) Maximum Delay (tree radius) with increasing network size on three different types of trees (BDMRST, SPT, and MIT) for single frequency scheduling.

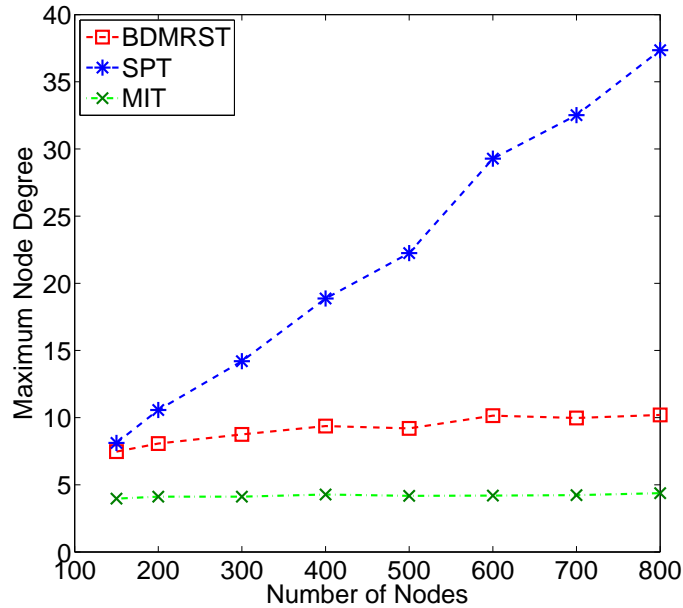


Figure 5.8: Maximum Node Degree with increasing network size on three different types of trees (BDMRST, SPT, and MIT) for single frequency scheduling.

constant node degrees throughout, as shown in Figure 5.8, which give rise to more number of hops. Thus, scheduling on a BDMRST achieves the best of both worlds in terms of having a small schedule length as well as very close to smallest possible maximum delay.

5.4.2 Multiple Frequencies on Schedule Length

Since multiple frequencies can eliminate interfering links and reduce the schedule length, we now evaluate their effects on BDMRST for our proposed multi-channel scheduling algorithm. Figure 5.9 show the schedule lengths with increasing network size for 1, 3, and 5 frequencies on a BDMRST. We observe that the schedule length does not improve with multiple frequencies. This is due to almost constant maximum hop distances to

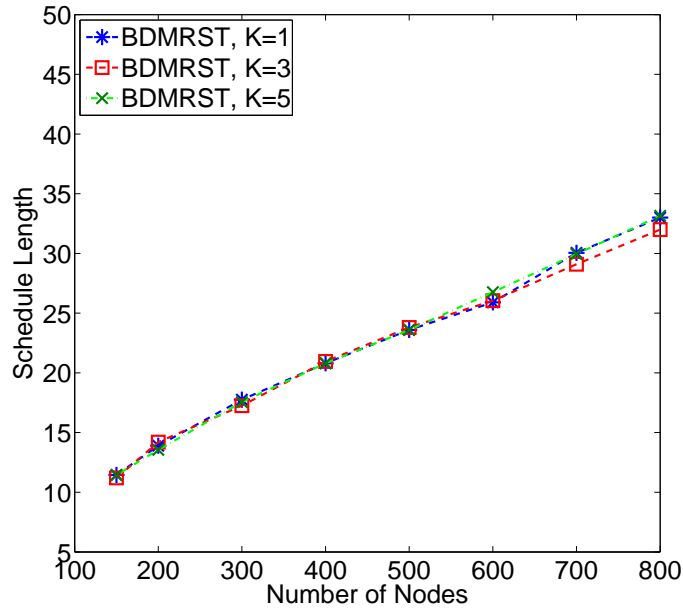
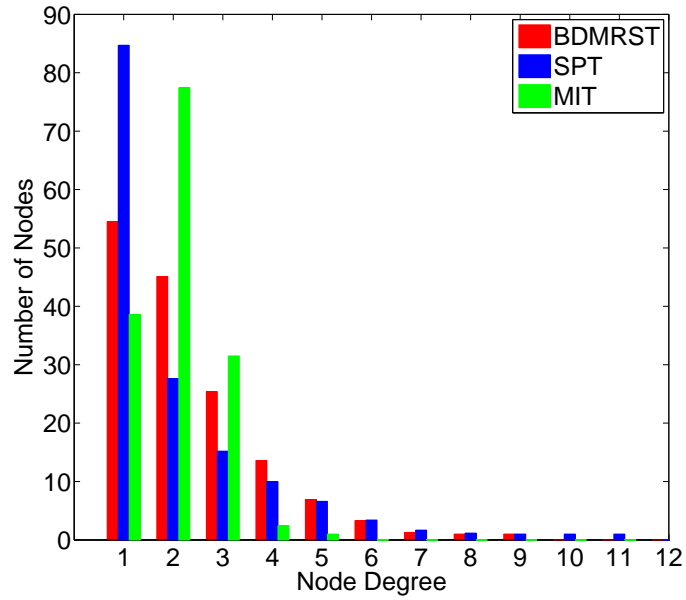


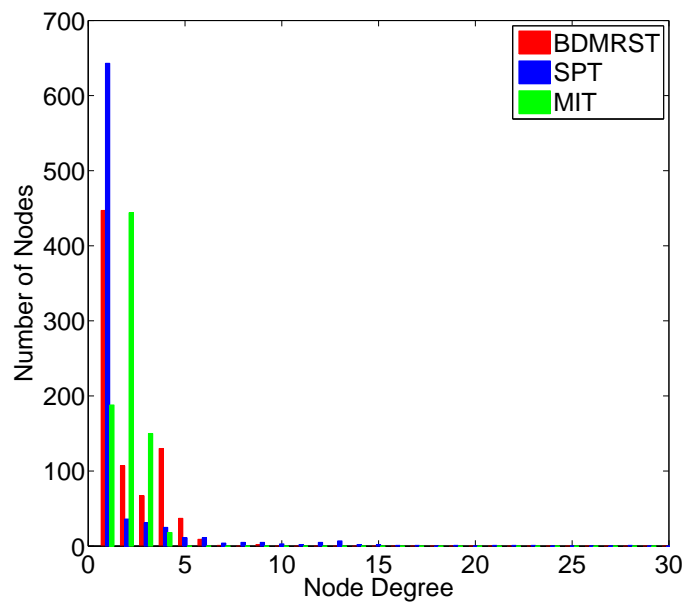
Figure 5.9: Effect of multiple frequencies on schedule lengths for BDMRST with different network sizes for $K = 1, 3,$ and 5 frequencies.

the sink and nearly constant maximum node degrees, as shown in Figure 5.7(b) and 5.8. In order to gain insights on the effects of node degrees on the schedule length, we plot the degree distribution of BDMRST, SPT, and MIT for two different network sizes with $N = 150$ and 800 nodes in Figure 5.10(a) and 5.10(b), respectively. The bar graphs show the number of nodes that have degrees of particular values averaged over 20 iterations for each tree type.

For all network sizes, we observe that most of the nodes on an SPT have degree one, whereas few have degrees very high. This is because large groups of degree one nodes (leaf nodes) are connected to common parents, giving rise to a lot of primary conflicts, and thereby being more resistant to improving the schedule length with multiple frequencies. In MIT, we see that most of the nodes have degree two, and no node has degree



(a)



(b)

Figure 5.10: Node Degree Distribution of BDMRST, SPT, and MIT for two different network sizes with (a) $N = 150$, and (b) $N = 800$ nodes.

more than five. This is because most of the paths from any node to the sink on an MIT

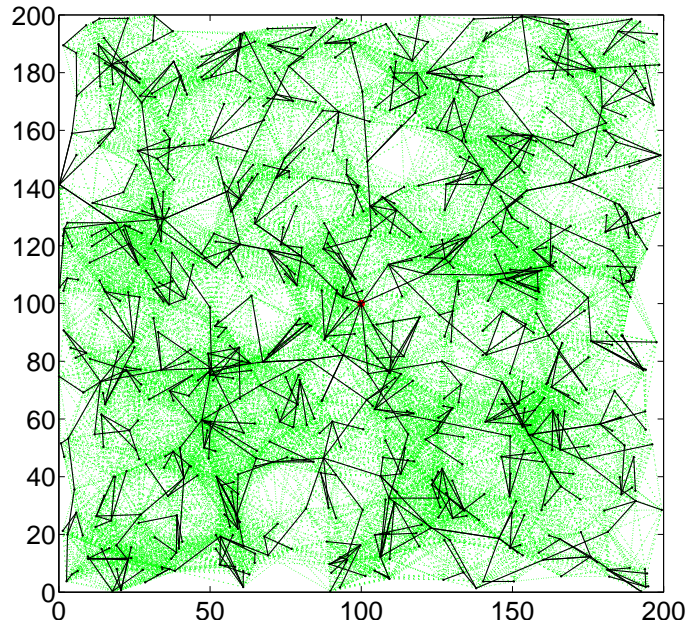


Figure 5.11: A BDMRST constructed on the same deployment of 800 nodes of Figure 5.1 and 5.2(a). The node degrees are more uniform compared to those on an SPT and MIT.

look like a linear topology. We also observe that an MIT has a lot of parent nodes compared to an SPT, thus further explaining the reason for much more improvement in the schedule length with multiple frequencies. Lastly, the degrees on a BDMRST are more evenly distributed for all network sizes, as illustrated in Figure 5.11 by a sample tree constructed on the same deployment of 800 nodes of Figure 5.1.

5.5 Summary

In this chapter, we discussed the trade-off between aggregated convergecast throughput and packet delays for fast and timely data collection in sensor networks. We showed that multi-channel scheduling combined with routing over bounded-degree minimum-radius spanning trees can help in achieving the best of both worlds in terms of maximizing

the throughput and minimizing the maximum packet delay. To this end, we designed a spanning tree construction algorithm that gives a constant factor bicriteria approximation guarantee on minimizing the tree radius under a given maximum node degree constraint.

Algorithm 5 Approximation algorithm for *Bounded-Degree Minimum-Radius Spanning Tree*

1. **Input:** $G = (V, E)$; sink s ; degree bound $\Delta^* \geq 2$
 2. **Output:** BDMRST T of G
 3. Tessellate the 2-D region into hexagonal grid cells, each of side length $R/2$.
 4. Associate each node to a unique cell whose center is closest to the node.
 5. **Phase 1: Backbone Tree**
 6. All cells are unmarked.
 7. Initialize T_B : $V_B \leftarrow \{s\}$, $E_B \leftarrow \phi$, mark cell of s .
 8. Choose one local root arbitrarily from each non-empty cell; let $\mathcal{R} = \{r_1, \dots, r_n\}$ be the set of local roots.
 9. $\mathcal{Q} \leftarrow \phi$;
 10. ENQUEUE(\mathcal{Q}, s);
 11. **while** $\mathcal{Q} \neq \phi$ **do**
 12. $u \leftarrow$ DEQUEUE(\mathcal{Q});
 13. **for all** unmarked cells c_j adjacent to u **do**
 14. $r_j \leftarrow$ local root in c_j ;
 15. **if** $d(u, r_j) \leq R$ **then**
 16. $V_B \leftarrow V_B \cup \{r_j\}$;
 17. $E_B \leftarrow E_B \cup \{(u, r_j)\}$;
 18. Mark c_j ;
 19. ENQUEUE(\mathcal{Q}, r_j);
 20. **else if** $d(u, r_j) > R$ and \exists helper node w_k **then**
 21. $V_B \leftarrow V_B \cup \{r_j, w_k\}$;
 22. $E_B \leftarrow E_B \cup \{(u, w_k), (w_k, r_j)\}$;
 23. Mark c_j ;
 24. ENQUEUE(\mathcal{Q}, r_j);
 25. **end if**
 26. **end for**
 27. **end while**
 28. **Phase 2: Local Spanning Tree**
 29. **for all** non-empty cells c_j **do**
 30. $r_j \leftarrow$ local root in c_j ;
 31. Let $V_j = \{v_1 \dots v_{n_j}\}$ be the set of not yet connected nodes in c_j (V_j induces a complete graph).
 32. Construct local spanning tree T_j of minimum radius with nodes in V_j such that no node exceeds degree Δ^* .
 33. **end for**
 34. **return** $T = T_B \cup \{T_j\}$.
-

Chapter 6

Efficient Distributed Topology Control in 3-Dimensional Wireless Networks

6.1 Motivation

With growing interest in diverse applications of sensor networks, such as structural health monitoring [156], underwater marine life and coral reef monitoring [2], smart homes, and industrial automation, a large number of these networks embedded in the physical world will be three dimensional . Current literature, however, is heavily focused on two-dimensional networks, and there is a tendency to believe that the results from 2-D will directly extend to 3-D. Unfortunately, this is not always true and gives rise to several challenges in terms of computational complexity of the protocols when applied in a 3-D setting [119]. For instance, consider the problem of deploying and configuring a network in order to guarantee complete coverage and connectivity of a region in 3-D space. In

Parts of this chapter are based on [53].

2-D, our assumption about uniform random deployment of nodes with high density is usually well accepted to achieve these goals. However, such high density deployment might not be practical or even possible due to geometric constraints in 3-D.

For n nodes deployed randomly in a unit cube, $[0, 1]^d$, in d dimensions, it is known that the critical transmission radius [86] for connectivity is $O\left(\left(\frac{\log n}{n}\right)^{1/d}\right)$. For $n = 1000$, the critical radius in 2-D is therefore 0.07, while in 3-D it is 0.2, resulting in a critical average node degree of 15 ($\approx \pi \cdot 0.07^2 \cdot 1000$) in 2-D and 34 ($\approx \frac{4\pi}{3} \cdot 0.2^3 \cdot 1000$) in 3-D. This implies that with 1000 nodes, a uniform random deployment in 3-D that is *almost surely connected* will have more than double the number of communication neighbors compared to its 2-D counterpart. Similarly, in terms of coverage, if the sensing region covered by a node is a ball of radius R_s around it, then under uniform random deployment, the ratio of the region covered to the total region is $1 - e^{-\lambda V_s}$, where λ is the density of deployment and V_s is the volume of the ball of radius R_s . For the region to be covered with high probability, say more than 0.99, we must have $1 - e^{-\lambda V_s} \geq 0.99$ which, after simplification, gives $\lambda V_s \geq 4.6$. Since the sensing region of a node will intersect with those of all nodes that are within distance $2R_s$ away, the number of such nodes, on an average, will be 37 ($\approx \lambda \cdot \frac{4\pi}{3} \cdot (2R_s)^3 = \lambda V_s \cdot 2^3 \geq 4.6 \cdot 2^3$) in 3-D, while the corresponding number in 2-D is only 18.

In many distributed protocols, the amount of computation on a node depends on the number of neighbors that are within its communication range and/or sensing range, and both these numbers are very high in 3-D as compared to 2-D. This signifies that simple

extensions of 2-D protocols might not be practical in 3-D due to very high computational requirements, and need to be designed explicitly. In addition, such high node degrees in 3-D could result in excessive interference, thus preventing nodes from transmitting concurrently and reducing network throughput. In such cases, controlling the transmission powers of nodes to form sparser yet connected topologies could substantially alleviate the effects of interference. Moreover, transmitting at lower power levels in a multi-hop network results in more relaying through intermediate nodes, which potentially could save energy consumption. Such power control schemes, which are commonly known as *topology control*, therefore play an important role in optimizing network throughput and prolonging lifetime. In this chapter, we address the problem of efficient distributed topology control in 3-D wireless sensor networks. We note that, this problem falls under the broader class of problems that guarantee a global network property by satisfying certain local constraints. In this case, this global property is network connectivity, and the local constraints, as we will see, are the existence of certain number of neighbors within each node's communication range.

Although the problem of efficient topology control has been well researched in 2-D, its extension to 3-D brings in several challenges that have not been adequately addressed. For instance, there is a natural ordering of nodes in 2-D in terms of angular directions, based on which many topology control algorithms, such as Cone-Based Topology Control (CBTC) [150], [93], are designed. However, no such ordering of angles is possible in 3-D; we only have the notion of solid angles that does not lend itself to any particular

ordering of nodes. There have been extensions of CBTC to 3-D networks, such as the one proposed by Bahramgiri *et al.* [12] using the notion of 3-D cones instead of angles, but their algorithm requires very high computational overhead – $O(d^3 \log d)$ compared to $O(d \log d)$ for 2-D CBTC, where d is the average number of communication neighbors of a node. Given that uniform random deployment of nodes in 3-D requires very high average node degrees for a network to be almost surely connected, an increase by a factor of even d^2 leads to very high computational overhead. This underlines the need for explicitly designing topology control algorithms for 3-D networks.

In this chapter, we focus on one particular aspect of 3-D networks – high node degrees – and employ transmission power control to construct sparse topologies and reduce interference [53]. This brings in additional leverage in optimizing the throughput-delay trade-off. Contrast this with the scenario when there is no flexibility in controlling the transmission power levels, and nodes always transmit at their maximum power. Under this setting, even if we apply our earlier spanning tree construction algorithm that guarantees constant factor approximations on the node degree as well as on the tree radius, the amount of interference caused because of high power levels is far more than in the case with power control. Keeping that mind, our goal here is to design a distributed algorithm, especially catered to 3-D networks with lower computational overhead, in order to minimize the transmission power levels while keeping the network connected, and then

create a spanning tree on the resulting network. In essence, we make use of power control techniques, in addition to routing tree construction and multi-channel scheduling, to optimize the throughput-delay performance in 3-D networks.

Our first approach is based on orthographic projections in 2-D that is simple to implement and runs in $O(d \log d)$ time, where d represents the average node degree. This approach borrows concepts from the 2-D CBTC technique and performs very well in practice, although it does not guarantee a connected network in theory. Our second approach is based on the computational geometry construct, called *Spherical Delaunay Triangulation* [119], [126] which also runs in $O(d \log d)$ time but is always guaranteed to produce a connected network. The rest of the chapter is organized as follows. In Section 6.2, we describe preliminaries and our solution approach. Section 6.3 discusses the multi-dimensional scaling (MDS) technique in 3-D that we use as a primitive. We present our heuristic based on 2-D orthographic projections in Section 6.4, and the more rigorous SDT based algorithm in Section 6.5. Section 6.6 presents detailed simulation results, and we summarize the chapter in Section 6.7.

6.2 Preliminaries and Approach

We first introduce some notations for the ease of exposition. Given a network modeled as a graph $G = (V, E)$, we associate a three tuple $(x_i, y_i, z_i) \in \mathbb{R}^3$ denoting the location coordinates for each node $u_i \in V$. We denote the spherical ball of radius R and center at u_i as $B(u_i, R)$. Given any three non-collinear points p_i, p_j , and p_k on the surface of

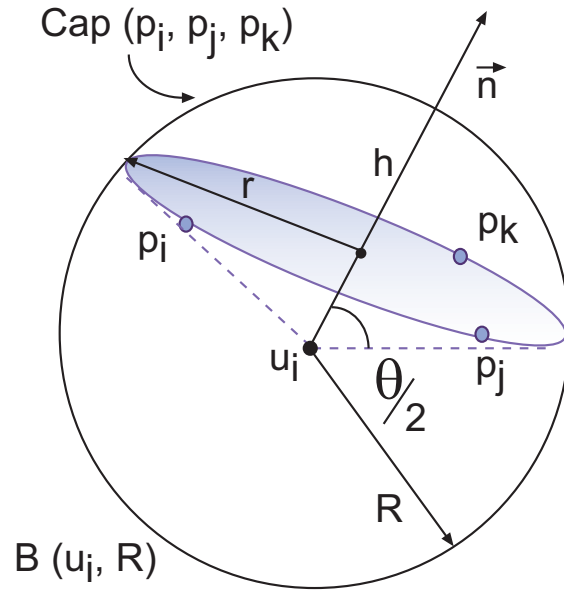


Figure 6.1: Three non collinear points p_i , p_j , and p_k on the surface of a sphere uniquely determine a spherical cap. The radius of the base of the cap is r , and h is its height. \vec{n} denotes normal to the cap.

a sphere, we define a *spherical cap* as the smaller (in terms of volume) portion of the sphere that is cut-off by a plane passing through these three points. Note that, a plane can be uniquely determined by a set of three non-collinear points in 3-D. We denote such a spherical cap by $Cap(p_i, p_j, p_k)$. The height of the cap is denoted by h and the radius of its base is denoted by r , as illustrated in Figure 6.1).

In general, the transmission range of a node is a monotonic function of its power level. We denote by $R(P)$ the transmission range when the power level is P . For a given P , we denote the set of neighbors of node u_i by $N_i(P)$. Since a node can change its neighbor set depending on its transmission power, thus causing changes in the network topology, we assume that the *maximum power graph* formed when all the nodes transmit at their maximum power P^{max} is connected. Our goal is to construct a connected spanning

subgraph of the maximum power graph using only local geometric information, such that it is locally optimal in terms of power levels, i.e., even if a single node transmits at a power level lower than that dictated by the subgraph, then the network will become disconnected.

Our approach in developing a topology control algorithm that works in 3-D with substantially lower computational complexity consists of two phases. In the first phase, we use MDS to find the relative locations of all the neighbors of each node when the transmission power is P^{max} . In the second phase, we propose two different strategies. The first one is heuristic-based, and uses the well known CBTC algorithm. The heuristic performs extremely well in practice, although theoretically it does not guarantee a connected network. The second strategy, which is more rigorous and always guarantees a connected network, uses the properties of spherical Delaunay triangulation. In the following, we describe these two approaches.

6.3 Phase 1: Multi-Dimensional Scaling in 3-D

Multi-dimensional scaling [1] is a statistical method that has been widely used to discover spatial structures and relationships among sets of objects from their observed similarity or dissimilarity data sets. The technique basically transforms a pairwise distance matrix among a set of objects into a set of coordinates, such that the pairwise Euclidean distances derived from these coordinates approximate the original distances as closely as possible. The distance matrix, however, cannot be analyzed directly using eigen-decomposition,

because distance matrices are not positive semi-definite. But if it can be converted into an equivalent cross-product matrix then eigen-decomposition is possible, which gives a principal component analysis (PCA). MDS precisely does that. Each object is represented as a point in a multi-dimensional space, and the points are so arranged that their pairwise distances have the strongest possible relation to the similarities among the pairs of objects. That is, two similar objects are represented by two points that are closer to each other, and two dissimilar objects are represented by two points that are further apart. Finding out the appropriate dimension is also part of the problem in MDS. However, in our case, since we know that the space is 3-D, we can get much better approximations of the relative location maps. The general MDS technique works in any dimensions and even in non-Euclidean space.

MDS has been applied for localization in 2-D [75] to calculate relative sensor locations based on their pairwise Received Signal Strength Intensity (RSSI) values. In our work, based on *link quality indicator* (LQI), we use MDS in 3-D as a primitive for finding relative location maps of all possible neighbors, $N_i(P^{max})$ for each node u_i . Note that, this is slightly different from the earlier approaches as a localization technique. Here, we are interested only in relative locations of each node's neighbors.

6.4 Phase 2: Orthographic Projections

The basic idea here is to simplify the original problem from 3-D by reducing it to multiple similar problems in 2-D using orthographic projections, and then solving the 2-D

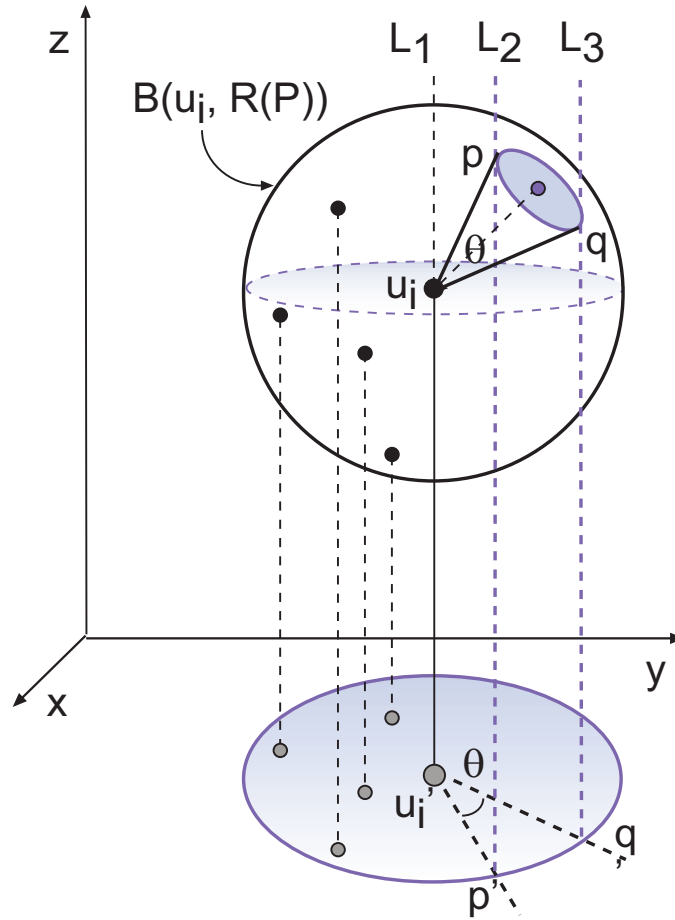


Figure 6.2: An empty sector of angle θ around u_i 's projected location on the xy plane.

problems using CBTC. The 2-D CBTC results says that, if every node adjusts its transmission power level such that there exists at least one neighbor in every sector of angle $\theta = 2\pi/3$ around it, then network connectivity can be guaranteed so long as the MPG is connected. This is called the θ constraint. We begin with the following lemma.

Lemma 10. *Consider the projections of the locations of node u_i and its neighbors for some power level P ($P \leq P^{\max}$) on each of the three orthogonal planes xy , yz , and zx , as illustrated in Figure 6.2. If there is an empty sector of angle θ around u_i 's projection*

on any of the planes, then there exist an infinite number of empty 3-D cones of angle θ around u_i 's location in the 3-D.

Proof. The proof is by construction. In Figure 6.2, there exists an empty sector of angle θ around the projected location u'_i of node u_i on the xy plane for transmission range $R(P)$. Consider the two intersection points p' and q' of the circle and the sector. If we raise the plane of the triangle $\triangle u'_i p' q'$ vertically upwards and parallel to the xy plane, it sweeps a triangular shaped volume bounded by three vertical planes, which are uniquely determined by three pairs of lines L_1, L_2 ; L_1, L_3 ; and L_2, L_3 . By construction, it is clear that all the 3-D cones of apex angle θ contained within the region formed by the intersection of this triangular shaped volume and the spherical ball $B(u_i, R(P))$ are all empty. □

The lemma implies that if there is an empty sector of angle $\theta = 2\pi/3$ on any of the three orthogonal projection planes, then there exist an infinite number of empty 3-D cones with apex angle $2\pi/3$ around node u_i , which in turn implies by the results of [150] that the network will be disconnected if u_i chooses to transmit at the corresponding power level.

We describe the algorithm in Figure 6. Each node u_i starts off by transmitting a “Hello” message at its minimum transmission power level. Neighboring nodes upon hearing the “Hello” message acknowledge back with a “Reply” message. Node u_i then projects the locations of those neighbors from which it heard the “Reply” message on to the xy , yz , and zx planes, as illustrated in Figure 6.3. Then, for each of the three planes,

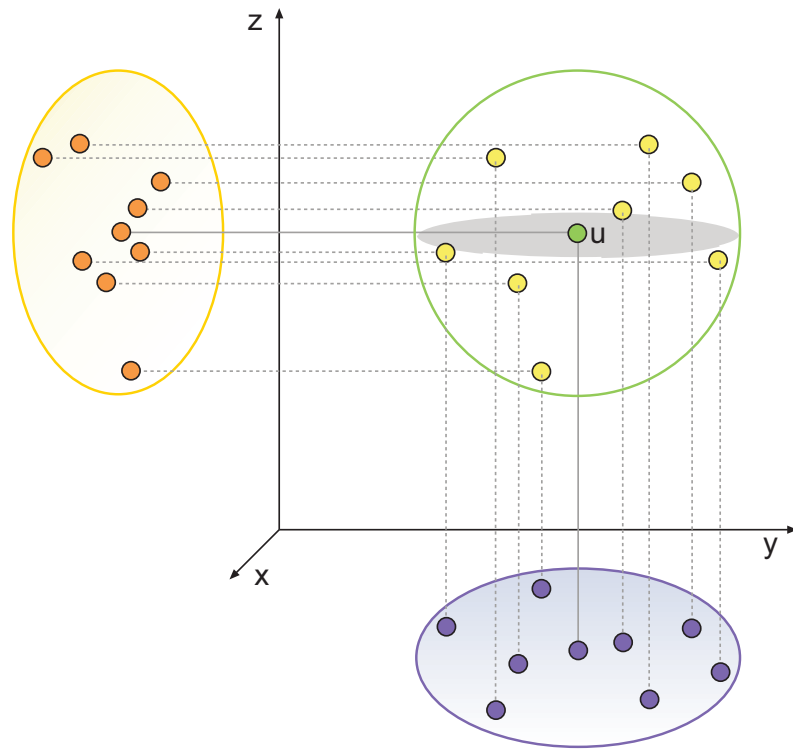


Figure 6.3: Projected locations on xy and zx planes of node u_i and its neighbors $N_i(P^{max})$ when u_i transmits at maximum power.

Algorithm 6 ORTHOGRAPHIC PROJECTION-BASED ALGORITHM

1. $N_i(P) \leftarrow \phi$
 2. $P \leftarrow P^{min}$
 3. $D_i \leftarrow \phi$: directions of projected neighbors
 4. **while** ($P \leq P^{max}$ and $gap_\theta(D_i)$) **do**
 5. Broadcast “Hello” message at power P and gather “Reply” messages from neighbors
 6. $N_i(P) \leftarrow N_i(P) \cup \{v \mid v \text{ replied}\}$
 7. Project locations of u_i and $N_i(P)$ on xy , yz , and zx planes
 8. **for** (each plane) **do**
 9. $D_i \leftarrow D_i \cup \{dir_i(v)\}$
 10. **if** ($gap_\theta(D_i)$) **then**
 11. $P \leftarrow increment(P)$;
 12. **break**
 13. **end if**
 14. **end for**
 15. **end while**
-

node u_i checks whether there is any empty sector of angle $2\pi/3$ around it using the CBTC technique. If *all* the three planes satisfy the θ constraint, it stops and chooses the current power level. Otherwise, it increments its power to the next level, sends another “Hello” message and repeats the above process until there is no empty sector of angle $2\pi/3$ around it on *all* the three projection planes, or until the maximum power level is reached. The minimum power that is required to guarantee the $\theta = 2\pi/3$ constraint on all the three planes is chosen as the transmission power for that node.

As in CBTC, we assume the existence of the following functions: (i) $increment(P)$ that takes the current power level and increases it to the next level, (ii) $dir_i(v)$ that takes the projected locations on a plane of u_i 's neighbors and sorts them with respect to some reference direction, and (iii) $gap_\theta(D_i)$ that takes input as a set of directions and checks if there is an empty sector of angle θ around the projected location of u_i .

The novelty of the heuristic described above is that the algorithm runs in $O(d \log d)$ time and does not assume directional information. However, it should be noted that the network topologies thus generated with transmission power levels as dictated by the algorithm are not always guaranteed to be connected. This can be intuitively seen by the following argument.

Consider a particular node u_i located at the origin and its set of neighbors that lies above the xy plane for a given power level. Project those neighbors on the three orthographic planes. Next, consider a particular 3-D cone of angle $\theta < \pi/2$ around node u_i contained within the first quadrant (i.e., positive x, y, z) and project the cone on the three planes as well. This will form three sectors of angle θ around u_i 's projected locations on the three planes. Now, let there be a particular neighbor that lies just outside and above the surface of the 3-D cone at such a position, which when projected on the three planes, falls within the respective sectors formed by the cones projection on two of the planes (say, xy and xz). Note that, unless a neighbor lies inside the cone, its projection will not fall inside all the three projected sectors. Now with little thought we can convince ourselves that there could be another node(s) that does not lie within the cone but falls within the projected cones sector on the third plane (yz). Therefore, we observe that even though the projected sectors are not empty, that is, they satisfy the θ constraint on all the three planes, the 3-D cone can be empty. This implies that if we base our conclusion of network connectivity by satisfying the θ constraint on the three planes, it might be

incorrect at times for such degenerate cases. Here we restrained $\theta < \pi/2$ to illustrate one particular instance; however, the argument holds true for $\theta = 2\pi/3$ as well.

6.5 Phase 2: Spherical Delaunay Triangulation

The second approach in Phase 2 of our algorithm is based on the properties of *Spherical Delaunay Triangulation*. In computational geometry, Delaunay triangulation is the dual of Voronoi diagrams [8] which, for a set of points, tessellate the 2-D (3-D) region into a set of convex polygons (polyhedra), such that any point lying within a polygon (polyhedron) is closest to the point that is inside the polygon (polyhedron). This is known as the *nearest neighborhood* property of Voronoi diagrams. Likewise, Delaunay triangulation follows the dual of the nearest neighborhood property, called the *empty circle* property, as defined below.

Definition 4. DELAUNAY EMPTY CIRCLE PROPERTY: *Given a set of points lying on a plane such that no four points are co-circular (i.e., affinely independent), the circumcircle around each of the Delaunay triangles is empty, i.e., it does not contain any of the points in its interior. The empty circle property generalizes in 3-D in the form of empty spheres for Delaunay tetrahedrization.*

When Delaunay triangulation is carried out on points that lie on the surface of a sphere, it produces spherical triangles, and the empty circle property still holds. That is, for any three points a , b , and c that form the vertices of a spherical triangle, the spherical

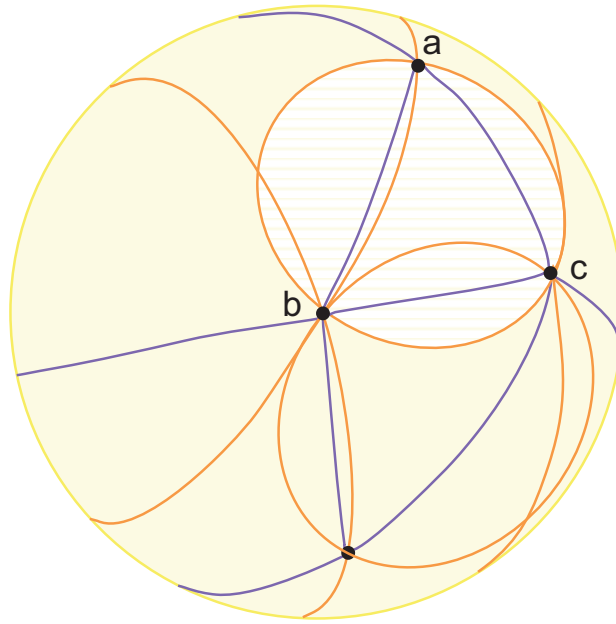


Figure 6.4: Spherical Delaunay Triangulation illustrating empty circle property: Spherical cap $Cap(a, b, c)$ is empty in its interior.

cap $Cap(a, b, c)$ is empty. This is illustrated in Figure 6.4. We use this *empty spherical cap property* as a primitive in our algorithm.

Consider node u_i and its set of neighbors $N_i(P)$ for a given power level P . We project the locations of these neighbors on the surface of a spherical ball centered at u_i and of radius $R(P)$. This construction basically means drawing radial lines connecting u_i and each of the neighbors until they intersect with the spherical surface. Then, we construct a spherical Delaunay triangulation with the projected points. This construction leads to the following lemma.

Lemma 11. *The 3-D cones that are formed with u_i as the apex, and the spherical caps generated from Delaunay triangulation as their bases are empty.*

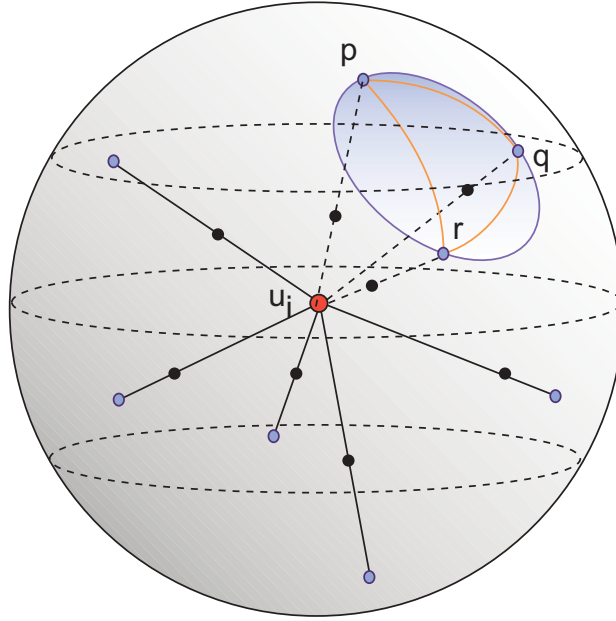


Figure 6.5: The 3-D cone with $Cap(p, q, r)$ as its base and u_i as its apex is empty. Black dots show the actual locations of the neighbors, blue dots show their projected locations on the surface of the spherical ball. Δpqr is a spherical Delaunay triangle.

Proof. The proof is by construction, as illustrated in Figure 6.5. Consider a particular 3D cone that has its base as the spherical cap formed by the vertices p , q , and r of a spherical triangle. Assume that the cone is not empty. This implies that there is some point which lies inside the cone, whose projection on the spherical surface by construction will lie in the interior of $Cap(p, q, r)$. But this is a contradiction, because according to the Delaunay empty spherical cap property this cap is empty. Hence, the cone is empty. \square

Theorem 10. *Let each node u_i construct a spherical Delaunay triangulation of its projected neighbor locations on the spherical surface for a given power level P . If the largest surface area Ω_i^{max} of the spherical cap for node u_i (except for boundary nodes) is no more than $2.72 \cdot R(P)^2$, then the network topology formed by the nodes with the corresponding transmission power levels is guaranteed to be at least one-connected.*

Proof. Let the 3-D cone formed with apex at node u_i , and base as the spherical cap with the largest surface area Ω_i^{max} has an apex angle θ . From simple trigonometry, the surface area of the spherical cap is $2\pi rh$, where $r = R(P) \cdot \sin(\theta/2)$ and $h = R(P) \cdot (1 - \cos(\theta/2))$. Therefore,

$$\begin{aligned}\Omega_i^{max} &= 2\pi rh \\ &= 2\pi R(P)^2 \cdot \sin(\theta/2)(1 - \cos(\theta/2))\end{aligned}\tag{6.1}$$

Now it is easy to verify that if $\Omega_i^{max} \leq 2.72 \cdot R(P)^2$, then $\theta \leq 2\pi/3$. This implies that if the surface area of the largest cap is no more than $2.72 \cdot R(P)^2$, then there is no empty 3-D cone of apex angle greater than $2\pi/3$ around u_i . Since this is true for every node (except the boundary nodes), the network is at least one connected from the result of [12]. □

The implication of the above theorem is that if every node adjusts its power level to have enough neighbors, such that, none of the caps of the spherical Delaunay triangulation has a surface area greater than the threshold mentioned above, then the network will be at least one connected so long as the maximum power graph is connected.

There is a subtlety with boundary nodes while checking for the largest spherical cap. We define a *boundary node* as one that lies *outside* the 3-D convex hull formed by all its neighbors when transmitting at maximum power. A node can identify itself as a boundary node in Phase 1 of the algorithm after running MDS and by constructing a 3-D

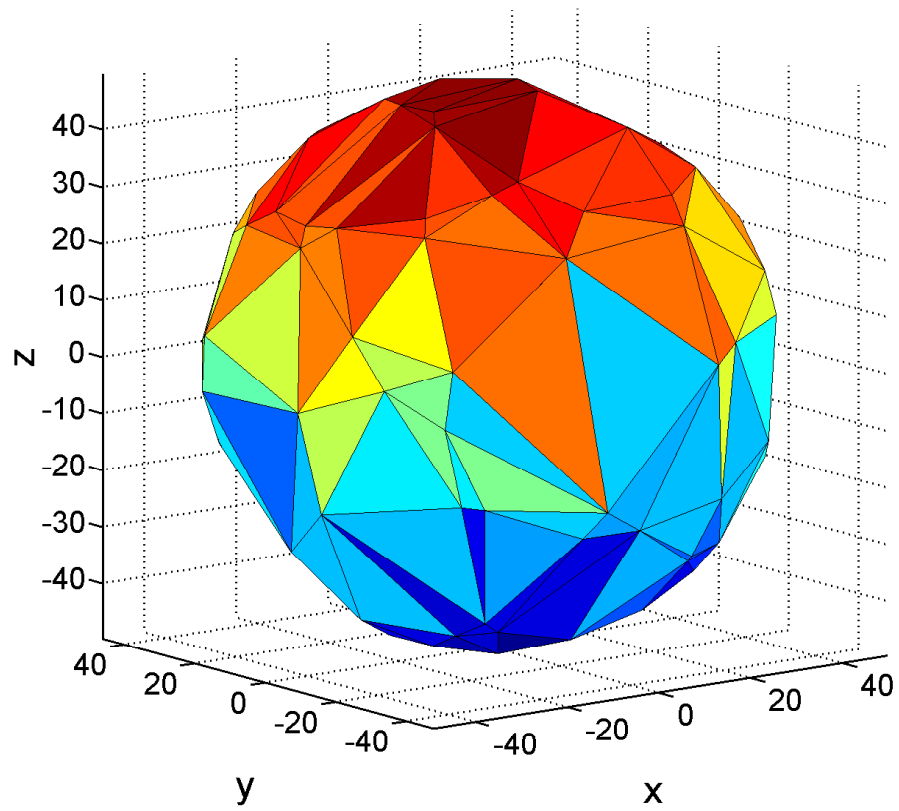


Figure 6.6: Spherical Delaunay triangulation using the Quickhull algorithm of a set of 100 points randomly distributed on the surface of a sphere of radius 50.

Algorithm 7 SPHERICAL DELAUNAY TRIANGULATION-BASED ALGORITHM

1. $N_i(P) \leftarrow \phi$
 2. $P \leftarrow P^{min}$
 3. **while** ($P \leq P^{max}$) **do**
 4. Broadcast “Hello” message at power P and gather “Reply” messages from neighbors;
 5. $N_i(P) \leftarrow N_i(P) \cup \{v \mid \text{node } v \text{ replied}\};$
 6. Project the locations of $N_i(P)$ on the surface of the spherical ball $B(u_i, R)$;
 7. Construct SDT with the projected points;
 8. Find the largest spherical cap surface area, Ω_i^{max} ;
 9. **if** $\Omega_i^{max} > 2.72 \cdot R^2$ **then**
 10. $P \leftarrow \text{increment}(P)$;
 11. **else**
 12. **break**;
 13. **end if**
 14. **end while**
-

convex hull with all the neighbors that lie within its communication range. Each node can perform this in $O(d \log d)$ time using the *Quickhull* algorithm [14], where d is its degree. Since a boundary node does not have neighbors in all directions around itself, its spherical Delaunay triangulation might form caps that are smaller than the threshold area but still have empty 3-D cones pointing outwards. In such cases, a boundary node calculates the difference in surface area of the spherical ball and the sum of the spherical triangles. If this difference is greater than the threshold then it further increases its power level. One advantage of this SDT based approach compared to the CBTC technique is that the boundary nodes do not end up with maximum power levels. Formally, the following steps shown in Algorithm 7 will be executed on each internal node u_i .

Constructing a spherical Delaunay triangulation in Step 7 of the algorithm is equivalent of finding a 3-D convex hull for the set of projected points on the sphere. This

can be done in $O(d \log d)$ time using the Quickhull algorithm [14], where d is number of neighbors (see Figure 6.6 for an illustration). Note that, the number of spherical caps thus generated is of $O(d)$. Hence, the time complexity of the above algorithm is $O(d \log d)$. This is a substantial improvement over the existing algorithms that run in $O(d^3 \log d)$ time. Since the average node degree in 3-D is very high compared to that in 2-D for a network to be connected with high probability under random deployment, an improvement of d^2 implies a much faster algorithm.

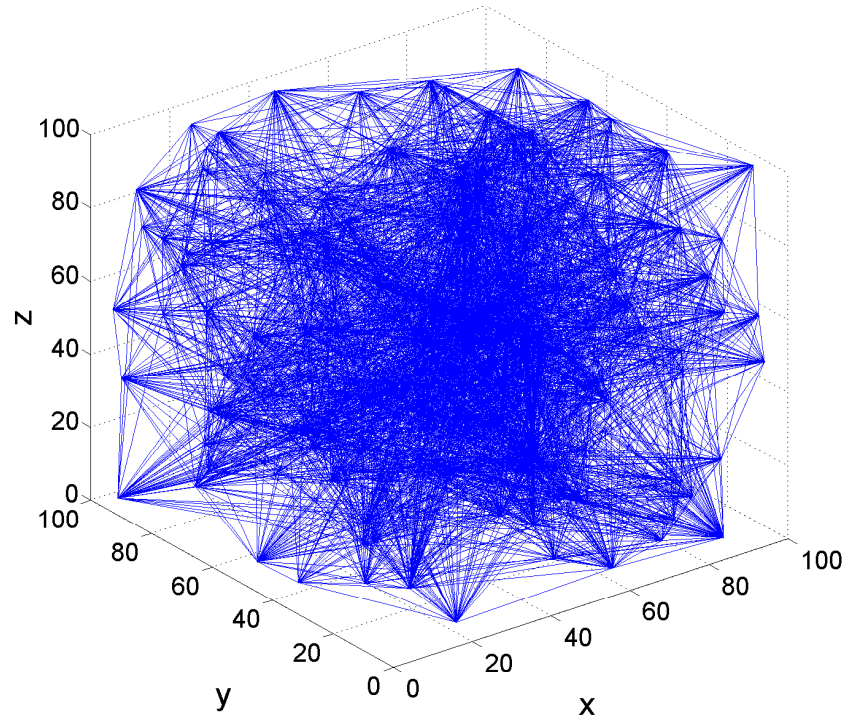
As we mentioned earlier, the orthographic projection based approach, although works very well in practice, does not theoretically guarantee a connected 3-D network, even when the θ constraint is satisfied on all the three planes. The degenerate cases can be identified by combining the two approaches in the following way. Before projecting the neighbors on the spherical surface in Step 6 of Algorithm 7, each node first projects them on the xy , yz , and zx planes. Then it increments its power level until all the three planes satisfy $\theta = 2\pi/3$ constraint, i.e., until the function $gap_\theta(D_i)$ returns *false* for all the three planes. Let this power level be P_{proj} . Only after this, the neighbors are projected on the surface of a sphere of radius $R = f(P_{proj})$ and Steps 7, 8, 9 and so on are followed. If the algorithm terminates with the node choosing a power level that is smaller than its maximum power level, then this is a degenerate case. This is because had the node settled down with its maximum power level P^{max} , it would mean that either the largest spherical cap area is no less than the threshold value of $\Omega_{max} = 2.72 \cdot R^2$. In the first case, the 3-D network will not be connected, whereas in the second case the network

finally formed is, in fact, the original maximum power graph, which was assumed to be connected.

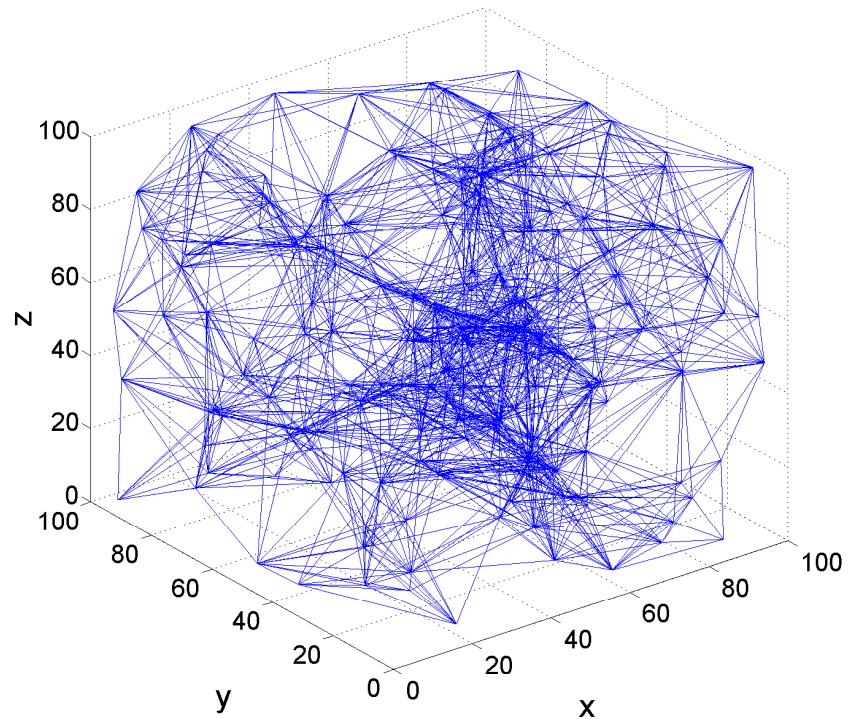
6.6 Evaluation

In order to understand the efficiency of our algorithms, we conducted simulations by generating random networks in 3-D of different sizes inside a cuboid of size $100 \times 100 \times 100$. Before running the projection based or the spherical Delaunay triangulation based algorithms, we run MDS to get relative neighbor locations for each node at their maximum transmission power level.

For the SDT based algorithm, we generated random network topologies for $n = 200$ nodes under different maximum power levels. The performance of our algorithm is measured in terms of the average node degree and average transmission power. As an example, we show one specific instance of the initial network topology (the connected maximum power graph) when $P^{max} = 40$ in Figure 6.7(a). In Figure 6.7(b), we show the same network after all the nodes have settled down with minimal transmission power according to the SDT algorithm. For the same instance, we plotted the initial and final node degrees in Figure 6.8(a). Finally, in Figure 6.8(b), we show the assigned minimal power levels for all the nodes. Note that, node degrees have drastically reduced. We also observe that only 7.5% of the nodes transmit at their maximum power level, and more than 25% of the nodes transmit at less than half the maximum power level (i.e., below 20). Next, we describe the general trends.

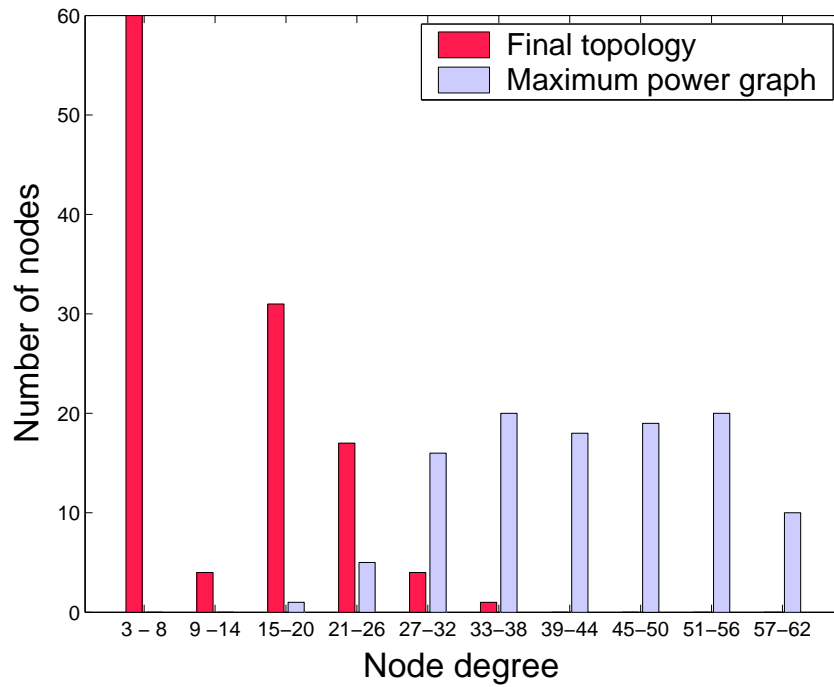


(a) Maximum power graph.

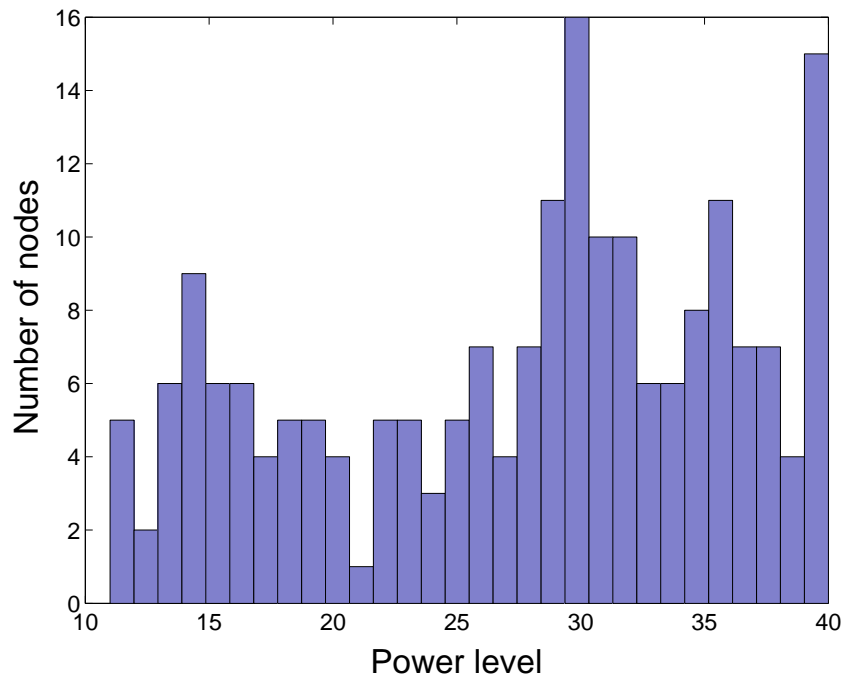


(b) Final connected network topology based on SDT.

Figure 6.7: (a) Network topology of the original maximum power graph. (b) Network topology after running the SDT-based algorithm. Here $n = 200$ and $P^{max} = 40$. Node degrees have drastically reduced.

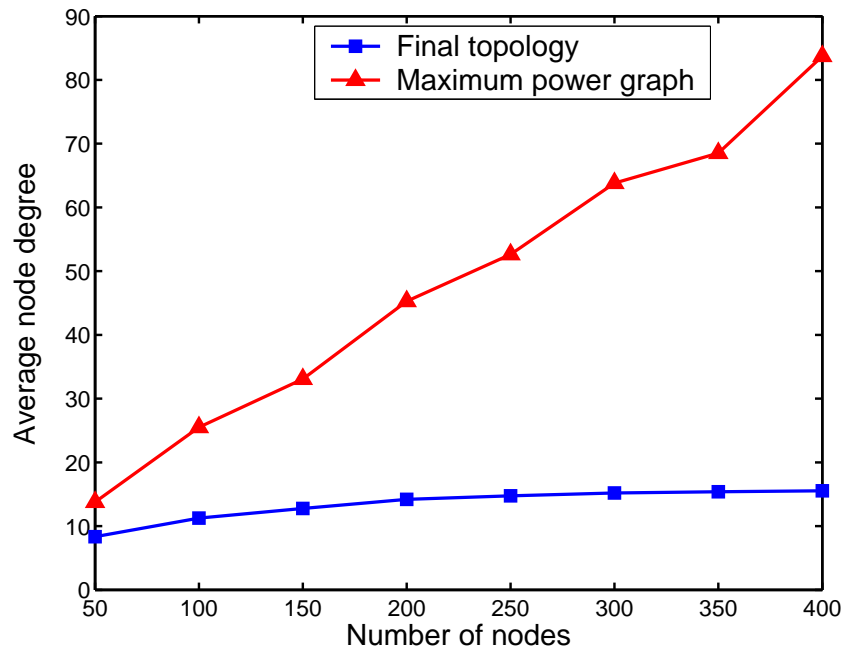


(a)

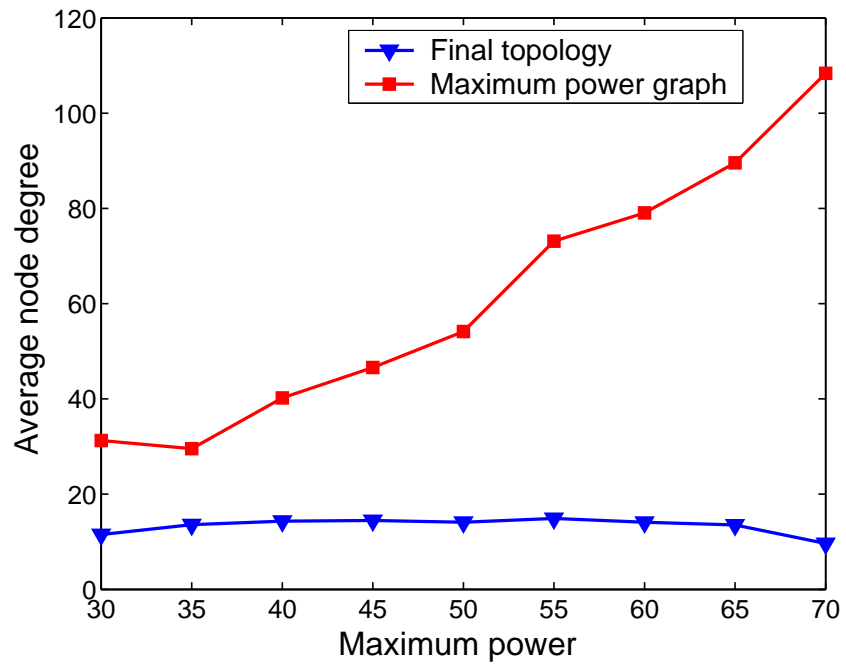


(b)

Figure 6.8: (a) Node degrees of the maximum power graph and that of the final topology for $n = 200$, $P^{max} = 40$. (b) Final assigned minimal transmission power levels of nodes, for $n = 200$, $P^{max} = 40$.



(a)



(b)

Figure 6.9: (a) Dependency of average node degree with network size. (b) Dependency of the average node degree with maximum power, for $n = 200$; node degree remains almost flat in the final topology based on the SDT algorithm.

We measured the dependencies between (i) average node degrees, and (ii) average transmission power, with different network sizes. In Figure 6.9(a), which shows the plot of average node degree with network size for $P^{max} = 40$, we observe that as the network size (density) increases, the average node degree increases sharply for maximum power graphs, whereas increases very slowly for network topologies formed based on the SDT algorithm. Figure 6.10 shows the plot of average transmission power levels with increasing network density for topologies generated based on SDT. We observe that as the network gets denser, the average transmission power decreases. This is expected because as the density increases, nodes can use lower power levels to form more number of multihop paths to guarantee network connectivity. Note that, in Figure 6.9(a) when the number of nodes increases to 400, the average node degree reaches ≈ 15 , which is in accordance with the percolation theory of critical average node degree for network connectivity ($\frac{4\pi}{3} \cdot (0.2)^3 \cdot (400) \approx 13.4$).

In Figure 6.9(b), we plot average node degrees for random topologies of 200 nodes with increasing maximum power levels. We observe that average node degrees almost remain flat in the final topology generated based on the SDT algorithm as compared to that of the maximum power graph. This is because of the fact that as the maximum power varies, only the very small number of boundary nodes get affected, because they are more likely to transmit at or close to the maximum power compared to the internal nodes in order to guarantee a minimal number of neighbors. The average transmission power level chosen by the nodes in the final topology is observed to vary from 26 to 21

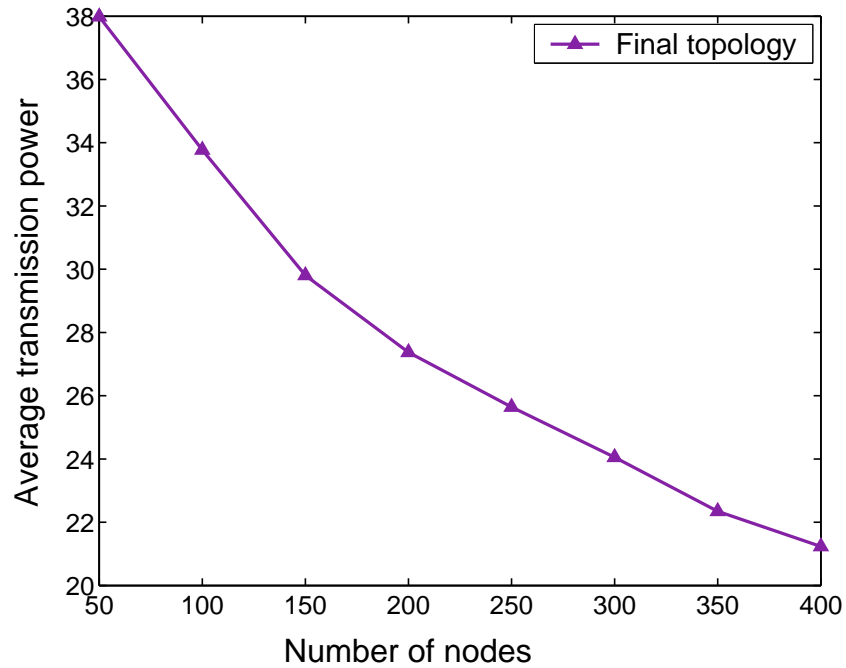


Figure 6.10: Dependency of average transmission power with network size.

as the maximum power increased from 30 to 70 (not plotted). Thus, the effectiveness of the algorithm is predominant at higher maximum power levels.

To compare the SDT based algorithm with the one described in [12] based on the procedure $gap-3D_{\alpha}()$, both of which check for empty 3-D cones, we simulated the algorithms and measured the CPU running times using the MATLAB *profile* tool. Our measurements show that the respective procedures $sdtcheck()$ (part of our implementation) and $gap-3D_{\alpha}()$ take most percentages (80%–90%) of the total execution times. In Figure 6.11, we plot of CPU execution times for these two functions on random network topologies of different sizes with maximum power set to 40. We observed that SDT performs much faster than the other one, and with increasing network size (or equivalently,

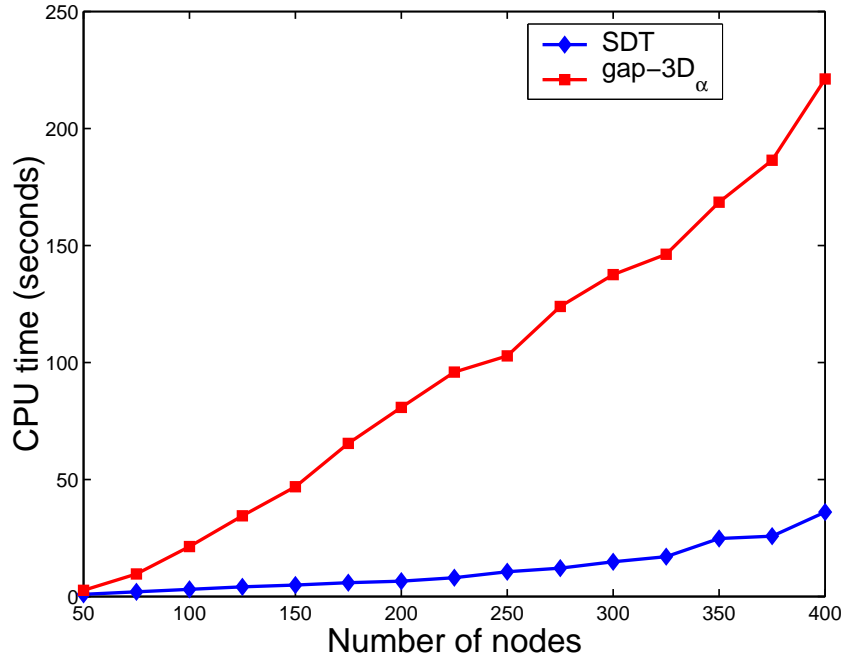
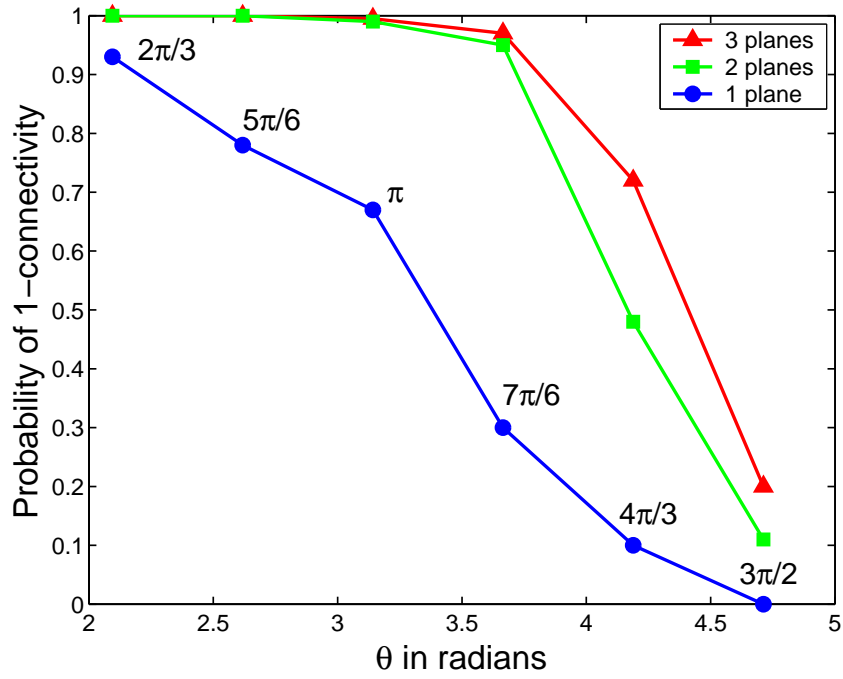


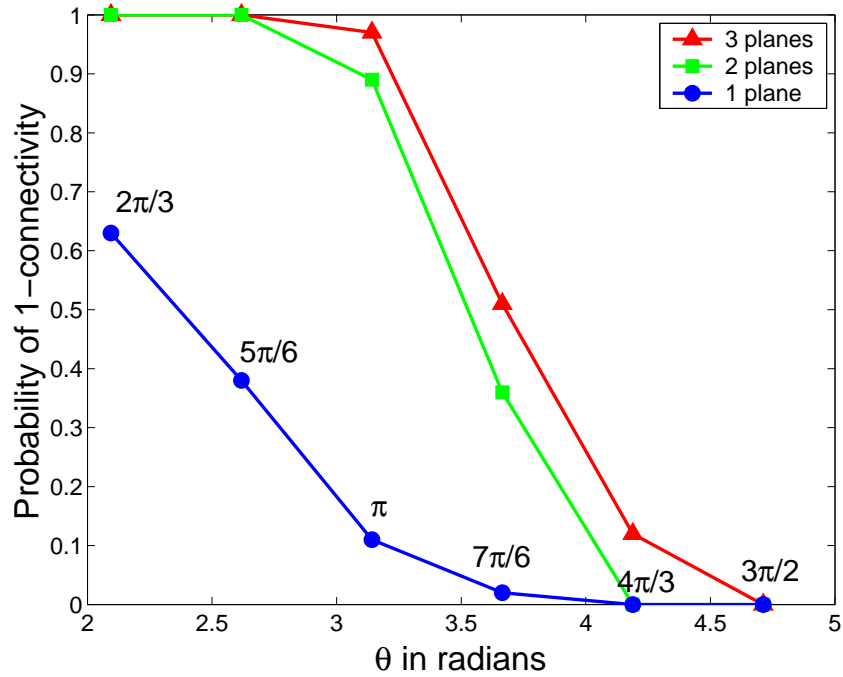
Figure 6.11: CPU execution time of the SDT based algorithm and the one [12] based on the procedure $gap-3D_{\alpha}()$ for different random topologies with $P^{max} = 40$.

with increasing average node degree), the difference between execution times becomes more predominant.

Finally, to measure the practicality of the orthographic projection based algorithm, we generated two random networks with $n = 50$ and 100 nodes, and $P^{max} = 100$. Then we ran the projection based algorithm and generated different network topologies where the θ constraint is satisfied on one, two, or all three planes. For each of these cases, we checked the connectivity of the original 3-D graph. In Figure 6.12, we plot the probability of connectivity with θ . Each point in the plot is averaged over 200 runs. We observe that when the θ constraint is satisfied on only one plane, the original 3-D graph is actually never connected. However, when each node guarantees $\theta = 2\pi/3$ on all three planes, the 3-D topology generated with that power level is found to be connected at *all*



(a) For $n = 50$ nodes



(b) For $n = 100$ nodes

Figure 6.12: Probability of network connectivity as the θ constraint is satisfied on 1, 2, or all 3 orthogonal planes.

times. Moreover, even when the constraint is satisfied on only two planes, we find that the 3-D graph is disconnected only a very small number (less than 1%) of times. This shows that the heuristic based approach works very well in practice.

6.7 Summary

In this chapter, we have designed efficient power control schemes for sensor networks deployed in 3-dimensions where very high density of nodes causes high interference which, in turn, lowers network throughput. We presented two efficient distributed topology control algorithms for 3-D networks. Our first approach is based on the idea of 2-D orthographic projections, by which we reduced and simplified the original 3-D problem into a set of three similar problems on 2-D, and used existing techniques from CBTC to show that network connectivity can be guaranteed almost at all times. In the second approach we used the properties of spherical Delaunay triangulation on the surface of a sphere to determine the existence of empty 3-D cones in an efficient way, and showed that the network topology generated based on this is always connected. Both the algorithms are computationally very efficient and scale as $O(d \log d)$ in time, where d is the average node degree. The d^2 improvement in running time implies a *much faster* algorithm especially in 3-D, as d is typically very high in 3-D compared to that in 2-D, and this is verified by measuring CPU execution times on topologies with increasing node degree. To substantiate our claims we conducted simulations on network topologies in

3-D for both the SDT based algorithm and the heuristic based approach. Doing a probabilistic analysis of network connectivity for the orthographic projection based approach and implementation of the SDT algorithm on real notes are part of our future work.

Chapter 7

Conclusions and Future Work

Data collection is a fundamental operation in wireless sensor networks where sensor nodes measure attributes about a phenomenon of interest and transmit their readings to a common base station. In applications such as structural health monitoring, security surveillance, health care, environmental monitoring, etc., it is often important to deliver data in a timely fashion. In this thesis, we focused on the algorithmic aspects of throughput-delay performance for fast data collection in tree-based sensor networks. We addressed the problem of jointly optimizing these two mutually conflicting performance objectives in the context of aggregated convergecast. Our approach comprised three techniques - (i) multi-channel TDMA scheduling, (ii) routing over optimal topologies, and (iii) transmission power control. Using a combination of these three mechanisms, we showed that it is possible to design efficient algorithms that have provably good worst-case performance bounds. In the following we briefly describe our contributions and discuss some future research directions and open problems.

7.1 Multi-Channel TDMA Scheduling

In Chapter 3, we exploited the benefits of multiple frequency channels and designed approximation algorithms for maximizing the aggregated convergecast throughput under a graph-based interference model. We considered two cases with respect to the knowledge of the routing topology. First, when the routing topology is known a priori, and second when the routing topology is unknown and can be arbitrary. We decoupled the *joint frequency and time slot assignment problem* into two separate subproblems of frequency assignment and time slot assignment, and showed that this decoupling still leads to good approximation ratios.

For the case of known routing topology, we considered two random geometric graph models : (i) unit disk graphs, where nodes have uniform transmission range, and (ii) general disk graphs, where nodes have different transmission ranges. The first one corresponds to a network where all the nodes transmit at a uniform power level, whereas the second one might correspond to a network where nodes can control their transmission power. Benefits of controlling transmission have been described in detail earlier in Chapter 2. To assign frequencies in the UDG model, we proposed a *receiver-based greedy frequency assignment strategy*, where the receivers of the given routing tree are assigned frequencies in such a way that the maximum number of edges transmitting on the same frequency is minimized. This corresponds to a suboptimal *load-balanced* frequency assignment which gives a *constant factor* approximation with respect to max-min load on any frequency. Our time slot assignment strategy is also greedy, where we showed that

the schedule length, which is an indicator of the aggregated convergecast throughput, is within a constant factor of the optimal for unit disk graph models. For the general disk graph model, we formulated the two subproblems as integer linear programs, and by using a randomized rounding strategy we showed that the overall approximation ratio of the schedule length is logarithmic with respect to the number of nodes in the network. Lastly, for the case of unknown routing topology, we proved that the approximation ratios still hold so long as the maximum node degree of any node in the tree is bounded by a constant. Simulation results show significant improvement when using multiple frequencies and underscores their benefits in scheduling. These results are perhaps the first ones in their category of designing provably good approximation algorithms with multiple frequencies for arbitrarily deployed networks where fast data collection is the primary motivation.

The algorithms proposed in Chapter 3, however, are limited to only graph-based interference models, which are somewhat idealistic in nature and does not capture the cumulative interference from many simultaneous transmitters that are located far away. To address this issue, in Chapter 4, these multi-channel scheduling algorithms are extended for the more realistic SINR-based interference model. By using the notion of *link diversity*, and by suitably reusing time slots, we showed that the modified algorithms still hold a good performance ratio that scales with the number of non-empty length classes. Although theoretically this number can be as large as the number of nodes in the network, in practice, it is a small constant.

7.2 Optimal Routing Topologies

In Chapter 5, we considered delay as another performance objective, and addressed the joint problem of optimizing both throughput and delay in the context of aggregated convergecast. We defined the maximum delay in terms of the largest hop distance from any node to the sink, called the *radius* of the tree, and constructed routing topologies that minimize the radius under a given degree constraint on the nodes. We noted that minimizing the radius and degree of a spanning tree at the same time are mutually conflicting objectives, and discovered that a high node degree creates a bottleneck in the network that limits the aggregated throughput. Accordingly, we formulated the joint optimization problem of both degree and radius as a *bicriteria optimization problem* with the goal to minimize the maximum hop distance in the tree such that the maximum node degree remains within a certain threshold. To this end, we designed an approximation algorithm that constructs a *bounded-degree-minimum-degree spanning tree* with constant factor bicriteria approximation ratios on both the objectives. Although the notion of bicriteria formulation for network design problems is not new, we are the first ones to consider throughput and delay under the same bicriteria optimization framework and designed algorithms with provably good performance bounds.

Once the routing topology is constructed, we evaluated the multi-channel scheduling algorithms proposed in Chapter 3 on such topologies. Our simulation results show

significant improvements on the delay as well as on aggregated throughput on bounded-degree-minimum-radius spanning trees; it gives the best of both worlds in terms of maximizing throughput and minimizing delay as compared to some of the other more commonly used routing topologies, such as minimum spanning trees where the radius is low but the average degree of a node is very high, and minimum interference trees where the radius is large but the average node degree is low.

7.3 Transmission Power Control

In Chapter 6, we considered sensor networks deployed 3-dimensions and designed efficient power control schemes from local geometric information in order to construct sparse topologies while maintaining network connectivity. Although the amount of literature in topology control in 2-D is enormous, there exist few works that address under a 3-D setting. Moreover, the existing solutions require directional information and incur a lot of computational overhead. As the density of nodes required to maintain a connected network in 3-D is very high, it negatively impacts the achievable throughput due to increased interference, and so with a sparser topology is likely help. Our contribution in this respect is two fold. We extended some of the existing works on topology control for 2-D to be applicable in a 3-D setting. In particular, we used orthographic projections to extend the well known cone-based topology control algorithm, and showed that although network connectivity cannot be theoretically guaranteed, it performs very well in practice. In addition, we proposed a robust new technique based on spherical Delaunay

triangulation that guarantees network connectivity and is superior to existing techniques in terms of overhead and computational requirements.

7.4 Future Work

7.4.1 Real Testbed Implementation

Although we evaluated our proposed algorithms using simulations, real implementation on sensor nodes where schedules are computed locally and are adaptive to network dynamics are necessary to enhance the operation of sensor networks and to meet application requirements. For instance, we observe a trend in using WSNs to support more complex operations ranging from industrial control to health care, which require complex operations like detection of events in real-time, or responsive querying of the network by collecting streams of data in a timely manner. Thus, supporting QoS metrics such as delay and reliability become more important. Therefore, distributed implementation and performance testing of the proposed algorithms on testbed or real deployments becomes essential. Additionally, real implementation and deployment will help in addressing the problems of intermittent connectivity and channel errors with unreliable links and handling asymmetric links.

7.4.2 Other Joint Objectives

Although there exist some works that address multiple joint objectives, more detailed investigations to address the trade-offs between conflicting objectives will be beneficial. Most of the studies consider the trade-offs between energy efficiency and latency objectives. Different trade-offs can be identified between other objectives, such as minimizing latency and maximizing reliability, or maximizing capacity and minimizing energy consumption. For instance, with the extension of WSNs in the visual domain where embedded cameras act as sensors, criteria such as reliability, QoS, and timeliness of the streamed data are becoming important. Solutions to address different objectives and trade-offs, for instance, consumed energy versus reconstructed image quality, should be explored within the perspective of data collection in WSNs. In addition, our proposed tree construction is applicable only when nodes transmit at a uniform power level. Extension of these results when nodes transmit at different power can be useful.

7.4.3 Traffic Patterns

We have considered fixed traffic patterns where every node generates a fixed number of packets (one in our case) in each data collection cycle. In real scenarios, some nodes may have a lot of packets that require more than one time slot per frame, while some others may not have any data to send in a time slot, thus wasting bandwidth. It will be interesting to explore the performance in such scenarios with random packet arrivals

and combining the solutions of TDMA scheduling with rate allocation algorithms, especially in applications where high data rates are necessary. In the proposed algorithms, we have considered data collection under perfect aggregation where a node can aggregate all the packets coming from its children as well as its own into a single packet before transmitting to its parent. Another possibility is to investigate different levels of aggregation, i.e., how much of the data received from the children is forwarded to the parent node. Investigating different levels of aggregation was proposed in the literature where the efficiency of different tree construction mechanisms were analyzed in terms of latency and energy metrics. This study can be extended for TDMA-based data collection algorithms in WSNs. Lastly, we have considered tree-based sensor networks where the routing tree is fixed for every packet traversal to the sink. It will be interesting to evaluate the throughput-delay performance where packets from a given source nodes can take different paths depending on network conditions.

7.4.4 Cross Layer Solutions

Our proposed algorithms provide cross layer solutions in some sense, where the schedules are computed together with transmission power control, optimal routing topologies, and multi-frequency scheduling. It is indeed essential to address the problems from a cross layer perspective to achieve the target functions and offer better performance. Along this line of research, Chafekar *et al.* in [20] extended the work by

Moscibroda [112] in designing cross-layer protocols using the SINR model and proposed polynomial time algorithms with provable worst-case performance guarantee for the latency minimization problem. Their cross-layer approach chooses power level for all transceivers, routes for all connections, and constructs an end-to-end schedule such that SINR constraints are satisfied. A prominent research direction is to consider such cross-layer approaches from a theoretical point of view. More research can be done in this direction to combine the existing work with the solutions at different layers. In most studies, static topologies are assumed. Problems related to dynamic topologies, such as topological changes and addition of new nodes are open. In addition, the time complexity of data gathering under various network conditions, such as when some nodes have no packet to transmit, or when no buffering is allowed remain open.

7.4.5 Realistic Interference Models

As was pointed by Moscibroda in [111], the type of interference model may heavily impact the achievable results. Use of realistic models for communication and interference is another direction that can be further investigated. Along this direction, Goussevskaia *et al.* [56] present the first NP-completeness proofs (by reducing from the Partition problem) on two scheduling problems using the SINR interference model. The first problem consists in finding a minimum-length schedule for a given set of links. The second problem receives a weighted set of links as input and consists in finding a maximum-weight subset of links to be scheduled simultaneously in one shot. In [111, 112], Moscibroda

et al. study a generalized version of the SINR interference model and obtain theoretical upper bounds on the scheduling complexity of arbitrary topologies. They prove that if signals are transmitted with correctly assigned transmission power levels, the number of time slots required to successfully schedule all links in an arbitrary topology is proportional to the squared logarithm of the number of nodes times a previously defined static interference measure. More of such works that bridge the gap between static graph-based interference models and the physical models are needed.

References

- [1] H. Abdi. Metric Multidimensional Scaling (MDS): Analyzing Distance Matrices 1 Overview. *Encyclopedia of Measurement and Statistics*, (Ed.) N.J. Salkind, 2007.
- [2] I. F. Akyildiz, D. Pompili, and T. Melodia. Underwater Acoustic Sensor Networks: Research Challenges. *Ad Hoc Networks*, 3(3):257–279, 2005.
- [3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey on Sensor Networks. *IEEE Communications Magazine*, 40(8):102–114, 2002.
- [4] G. Alnifie and R. Simon. A multi-channel defense against jamming attacks in wireless sensor networks. In *3rd ACM workshop on QoS and security for wireless and mobile networks (Q2SWinet)*, pages 95–104, October 2007.
- [5] V. Annamalai, S. K. S. Gupta, and L. Schwiebert. On tree-based convergecasting in wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1942–1947, March 2003.
- [6] ANRG. Autonomous Networks Research Group. <http://anrg.usc.edu>.
- [7] Atmel. AT91RM9200. http://www.atmel.com/dyn/resources/prod_documents/1768s.pdf.
- [8] F. Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, 1991.
- [9] B. Awerbuch. Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems. In *19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 230–240, 1987.
- [10] A. Baggio. Wireless sensor networks in precision agriculture. In *Workshop on real-world wireless sensor networks (REALWSN)*, pages 106–108, June 2005.
- [11] P. Bahl, R. Chandra, and J. Dunagan. SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *10th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 216–230, October 2004.

- [12] M. Bahramgiri, M. Hajiaghayi, and V. S. Mirrokni. Fault-tolerant and 3-dimensional distributed topology control algorithms in wireless multi-hop networks. *Wireless Networks*, 12(2):179–188, 2006.
- [13] H. Baldus, K. Klabunde, and G. Musch. Reliable set-up of medical body-sensor networks. In *1st European Workshop on Wireless Sensor Networks (EWSN)*, pages 353–363, 2004.
- [14] B. C. Barber, D. P. Dobkin, and H. Huhdanpaa. The Quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.
- [15] J. Beutel, S. Gruber, A. Hasler, R. Lim, A. Meier, C. Plessl, I. Talzi, L. Thiele, C. Tschudin, M. Woehrle, and M. Yuecel. PermaDAQ: A Scientific Instrument for Precision Sensing and Data Recovery in Environmental Extremes. In *8th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 265–276, 2009.
- [16] K. Bharat-Kumar and J. Jaffe. Routing to Multiple Destinations in Computer Networks. *IEEE Transactions on Communications*, 31(3):343–351, 1983.
- [17] M. Burkhart, P. von Rickenbach, R. Wattenhofer, and A. Zollinger. Does topology control reduce interference? In *5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 9–19, 2004.
- [18] M. Caccamo, L.Y. Zhang, S. Lui, and G. Buttazzo. An implicit prioritized access protocol for wireless sensor networks. In *23rd IEEE Real-Time Systems Symposium (RTSS)*, pages 39–48, December 2002.
- [19] Berkeley Wireless Research Center. PicoRadio. http://bwrc.eecs.berkeley.edu/research/pico_radio.
- [20] D. Chafekar, V. S. Anil Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. Cross-layer latency minimization in wireless networks with SINR constraints. In *8th ACM International Symposium on Mobile Ad-hoc Networking and Computing (MobiHoc)*, pages 110–119, September 2007.
- [21] S. Chatterjea. *Distributed and Self-Organizing Data Management Strategies for Wireless Sensor Networks. A Cross-Layered Approach*. PhD thesis, University of Twente, Enschede, the Netherlands, September 2008.
- [22] K. Chaudhuri, S. Rao, S. Riesenfeld, and K. Talwar. A push-relabel approximation algorithm for approximating the minimum-degree MST problem and its generalization to matroids. *Theoretical Computer Science*, 410(44):4489–4503, 2009.

- [23] K. Chaudhuri, S. Rao, S. Riesenfeld, and K. Talwar. What Would Edmonds Do? Augmenting Paths and Witnesses for Degree-Bounded MSTs. *Algorithmica*, 55(1):157–189, 2009.
- [24] J. Chen, S. Sheu, and C. Yang. A new multichannel access protocol for IEEE 802.11 ad hoc wireless LANs. In *14th IEEE Personal, Indoor and Mobile Radio Communications Symposium (PIMRC)*, pages 2291–2296, September 2003.
- [25] X. Chen, X. Hu, and J. Zhu. Minimum Data Aggregation Time Problem in Wireless Sensor Networks. In *1st International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, pages 133–142, December 2005.
- [26] K. Chintalapudi, T. Fu, J. Paek, N. Kothari, S. Rangwala, J. Caffrey, R. Govindan, E. Johnson, and S. Masri. Monitoring Civil Structures with a Wireless Sensor Network. *IEEE Internet Computing*, 10(2):26–34, 2006.
- [27] I. Chlamtac and S. Kutten. Tree-Based Broadcasting in Multihop Radio Networks. *IEEE Transactions on Computers*, 36(10):1209–1233, 1987.
- [28] H. Choi, J. Wang, and E. A. Hughes. Scheduling for information gathering on sensor network. *Wireless Networks*, 15(1):127–140, 2009.
- [29] J. Crichigno, M. Wu, and W. Shu. Protocols and architectures for channel assignment in wireless mesh networks. *Ad Hoc Networks*, 6(7):1051–1077, 2008.
- [30] Crossbow. MICAz, Mica2 - Wireless Measurement System. <http://www.xbow.com>.
- [31] R. L. Cruz and A. Santhanam. Optimal Routing, Link Scheduling , and Power Control in Multi-hop Wireless Networks. In *22nd IEEE International Conference on Computer Communications (INFOCOM)*, pages 702–711, April 2003.
- [32] DARPA. SensIT Project. <http://dtsn.darpa.mil/ixo/sensit.asp>.
- [33] MIT Distributed Robotics Laboratory. Flood Early Warning System. <http://groups.csail.mit.edu/drl/wiki/index.php/floodews>.
- [34] F. J. Dyson. *Infinite in All Directions*. Cornelia and Michael Bessie Books, New York, 1988.
- [35] T. A. ElBatt and A. Ephremides. Joint Scheduling and Power Control for Wireless Ad-hoc Networks. In *21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 976–984, June 2002.
- [36] J. Elson and D. Estrin. Sensor networks: A bridge to the physical world (Book Chapter). *Wireless sensor networks*, Kluwer Academic Publishers, pages 3–20, 2004.

- [37] S. C. Ergen and P. Varaja. TDMA scheduling algorithms for sensor networks. Technical report, University of California, Berkeley, July 2005.
- [38] R. J. Faudree, A. Gyárfás, R. H. Schelp, and Z. Tuza. The Strong Chromatic Index of Graphs. *Ars Combinatoria*, B(29):205–211, 1990.
- [39] R. P. Feynman. There’s plenty of room at the bottom. *Microelectromechanical Systems*, 1(1):60–66, 1992.
- [40] C. Florens, M. Franceschetti, and R. J. McEliece. Lower bounds on data collection time in sensory networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1110–1120, 2004.
- [41] C. Florens and R. McEliece. Scheduling algorithms for wireless ad-hoc sensor networks. In *IEEE Global Communications Conference (GLOBECOM)*, pages 6–10, November 2002.
- [42] C. Florens and R. McEliece. Packets Distribution Algorithms for Sensor Networks. In *22nd IEEE International Conference on Computer Communications (INFOCOM)*, pages 1063–1072, April 2003.
- [43] T. Fountain, S. Tilak, P. Shin, S. Holbrook, R. J. Schmitt, A. Brooks, L. Washburn, and D. Salazar. Digital Moorea Cyberinfrastructure for Coral Reef Monitoring. In *5th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, December 2009.
- [44] M. Fürer and B. Raghavachari. Approximating the minimum degree spanning tree to within one from the optimal degree. In *3rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 317–324, January 1992.
- [45] M. Fürer and B. Raghavachari. Approximating the minimum-degree Steiner tree to within one of optimal. *Journal of Algorithms*, 17(3):409–423, 1994.
- [46] S. Gandham, Y. Zhang, and Q. Huang. Distributed Minimal Time Convergecast Scheduling in Wireless Sensor Networks. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 50–57, July 2006.
- [47] S. Gandham, Y. Zhang, and Q. Huang. Distributed time-optimal scheduling for convergecast in wireless sensor networks. *Computer Networks*, 52(3):610–629, 2008.
- [48] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, 1990.
- [49] B. Gates. The disappearing computer. *The Economist*, December 2002.

- [50] A. Ghosh, O. D. Incel, V. S. Anil Kumar, and B. Krishnamachari. Multi-channel scheduling algorithms for fast aggregated convergecast in sensor networks. In *6th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 363–372, October 2009.
- [51] A. Ghosh, O. D. Incel, V. S. Anil Kumar, and B. Krishnamachari. Multi-Channel Scheduling and Spanning Trees: Throughput-Delay Trade-off for Fast Data Collection in Sensor Networks (under submission). *IEEE/ACM Transactions on Networking*, 2010.
- [52] A. Ghosh, O. D. Incel, V. S. Anil Kumar, and Bhaskar Krishnamachari. Optimal Spanning Trees for Fast data Collection in Sensor Networks (poster). In *29th Annual IEEE Conference on Computer Communications (INFOCOM)*, March 2010.
- [53] A. Ghosh, Y. Wang, and B. Krishnamachari. Efficient Distributed Topology Control in 3-Dimensional Wireless Networks. In *4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 91–100, June 2007.
- [54] A. Giannoulis, T. Salonidis, and E. Knightly. Congestion Control and Channel Assignment in Multi-Radio Wireless Mesh Networks. In *5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 350–358, June 2008.
- [55] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 1–14, November 2009.
- [56] O. Goussevskaia, Y. A. Oswald, and R. Wattenhofer. Complexity in Geometric SINR. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, September 2007.
- [57] R. L. Graham. Bounds on Multiprocessing Timing Anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, March 1969.
- [58] A. Gupta, C. Gui, and P. Mohapatra. Exploiting Multi-Channel Clustering for Power Efficiency in Sensor Networks. In *1st International Conference on Communication System Software and Middleware (COMSWARE)*, pages 1–10, January 2006.
- [59] P. Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000.
- [60] M. Hamalainen, P. Pirinen, and Z. Shelby. Advanced Wireless ICT Healthcare Research. *16th IST Mobile and Wireless Communications Summit*, pages 1–5, July 2007.

- [61] B. Han, V. S. Anil Kumar, M. Marathe, S. Parthasarathy, and A. Srinivasan. Distributed Strategies for Channel Allocation and Scheduling in Software-Defined Radio Networks. In *28th IEEE Conference on Computer Communications (INFOCOM)*, April 2009.
- [62] N. Harvey, R. E. Ladner, L. Lovász, and T. Tamir. Semi-matchings for bipartite graphs and load balancing. *Journal of Algorithms*, 59(1):53–78, 2006.
- [63] A. Hasler, I. Talzi, J. Beutel, C. Tschudin, and S. Gruber. Wireless Sensor Networks in Permafrost Research - Concept, Requirements, Implementation and Challenges. In *9th International Conference on Permafrost*, pages 669–674, July 2008.
- [64] R. Hassin and A. Levin. An Efficient Polynomial Time Approximation Scheme for the Constrained Minimum Spanning Tree Problem Using Matroid Intersection. *SIAM Journal on Computing*, 33(2):261–268, 2004.
- [65] R. Hassin and A. Tamir. On the minimum diameter spanning tree problem. *Information Processing Letters*, 53(2):109–111, 1995.
- [66] W. R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, pages 174–185, 1999.
- [67] J. M. Ho and D. T. Lee. Bounded diameter minimum spanning trees and related problems. In *5th Annual Symposium on Computational Geometry (SCG)*, pages 276–282, June 1989.
- [68] O. D. Incel, A. Ghosh, and B. Krishnamachari. Scheduling Algorithms for Tree-Based Data Collection in Wireless Sensor Networks (Book Chapter). *Theoretical Aspects of Distributed Computing in Sensor Networks*, Springer, Ed. Sotiris Nicotseas, Jose Rolim, 2010.
- [69] O. D. Incel, A. Ghosh, B. Krishnamachari, and K. Chintalapudi. Fast Data Collection in Tree-Based Wireless Sensor Networks (under revision). *IEEE Transactions on Mobile Computing*, 2009.
- [70] O. D. Incel and B. Krishnamachari. Enhancing the Data Collection Rate of Tree-Based Aggregation in Wireless Sensor Networks. In *5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 569–577, June 2008.
- [71] Texas Instruments. CC1000 Single Chip Very Low Power RF Transceiver. <http://focus.ti.com/lit/ds/symlink/cc1000.pdf>.

- [72] Texas Instruments. CC2420 Single-Chip 2.4 GHz IEEE 802.15.4 Compliant and ZigBee(TM) Ready RF Transceiver. <http://focus.ti.com/lit/ds/symlink/cc2420.pdf>.
- [73] Texas Instruments. MSP430 Ultra-Low-Power Microcontrollers. <http://focus.ti.com/lit/wp/slay014/slay014.pdf>.
- [74] N. Jain, S. Das, and A. Nasipuri. A Multichannel CSMA MAC protocol with Receiver-Based Channel Selection for MultiHop Wireless Networks. In *10th IEEE International Conference on Computer Communications and Networks (IC3N)*, pages 432–439, October 2001.
- [75] X. Ji and H. Zha. Sensor Positioning in Wireless Ad-hoc Sensor Networks Using Multidimensional Scaling. In *23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 2652–2661, March 2004.
- [76] D. B. Johnson, D. A. Maltz, and J. Broch. *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks*, pages 139–172. Addison-Wesley, 2001.
- [77] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Mobile Networking for Smart Dust, booktitle = 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom), year = 1999, month = August, location = Seattle, WA.
- [78] R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman. Advances in packet radio technology. *Proceedings of the IEEE*, 66(11):1468–1496, November 1978.
- [79] I. Katzela and M. Naghshineh. Channel Assignment Schemes for Cellular Mobile Telecommunications: A Comprehensive Survey. *IEEE Personal Communications*, pages 10–31, June 1996.
- [80] S. Khuller, B. Raghavachari, and N. Young. Balancing minimum spanning and shortest path trees. In *4th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 243–250, 1993.
- [81] Y. Kim, H. Shin, and H. Cha. Y-MAC: An Energy-Efficient Multi-channel MAC Protocol for Dense Wireless Sensor Networks. In *7th international conference on Information processing in sensor networks (IPSN)*, pages 53–63, April 2008.
- [82] P. Klein, A. Agrawal, R. Ravi, and S. Rao. Approximation through multicommodity flow. In *31st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 726–737, October 1990.

- [83] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos. Multicast routing for multimedia communication. *IEEE/ACM Transaction on Networking*, 1(3):286–292, 1993.
- [84] J. Könemann, A. Levin, and A. Sinha. Approximating the Degree-Bounded Minimum Diameter Spanning Tree Problem. *Algorithmica*, 41(2):117–129, 2005.
- [85] J. Könemann and R. Ravi. A matter of degree: improved approximation algorithms for degree-bounded minimum spanning trees. In *32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 537–546, 2000.
- [86] B. Krishnamachari, S. B. Wicker, R. Béjar, and M. Pearlman. Critical Density Thresholds in Distributed Wireless Networks. In *Communications, Information and Network Security*. Kluwer Publishers, 2002.
- [87] M. Kubisch, H. Karl, A. Wolisz, L.C. Zhong, and J. Rabaey. Distributed algorithms for transmission power control in wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 558–563, March 2003.
- [88] V. S. Anil Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. End-to-end packet-scheduling in wireless ad-hoc networks. In *15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1021–1030, January 2004.
- [89] P. Kyasanur and N. H. Vaidya. Capacity of multi-channel wireless networks: impact of number of channels and interfaces. In *11th annual international conference on Mobile computing and networking (MOBICOM)*, pages 43–57, August 2005.
- [90] N. Lai, C. King, and C. Lin. On Maximizing the Throughput of Convergecast in Wireless Sensor Networks. In *3rd International Conference on Advances in Grid and Pervasive Computing (GPC)*, pages 396–408, May 2008.
- [91] F. T. Leighton. A Graph Coloring Algorithm for Large Scheduling Problems. *Journal of Research of the National Bureau of Standards*, 84(6):489–506, 1979.
- [92] J. Li, Z. J. Haas, M. Sheng, and Y. Chen. Performance evaluation of modified IEEE 802.11 MAC for multi-channel multi-hop ad hoc network. In *17th International Conference on Advanced Information Networking and Applications (AINA)*, pages 312–317, March 2003.
- [93] L. Li, J. Y. Halpern, P. Bahl, Y.-M. Wang, and R. Wattenhofer. A cone-based distributed topology-control algorithm for wireless multi-hop networks. *IEEE/ACM Transaction on Networking*, 13(1):147–159, 2005.

- [94] X.-Y. Li and Y. Wang. Simple Heuristics and PTASs for Intersection Graphs in Wireless Ad Hoc Networks. In *6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, pages 62–71, 2002.
- [95] C. M. Liang, R. Musaloiu, and A. Terzis. Typhoon: A Reliable Data Dissemination Protocol for Wireless Sensor Networks. In *5th European Conference on Wireless Sensor Networks (EWSN)*, pages 268–285, January 2008.
- [96] G. Lu and B. Krishnamachari. Minimum latency joint scheduling and routing in wireless sensor networks. *Ad Hoc Networks*, 5(6):832–843, 2007.
- [97] G. Lu, B. Krishnamachari, and C. S. Raghavendra. An adaptive energy-efficient and low-latency MAC for tree-based data gathering in sensor networks. *Wireless Communications and Mobile Computing*, 7(7):863–875, 2007.
- [98] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a Tiny AGgregation service for ad-hoc sensor networks. *SIGOPS Operating Systems Review*, 36(SI):131–146, 2002.
- [99] R. Maheshwari, H. Gupta, and S. R. Das. Multichannel MAC Protocols for Wireless Networks. In *3rd Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 393–401, September 2006.
- [100] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless sensor networks for habitat monitoring. In *1st ACM international workshop on Wireless sensor networks and applications (WSNA)*, pages 88–97, September 2002.
- [101] B. Malhotra, I. Nikolaidis, and M. A. Nascimento. Aggregation Convergecast Scheduling in Wireless Sensor Networks. Technical report, University of Alberta, 2009.
- [102] J. Mao, Z. Wu, and X. Wu. A TDMA scheduling scheme for many-to-one communications in wireless sensor networks. *Computer Communications*, 30(4):863–872, 2007.
- [103] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt, III. Bicriteria network design problems. *Journal of Algorithms*, 28(1):142–171, July 1998.
- [104] C. Meesookho, S. Narayanan, and C. Raghavendra. Collaborative classification applications in sensor networks. In *2nd IEEE Sensor Array and Multichannel Signal Processing Workshop*, pages 370–374, August 2002.

- [105] S. Mertens. The Easiest Hard Problem: Number Partitioning. *Condensed Matter*, October 2003.
- [106] Sun Microsystems. Sun SPOT: Sun Small Programmable Object Technology. <http://sunspotworld.com>.
- [107] J. Misra and D. Gries. A constructive proof of Vizing's Theorem. *Information Processing Letters*, 41(3):131–133, 1992.
- [108] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
- [109] J. Mo, H. Sheung, W. So, and J. Walrand. Comparison of MultiChannel MAC Protocols. In *8-th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, pages 209–218, October 2005.
- [110] A. H. Mohsenian and V. W. S. Wong. Joint Optimal Channel Assignment and Congestion Control for Multi-channel Wireless Mesh Networks. In *IEEE International Conference on Communications (ICC)*, pages 1984–1989, June 2006.
- [111] T. Moscibroda, R. Wattenhofer, and Y. Weber. Protocol Design Beyond Graph-Based Models. In *5th Workshop on Hot Topics in Networks (HotNets)*, November 2006.
- [112] T. Moscibroda, R. Wattenhofer, and A. Zollinger. Topology control meets SINR: The scheduling complexity of arbitrary topologies. In *7th ACM International Symposium on Mobile Ad-hoc Networking and Computing (MobiHoc)*, pages 310–321, May 2006.
- [113] L. Nachman, R. Kling, R. Adler, J. Huang, and V. Hummel. The Intel Mote platform: A Bluetooth-based sensor network for industrial monitoring. In *4th International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 437–442, April 2005.
- [114] NCTTA. National Collegiate Table Tennis Association. <http://nctta.org>.
- [115] O. Ore. *The Four-Color Problem*. Academic Press, 1967.
- [116] M. Pan and Y. Tseng. Quick convergecast in ZigBee beacon-enabled tree-based wireless sensor networks. *Computer Communications*, 31(5):999–1011, 2008.
- [117] C. H. Papadimitriou. The complexity of the capacitated tree problem. *Networks*, 8(3):217–230, 1978.

- [118] S. Park, A. Savvides, and M. B. Srivastava. SensorSim: A simulation framework for sensor networks. In *3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems (MSWIM)*, pages 104–111, August 2000.
- [119] S. Poduri, S. Patten, B. Krishnamachari, and G. S. Sukhatme. Sensor Network Configuration and the Curse of Dimensionality. In *3rd IEEE Workshop on Embedded Networked Sensors (EmNets)*, May 2006.
- [120] D. O. Popa, H. E. Stephanou, C. Helm, and A. C. Sanderson. Robotic deployment of sensor networks using potential fields. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 642–647, April 2004.
- [121] G. J. Pottie. Wireless integrated network sensors (WINS): The web gets physical. *National Academy of Engineering: The Bridge*, 31(4):22–27, 2001.
- [122] H. Ramamurthy, B. S. Prabhu, R. Gadh, and A. M. Madni. Wireless Industrial Monitoring and Control Using a Smart Sensor Platform. *IEEE Sensors Journal*, 7(5):611–618, May 2007.
- [123] N. Ramanathan, L. Balzano, M. Burt, D. Estrin, T. Harmon, C. Harvey, J. Jay, E. Kohler, S. Rothenberg, and M. Srivastava. Rapid deployment with confidence: Calibration and fault detection in environmental sensor networks. Technical report, Center for Embedded Networked Sensing, UCLA and Department of Civil and Environmental Engineering, MIT, 2006.
- [124] R. Ravi. Rapid rumor ramification: Approximating the minimum broadcast time. In *35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 202–213, Washington, DC, 1994.
- [125] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt, III. Many birds with one stone: Multi-objective approximation algorithms. In *25 Annual ACM Symposium on Theory of Computing (STOC)*, pages 438–447, 1993.
- [126] R. J. Renka. Algorithm 772: STRIPACK: Delaunay triangulation and Voronoi diagram on the surface of a sphere. *ACM Transactions on Mathematical Software*, 23(3):416–434, 1997.
- [127] Ski Resorts. Big Bear Mountains. <http://www.bearmountain.com>.
- [128] S. Roy, A. Das, R. Vijayakumar, H. Alazemi, H. Ma, and E. Alotaibi. Capacity Scaling with Multiple Radios and Multiple Channels in Wireless Mesh Networks. In *1st IEEE Workshop on Wireless Mesh Networks (WiMesh)*, September 2005.
- [129] M. R. Salavatipour. A Polynomial Time Algorithm for Strong Edge Coloring of Partial k-Trees. *Discrete Applied Mathematics*, 143(1-3):285–291, 2004.

- [130] Nordic Semi-Conductors. nRF905 Multiband Transceiver. <http://www.nordicsemi.com>.
- [131] Sentilla. Tmote Sky, Ultra low power IEEE 802.15.4 compliant wireless sensor module. <http://www.sentilla.com>.
- [132] N. Shacham and P. King. Architectures and Performance of Multichannel Multi-hop Packet Radio Networks. *IEEE Journal on Selected Areas in Communications*, 5(6):1013–1025, 1987.
- [133] W. Shang, P. Wan, and X. Hu. Approximation algorithm for minimal convergecast time problem in wireless sensor networks. *Wireless Networks*, 16(5):1345–1353, 2010.
- [134] M. Singh and L. C. Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 661–670, 2007.
- [135] J. So and N. H. Vaidya. Multi-channel MAC for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver. In *5th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*, pages 222–233, 2004.
- [136] D. Song, B. Krishnamachari, and J. Heidemann. Experimental study of the effects of transmission power control and blacklisting in wireless sensor networks. In *1st Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 289–298, October 2004.
- [137] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, and M. Nixon. WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control. In *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 377–386, April 2008.
- [138] W.-Z. Song, F. Yuan, R. LaHusen, and B. Shirazi. Time-optimum packet scheduling for many-to-one routing in wireless sensor networks. *International Journal of Parallel, Emergent and Distributed Systems*, 22(5):355–370, 2007.
- [139] N. A. Streitz, A. Kameas, and I. Mavrommati. The Disappearing Computer, Interaction Design, System Infrastructures and Applications for Smart Environments. In *The Disappearing Computer*, volume 4500 of *Lecture Notes in Computer Science*. Springer, 2007.
- [140] Z. Tang and J. J. Garcia-Luna-Aceves. Hop Reservation Multiple Access (HRMA) for Ad-Hoc Networks. In *18th IEEE International Conference on Computer Communications (INFOCOM)*, pages 194–201, March 1999.

- [141] Crossbow Technology. Wildfire detection. <http://www.xbow.com/Eko/EnvironmentalWildfireDetection.aspx>.
- [142] USC Table Tennis. Ping Pong Posse. <http://pingpongposse.com>.
- [143] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. A taxonomy of wireless micro-sensor network models. *SIGMOBILE Mobile Computing and Communications Review*, 6(2):28–36, 2002.
- [144] H.-W. Tsai and T.-S. Chen. Minimal Time and Conflict-Free Schedule for Convergecast in Wireless Sensor Networks. In *IEEE International Conference on Communications (ICC)*, pages 2808–2812, May 2008.
- [145] A. Tzamaloukas and J. J. Garcia-Luna-Aceves. Channel-Hopping Multiple Access. In *IEEE International Communications Conference (ICC)*, pages 415–419, June 2000.
- [146] S. Upadhyayula and S. K. S. Gupta. Spanning tree based algorithms for low latency and energy efficient data aggregation enhanced convergecast (DAC) in wireless sensor networks. *Ad Hoc Networks*, 5(5):626–648, 2007.
- [147] R. Vedantham, S. Kakumanu, S. Lakshmanan, and R. Sivakumar. Component based channel assignment in single radio, multi-channel ad hoc networks. In *12th Annual International Conference on Mobile Computing and Networking (MOBI-COM)*, pages 378–389, 2006.
- [148] T. Voigt. Self-organizing, Collision-free, Multi-channel Convergecast. In *European Conference on Wireless Sensor Networks (EWSN) (poster paper)*, page 2, February 2008.
- [149] A. Warburton. Approximation of Pareto optima in multiple-objective, shortest-path problems. *Operations Research*, 35(1):70–79, 1987.
- [150] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang. Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks. In *20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1388–1397, April 2001.
- [151] G. Werner-Allen, K. Lorincz, M. Welsh, O. Marcillo, J. Johnson, M. Ruiz, and J. Lees. Deploying a Wireless Sensor Network on an Active Volcano. *IEEE Internet Computing*, 10(2):18–25, 2006.
- [152] A. D. Wood, J. A. Stankovic, and G. Zhou. DEEJAM: Defeating Energy-Efficient Jamming in IEEE 802.15.4-based Wireless Networks. In *4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 60–69, June 2007.

- [153] W. Wu, H. Du, X. Jia, Y. Li, and S. C.-H. Huang. Minimum Connected Dominating Sets and Maximal Independent Sets in Unit Disk Graphs. *Theoretical Computer Science*, 352(1):1–7, 2006.
- [154] Y. Wu, J.A. Stankovic, T. He, and S. Lin. Realistic and Efficient Multi-Channel Communications in Wireless Sensor Networks. In *27th IEEE Intl. Conference on Computer Communications (INFOCOM)*, pages 1193–1201, April 2008.
- [155] W. Xiao and D. Starobinski. Exploiting multi-Channel diversity to speed up over-the-air programming of wireless sensor networks. In *3rd Intl. Conference on Embedded Networked Sensor Systems (SenSys)*, pages 292–293, November 2005.
- [156] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *2nd Intl. Conf. on Embedded Networked Sensor Systems (SenSys)*, pages 13–24, 2004.
- [157] W. Xu, W. Trappe, and Y. Zhang. Channel surfing: Defending wireless sensor networks from interference. In *6th International Conference on Information Processing in Sensor Networks (IPSN)*, pages 499–508, April 2007.
- [158] G. Yu and C. Chang. An efficient cluster-based multi-channel management protocol for wireless ad hoc networks. *Computer Comm.*, 30(8):1742–1753, 2007.
- [159] L. Yu, N. Wang, and X. Meng. Real-time forest fire detection with wireless sensor networks. In *Proceedings of Wireless Communications, Networking and Mobile Computing (WCNC)*, pages 1214–1217, March 2005.
- [160] J. Zander. Performance of optimum transmitter power control in cellular radio systems. *IEEE Transactions on Vehicular Technology*, 41(1):57–62, 1992.
- [161] H. Zhang, P. Soldati, and M. Johansson. Optimal link scheduling and channel assignment for convergecast in linear wireless HART networks. In *WiOPT*, pages 82–89, 2009.
- [162] J. Zhang, G. Zhou, C. Huang, S. H. Son, and J. A. Stankovic. TMMAC: An Energy Efficient Multi-Channel MAC Protocol for Ad Hoc Networks. In *IEEE International Conference on Communications (ICC)*, pages 3554–3561, June 2007.
- [163] Y. Zhang, S. Gandham, and Q. Huang. Distributed Minimal Time Convergecast Scheduling for Small or Sparse Data Sources. In *28th IEEE International Real-Time Systems Symposium (RTSS)*, pages 301–310, December 2007.
- [164] C. Zhou and B. Krishnamachari. Localized Topology Generation Mechanisms for Self-Configuring Sensor Networks. In *IEEE Global Communications Conference (GLOBECOM)*, pages 1269–1273, December 2003.