NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

INTERFERENCE AWARE ROUTING USING SPATIAL REUSE IN WIRELESS SENSOR NETWORKS

by

Michael A. Woods

December 2013

Thesis Advisor:	Preetha Thulasiraman
Second Reader:	Rachel Goshorn

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** December 2013	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE INTERFERENCE AWARE ROUTING USING SPATIAL REUSE IN WIRELESS SENSOR NETWORKS		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Michael A. Woods			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB protocol number ____N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE** A	

13. ABSTRACT (maximum 200 words)

A wireless sensor network (WSN) is comprised of sensor nodes designed to collect and transmit data efficiently. For this reason, WSNs are relied upon by the Department of Defense for deployment in remote and hostile areas. The performance of a WSN is degraded by the amount of interference experienced by nodes during simultaneous transmissions. Transmitting in the presence of interference can affect the lifetime of sensor nodes by requiring multiple re-transmissions of data. in this thesis, we propose a routing algorithm that uses spatial time-division multiple access (STDMA) to schedule simultaneous transmissions such that interference is mitigated and transmission time slots are reused appropriately. We integrate STDMA with a physical interference model that facilitates the calculation of interference metrics based on signal-to-interference ratio. Using the interference metrics as link costs, we implement Dijkstra's algorithm to determine the least interference path from a sensor node to the gateway. Via simulations using MATLAB and QualNet, we show that this approach to interference mitigation helps network performance by decreasing end-to-end delay. We develop this algorithm as a proof-of-concept to show that, despite the computational complexity associated with interference based scheduling, STDMA can have a real impact on network design and performance.

14. SUBJECT TERMS WSN, wireless, sensor, network, physical interference, interference, TDMA, STDMA, routing, Dijkstra, routing protocol, MAC, medium access layer, physical layer, network layer			**15. NUMBER OF PAGES** 109
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

INTERFERENCE AWARE ROUTING USING SPATIAL REUSE IN WIRELESS SENSOR NETWORKS

Michael A. Woods
Lieutenant, United States Navy
B.S.E.E., University of Oklahoma, 2004

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

December 2013

Author: Michael A. Woods

Approved by: Preetha Thulasiraman
Thesis Advisor

Rachel Goshorn
Second Reader

R. Clark Robertson
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

A wireless sensor network (WSN) is comprised of sensor nodes designed to collect and transmit data efficiently. For this reason, WSNs are relied upon by the Department of Defense for deployment in remote and hostile areas. The performance of a WSN is degraded by the amount of interference experienced by nodes during simultaneous transmissions. Transmitting in the presence of interference can affect the lifetime of sensor nodes by requiring multiple re-transmissions of data. In this thesis, we propose a routing algorithm that uses spatial time-division multiple access (STDMA) to schedule simultaneous transmissions such that interference is mitigated and transmission time slots are reused appropriately. We integrate STDMA with a physical interference model that facilitates the calculation of interference metrics based on signal-to-interference ratio. Using the interference metrics as link costs, we implement Dijkstra's algorithm to determine the least interference path from a sensor node to the gateway. Via simulations using MATLAB and QualNet, we show that this approach to interference mitigation helps network performance by decreasing end-to-end delay. We develop this algorithm as a proof-of-concept to show that, despite the computational complexity associated with interference based scheduling, STDMA can have a real impact on network design and performance.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

CAP	contention access period
CFP	contention free period
CSMA/CA	carrier sense multiple access with collision avoidance
DLL	data link layer
IEEE	Institute of Electrical and Electronics Engineers
ISM	industrial, scientific, and medical
LR-PAN	low-rate personal area network
MAC	media access layer
OSI	open systems interconnection
PHY	physical access layer
STDMA	spatial time-division multiple access
WSN	wireless sensor network
UWB	ultra wide band

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Efficient design and implementation of wireless sensor networks (WSNs) has become an increasingly active area of research in recent years due to the vast potential of sensor networks to enable applications that connect the physical world to the virtual world [1]. There is significant interest in the development and use of WSNs for military applications to increase battlefield information dominance. WSNs provide the ability to maintain an information gathering capability in remote and hostile environments.

Like all wireless networks, the performance of a WSN is degraded by the wireless interference experienced at individual nodes during simultaneous data transmissions. Additionally, WSNs have limitations due to the limited battery capacity of the nodes [1]. The lifetime of a sensor node can be significantly reduced if excessive wireless interference necessitates multiple re-transmissions of data. Thus, interference and sensor node lifetime go hand in hand. When designing a network protocol stack suitable for use in WSNs, interference at the physical layer must be taken into account to provide for reliable and efficient WSN operation.

In this thesis we concentrate on modeling interference by exploiting a concept known as spatial reuse. Essentially, spatial reuse allows nodes to simultaneously transmit during the same time slot as long as they have sufficient spatial separation. Spatial reuse is modeled at the data link layer (DLL) using spatial time-division multiple access (STDMA) [2]. STDMA governs the schedule of when nodes transmit (i.e., which time slot is assigned to a node for transmission). When using STDMA, multiple nodes are assigned to transmit during the same time slot, thus allowing appropriate time slot reuse to increase network performance. We integrate the STDMA protocol with a physical interference model in order to capture interference metrics using signal-to-interference ratio (SIR). Our focus is to use these captured interference metrics to determine the least interfering paths between sensor nodes and the gateway to allow for mitigated interference routing.

In order to solve the problem of routing in a WSN with interference generated from simultaneous transmissions (inter-node interference), it is important to first understand the models that exist to capture interference in wireless networks. The nature of the wireless communications medium used in WSNs results in the inclusion of inter-node interference in addition to the desired transmitted signal at all receivers in the network. Typically SIR is the key parameter in evaluating signal reception in wireless networks. There are two primary interference models used in networking literature to capture the effects of interference on wireless transmissions. These are the Protocol Interference Model (PrIM) and the Physical Interference Model (PhIM). Under the PrIM, two nodes can successfully communicate if and only if the receiver is within the transmission range of the sender and the receiver is out of the interference range of other simultaneous senders. Under the PhIM, a receiver can successfully receive data from the sender if and only if the SIR of the sender at the receiver is no less than a predefined threshold. The PrIM simplifies the communication model of wireless networks and is therefore more convenient for analysis. By contrast, the PhIM takes the received physical signal strength as a criterion to measure interference, which has been proven to be more accurate and reliable [2].

In this thesis, we use the PhIM to capture interference metrics in the WSN. Specifically, we use STDMA to schedule transmissions and then use PhIM to calculate the SIR witnessed at each receiving node.

We implement STDMA with the following in mind: Given our goal of maximizing spatial reuse and controlling interference, we do not allow one-hop neighbors to simultaneously transmit because of the excess interference that would be created from these transmissions. We allow for two-hop neighbors to transmit. These neighbors form a transmission group (i.e., all nodes within the same transmission group can transmit simultaneously). An example network with color-coded transmission groups is shown in Figure 1. For example, in Figure 1, all blue nodes may transmit simultaneously in the same time slot. This holds for all the colored node groups.

The next step is to compute the interference experienced during each reception due to simultaneous transmissions throughout the transmission group. In other words,

what interference does node 2 experience when listening to node 1 due to transmissions from nodes 3, 5, 11, 13, and 15? For this calculation we assume that all nodes in the transmission group have data to send, which provides the maximum possible interference. The calculations for the SIR at each node can be calculated as

$$SIR_i(j) = \frac{P_i(j)}{\sum_{k \in TG_j} P_i(k)} \quad . \tag{1}$$

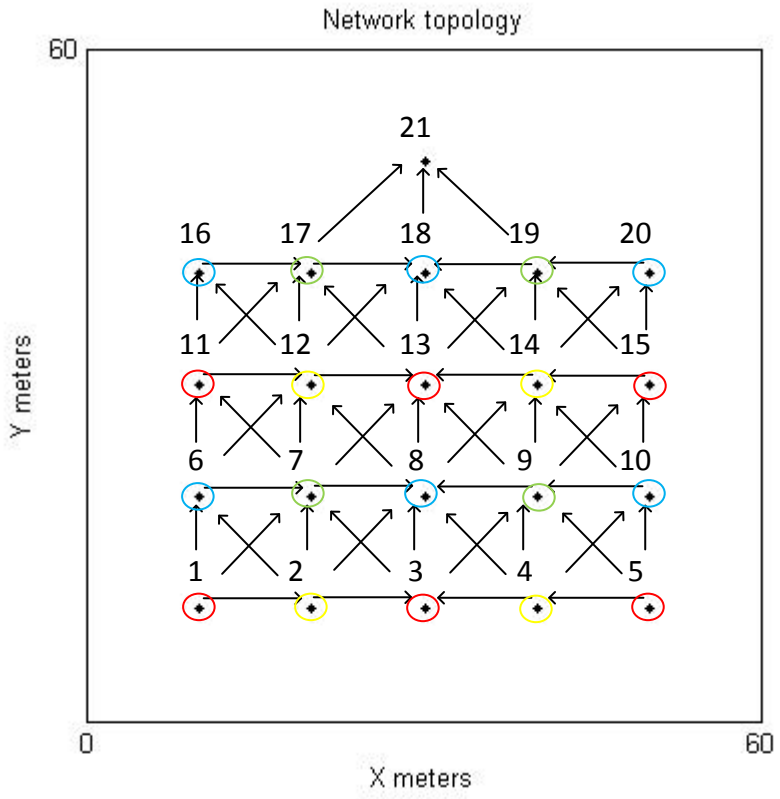

Figure 1. Example of a network that depicts color-coded transmission groups. All nodes within the same color-coded transmission group may transmit simultaneously within the STDMA schedule.

The SIR calculation given by Equation (1) states that the interference received at node i during reception from transmitting node j (i.e., node 2 from node 1) is the aggregate of the transmitted power from all other nodes in the transmission group of node j (TG_j),

which is received at node i. Equation (1) maintains the assumption that all nodes in the transmission group send data simultaneously and that the interference is at its maximum possible value. This calculation is repeated for every combination of receiving nodes and applicable transmission groups, which enables us to assign an interference metric (i.e., the calculated SIR value) to each edge in our interference graph.

To facilitate the SIR calculation, we take advantage of graph theoretic models. Any network can be represented as a graph $G(V,E)$ where V is the set of nodes/vertices in the network and E is the set of edges/links in the network. We transpose the original network graph G into what is known as a conflict graph G_I. In a conflict graph, denoted as $G_I(V_I,E_I)$, a vertex is introduced for each link in the original network $G(V,E)$. An edge in $G_I(V_I,E_I)$ connects two vertices in the conflict graph if these two links interfere in the original graph (as in the case of simultaneous transmissions). To characterize the interference relationship between two links in the conflict graph, we assign a SIR to the links that reflect the aggregate SIR for all simultaneously transmitting nodes. We treat the SIR as link weights such that a higher SIR equates to a lower cost for transmission. Specifically, we assign to each link in the conflict graph the value of the inverse SIR for the receiving node of the link. The graphical results of the link weight calculations and assignments for the network in Figure 1 are shown in Figure 2.

We use the link weights and connectivity graphs developed above as inputs to Dijkstra's least cost algorithm to calculate the shortest path from a sensor to the gateway with least interference.

Using MATLAB and QualNet, we found that the algorithm we developed improves network delay but hinders network throughput for various fading and traffic models when compared to the traditional time-division multiple access (TDMA) scheme. The results shown in Table 1 compare the network performance for several scenarios that use a TDMA schedule and several scenarios that use interference-aware STDMA scheduling. In addition to different transmission schedules, the scenarios include variable packet traffic types and fading channels for both TDMA and STDMA scheduling. The results of Table 1 specifically relate the performance of TDMA versus STDMA in terms

of throughput and delay. From the results, it can be seen that there is a significant improvement to network delay when using STDMA rather than TDMA. However, the throughput performance does not increase notably and in some configurations decreases or remains the same.

Our interpretation of the results is that time slot re-use allows slots to matriculate faster across the network but at the expense of interference causing the received SIR to drop below the required threshold. This loss of packets results in a decreased throughput. Despite the tradeoff between delay and throughput, we see the results obtained as a proof-of-concept that a deterministic PhIM can provide value in network design and performance despite its computational complexity.

An interference aware routing mechanism that considers the scheduling of simultaneous transmissions using STDMA can offer quantifiable gains to the overall network performance of WSNs. The technique presented in this thesis as a proof-of-concept shows that the PhIM integrated with a STDMA approach to obtain least interfering paths can provide for efficient utilization of network resources for a given network implementation.

Table 1. Results comparing delay and throughput when using STDMA versus TDMA under different fading and traffic models for a 10-node network.

MAC Protocol	Fading Model	Traffic Model	Throughput (kbps)	Delay (sec)	Throughput Improvement using STDMA (%)	Delay Improvement using STDMA (%)
TDMA	None	CBR	4283	0.1237	−0.4	24.3
STDMA	None	CBR	4266	0.0937		
TDMA	Rayleigh	CBR	4105	0.1194	3.9	21.5
STDMA	Rayleigh	CBR	4266	0.0937		
TDMA	None	VBR	5936	0.1493	3.4	31.0
STDMA	None	VBR	5733	0.1030		
TDMA	Rayleigh	VBR	5576	0.1488	−0.004	29.8
STDMA	Rayleigh	VBR	5554	0.1044		

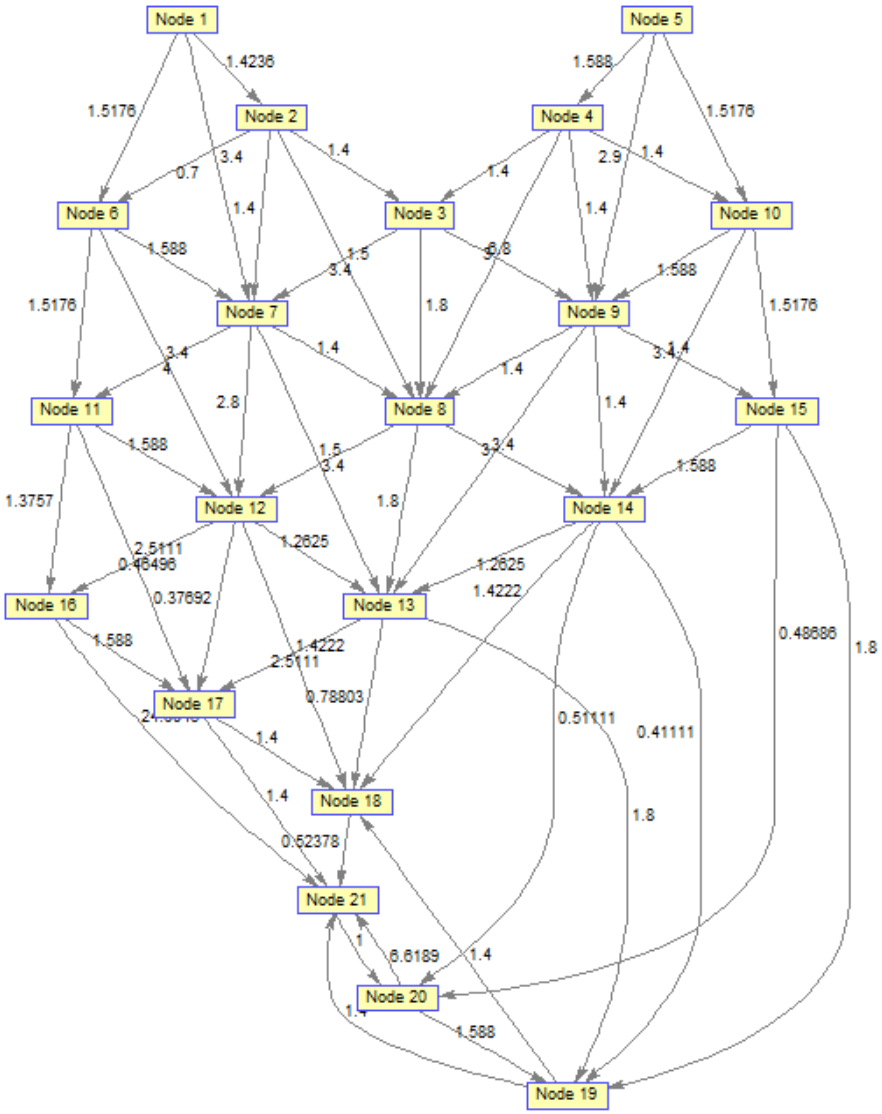

Figure 2. Conflict graph with associated SIR values for the network graph given in Figure 1.

LIST OF REFERENCES

[1] Mark A. Perillo and Wendi B. Heinzelman, *Wireless Sensor Network Protocols, Appears in Fundamental Algorithms and Protocols for Wireless and Mobile Networks*. Boca Raton, FL: CRC Hall, 2005.

[2] P. Varbrand and D. Yuan, "Resource Allocation of Spatial Time Division Multiple Access in Multi-hop Radio Networks," *Resource Management in Wireless Networking*, vol.16, pp. 198–222, 2005.

[3] Shouling Ji, Beyah, R., Yingshu Li, "Continuous Data Collection Capacity of Wireless Sensor Networks under Physical Interference Model," *Proceedings of IEEE 8th International Conference on Mobile Adhoc and Sensor Systems* (MASS), Valencia, Spain, 2011, pp. 222–231.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my parents, spouse, family, and friends for all of the support throughout my academic and military career. I would also like to express my sincere graditude for the faculty at the Naval Postgraduate School for creating a positive learning environment that has helped me vastly improve my engineering skills.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION AND MOTIVATION

In recent years there has been a rising interest in data-centric warfare in all branches of the military. Wireless sensor networks (WSNs) provide a flexible technology option to achieve the objective of gathering information about the physical environment and leveraging this information to achieve battlespace dominance in the air, on land and at sea. The miniaturization of computer processors and advancement of communications technologies allows for a widespread distribution of sensors, in an ad-hoc manner, for data extraction in real time.

WSNs are wireless ad-hoc networks that are characterized by low power, low processing capable measurement nodes that gather data from the environment. The sensor nodes also act as routing elements for the movement of this data from a source node to a destination. The destination node in a WSN is characterized as a gateway or sink node. The gateway is a point of aggregation where information from all the sensors is received and processed. The gateway eventually sends this gathered information to a command and control unit for processing and dissemination.

The small physical size and inherently limited capabilities of the WSN nodes combined with the desire to use the nodes without replacement serves as the motivation to develop simple and efficient communications mechanisms for a WSN. The most lifetime-limiting component for a WSN node is the power supply, and the most demanding power consuming component is the transceiver antenna in the transmitting mode. It is with these considerations in mind that it becomes readily apparent that an effort must be made to minimize transmissions from the sensor node while still forwarding the required collected information.

In a WSN, which can consist of hundreds or thousands of nodes, sequencing the transmission and movement of data is required to minimize delay of data delivery from the network and to prevent node-to-node interference as data flows. This unique routing demand, coupled with limited node battery life, requires WSN communications protocols

to be tailored to provide low power consumption and congestion free data transfer. With these protocols, WSNs across a potentially large physical area in an effort to provide high end-to-end data throughput.

This thesis focuses on modeling inter-node interference, and using the interference relationships as a basis for a least-cost routing algorithm to increase the data throughput of a WSN. For the remainder of this thesis, the term interference refers to inter-node interference that is generated by simultaneous transmissions.

A. WIRELESS SENSOR NETWORK NODE ARCHITECTURE

The desire for accurate and timely remote gathering of information has served as the motivation for the development and rapid maturation of WSN technologies. WSNs serve an important purpose in military specific applications such as force protection systems designed to detect and report detection of enemy movements.

The WSN nodes consist of four key components in addition to any number of optional application dependent components. The key components are a sensing unit, on-board processor, radio transceiver, and power supply. These WSN nodes are used for environmental sensing, data processing, communications, and stand-alone battery power [1]. The key components of a WSN node are shown in Figure 1. The sensor on a WSN node is designed to sense a specific physical quantity, transfer this physical quantity into an electrical signal, and then digitize the information for subsequent processing and forwarding. Once this digital information has been collected, the node's processing unit performs a variety of tasks including data averaging and packet construction in preparation for transmission. The radio transceiver transmits packets that originate from the node while also forwarding packets received from surrounding nodes. The power supply, a battery, is used to power all the aforementioned functions and is, therefore, a critical design feature of any WSN node.

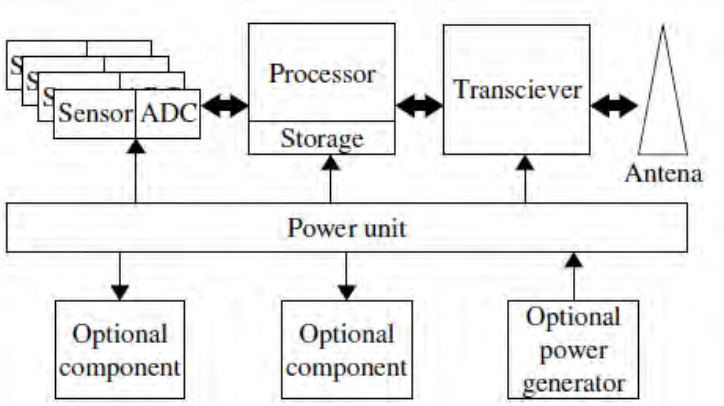

Figure 1. WSN node architecture from [1].

B. LAYERING ARCHITECTURE OF A NETWORK

When designing a communications network for widespread interoperability, the most common model used is the Open Systems Interconnection (OSI) reference model shown in Figure 2. The OSI model is used a reference because it reduces the complexity of network design and analysis and separates the functions logically into modular sections to create an understood framework for network implementation. The OSI model consists of seven distinct protocol layers that are often referred to as a protocol stack. Each layer in the stack communicates with the layers above and below it. The layers of a network operate to provide reliable and effective data communication over a variety of physical media channels based on the needs of the network designer and the network users. The lower layers of the model—physical layer (PHY), data link layer (DLL), network layer, and transport layer—are primarily concerned with the formatting, encoding, and transmission of data over the network [2]. The functions of these layers are implemented in both hardware and software in WSNs. The higher layers of the OSI model—session, presentation, and application—are primarily concerned with interacting with the user and are usually implemented in software [2].

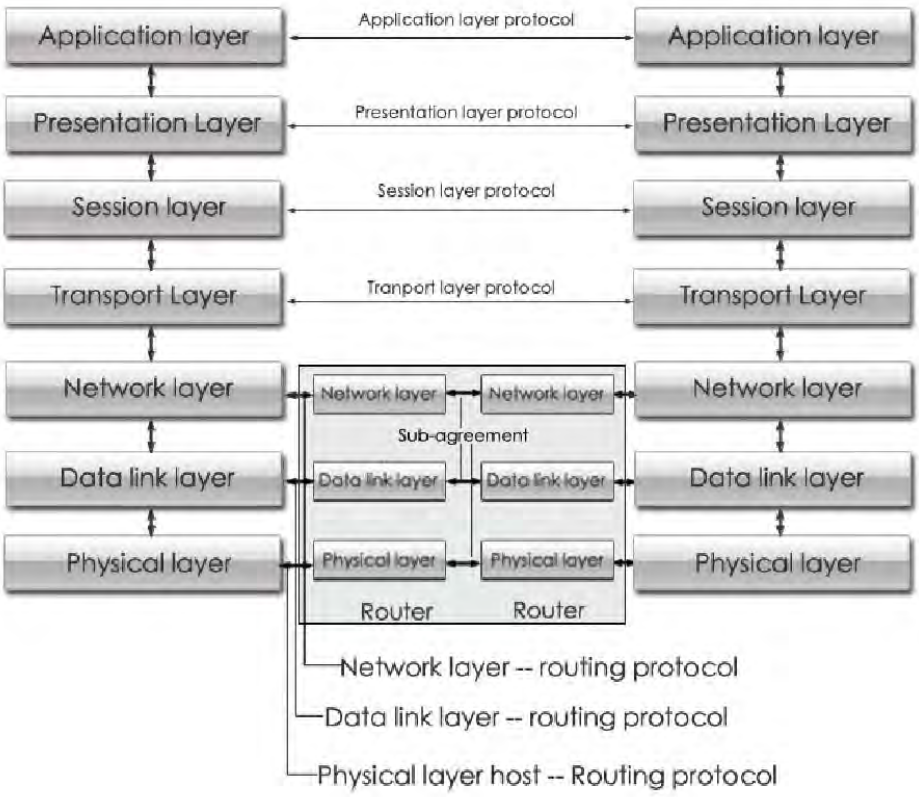

Figure 2. OSI network layering model from [3].

The typical WSN networking protocol stack consists of a number of layered protocols similar to the OSI network stack. Specifically, there is a physical layer (PHY Layer) for radio communication, data link layer (DLL) for packet transmission coordination, and a networking layer, which provides an efficient network-wide packet routing solution.

The PHY layer specifies all network communications parameters such as frequency bands and modulation schemes. In order for the PHY layer to operate correctly, each node must use the same pulse shaping and timing mechanisms. Choices for PHY layer parameters are based on balancing the needs of providing adequate performance while limiting the complexity and power required for operation.

The DLL is typically separated into the medium access control (MAC) and logical link control (LLC) sub-layers. The MAC sub-layer serves to allow the users coordinated

access to the transmission medium by the use of either distributed or centralized coordination functions. Distributed coordination involves the use of information local to each node. Nodes then make logical decisions in order to achieve collision free transmissions. Centralized coordination describes the use of a centralized entity to coordinate access to the transmission medium. In the case of a WSN, the entity is the gateway node and is used to create a transmission schedule that is sent to all other nodes. An example of a MAC scheduling function is the time-division multiple access (TDMA) scheme wherein each node is assigned a time slot on a schedule during which it is allowed to transmit information to neighboring nodes. TDMA allows collision free transmission with little complexity but does not maximize network performance because only one node can transmit at a time, and each node remains idle for long periods of time. In order to improve the performance of TDMA scheduling, spatial time-division multiple access (STDMA) allows multiple nodes to utilize the same time slot if they are located at a sufficient distance away from each other in order to limit the interference caused by simultaneous transmissions.

The network layer is used to establish the optimal end-to-end routing path across the network. Additionally, the networking layer provides component (i.e., node and gateway) addressing, packet fragmentation, and packet reassembly.

There exists a wide variety of Institute of Electrical and Electronics Engineers (IEEE) standards that can be used as the design basis for a reliable wireless communications system. In Chapter II, we describe the IEEE 802.15.4 standard, which specifies the PHY and DLL layers for low-rate personal area networks (LR-PAN). This standard provides the lower layer implementation definitions for many WSN protocol designs and is, therefore, an appropriate example to provide a framework for the functionality of the PHY and DLL layers in a WSN. There are a number of network layer protocols that are appropriate for WSN operation. We choose the Dijkstra least-cost routing algorithm, which is discussed in Chapter V.

In addition to the three lower layers, there are protocols layered above these that provide application specific functionality. In this thesis we are not concerned with the

functionality of these top layers. Our focus is on the design and implementation of routing protocols considering the PHY, DLL, and network layers.

C. INTERFERENCE MODELING

As wireless communication systems have evolved, with a more aggressive utilization of spectral resources and more efficient transmission systems, a better understanding of the effects of interference on the performance of such systems is of paramount importance. This has motivated the development of more accurate interference models [4]. The resource constraints of WSN nodes serve as a motivating factor to design the most efficient communications protocols at the PHY layer in an effort to increase performance at higher layers in the presence of interference.

The purpose of an interference model is to determine the effects of interference at different protocol layers. An interference model can be viewed as a combination of the following sub-models:

- **Propagation Model**: Uses communications theory to describe the effects of attenuation, shadowing, noise, and fading on the received signal.
- **Transmitter Distribution Model**: Describes how the transmitters (i.e., nodes in a WSN) are distributed in the area of interest.
- **Network Operation Model**: Describes the MAC access technique in the network, which specifies the network coordination function (i.e., the use of a transmission schedule).
- **Traffic Model**: Describes the assumed statistical distribution of arrival of packets into the network.

The components of the sub-models appear in all interference models and can be modeled as deterministic or random processes according to the scenario considered. Interference models are largely divided into (1) statistical interference models, which attempt to statistically describe the wireless interference, and (2) interference models that seek to describe the effects of the interference on the operation of the network of interest

[4]. The two most prominent subdivisions of the latter group are the protocol interference model (PrIM) and physical interference model (PhIM). In addition to the PrIM and PhIM models, there are graph-based models that have been developed to describe the effects of interference in the context of transmission scheduling and node topology control. The PrIM states that a transmission between two nodes is successfully received if the receiving node is within the transmission radius of the transmitting node and outside the interference radii of all simultaneously transmitting nodes. The PhIM states that transmission between two nodes is successfully received if the signal-to-interference ratio (SIR) is greater than a minimum value [5]. In general, the PrIM allows for faster interference computation and the PhIM provides more accurate interference modeling. All classes of interference models relate to the radio capture phenomenon, which describes the ability of a node to successfully receive a radio transmission in the presence of inter-node interference [4].

D. MOTIVATIONS AND CONTRIBUTIONS OF THE THESIS

In the wireless networking literature, there are many studies that validate the advantages of using a PhIM approach in contrast to a simpler PrIM for interference modeling. Despite the fact that the PhIM provides a more accurate representation of interference in a WSN, this accuracy comes at the cost of a significant increase in complexity for the MAC and PHY layer protocols. In this thesis we propose a simplistic model of a PhIM in order to: (1) perform a weighted conflict graph analysis based on the SIR on all links in a WSN, (2) use the SIR obtained in the graph analysis as the cost metric for a network layer routing algorithm, and (3) demonstrate the advantages of using a least-cost routing algorithm in a STDMA MAC protocol in comparison to a TDMA MAC protocol.

The contribution of the research proposed in this thesis is a proof-of-concept for a cross-layer routing and medium access system using physical interference as the routing criteria. The WSN models and assumptions aid in the demonstration of the proposed concept. Future work is proposed that will provide a more robust protocol level implementation for a more general WSN setup.

E. ORGANIZATION OF THE THESIS

The remainder of this thesis is organized as follows. An in-depth explanation of the OSI reference model and a detailed discussion of the IEEE 802.15.4 Standard for LR-PAN is provided in Chapter II. The concepts of interference modeling are detailed in Chapter III by describing, comparing, and contrasting the most widely used interference models in the literature. The MAC protocol implementation and the advantages of using STDMA in lieu of TDMA for WSNs are described in Chapter IV. The concept of least-cost routing and the operation of the Dijkstra algorithm as an implementation of a least-cost routing mechanism is explained in Chapter V. The experimental setup and an evaluation of the simulation results is discussed in Chapter VI. The conclusion and recommended future work are presented in Chapter VII. All the MATLAB code used in the simulations are contained in the appendix.

II. NETWORK LAYER PROTOCOLS

In this chapter we present an explanation of both the general construction of layered network protocols and the specific example of the implementation of the IEEE 802.15.4 standard for WSN implementation. As briefly discussed in Chapter I, Section B, modular network layering protocols were introduced by the OSI reference model in the late 1970s. This approach to network design remains the standard approach to developing a network protocol stack. The framework provided by the OSI model is used in nearly all WSN designs and is used as a basis for discussion of the network layers in this section. In addition to the general information provided by the OSI model about the functionality provided by a given protocol layer, the IEEE 802.15.4 standard is used as an example to provide a WSN specific implementation of the two lowest layers of the stack. Beyond the PHY and DLL layers, the discussion of higher layer functionality in this thesis is focused primarily on the routing protocol of the network layer.

A. PHYSICAL LAYER

The lowest layer in the OSI model, the PHY layer, is responsible for the transmission and reception of data. It dictates the radio frequency bands to be employed and the type of frequency spreading and modulation techniques to be used for communication [6]. The original IEEE 802.15.4 was released in 2003 and has received five updates, including the most recent update in 2011. With each of these releases, the PHY layer has been updated to provide more flexibility and increased performance with an accompanying increase in complexity.

The original U.S. IEEE 802.15.4 standard provides frequencies in the 2.4 GHz industrial scientific medical (ISM) radio band, for unlicensed device operation, using direct sequence spread spectrum techniques. This wide-band implementation supports 16 channels with a guard band of 5 MHz between them. The bands range from 2400-2483.5 MHz and support a data rate of 250 kb/s. The scheme uses offset quadrature phase-shift keying modulation and half-sine pulse shaping. Updates to the standard have increased

the bandwidth from wide-band to ultra wide-band (UWB), which increased the data rate to up to 1 Mb/s. Additionally, multiple modulation schemes were added to allow for variable data rates, which resulted in a trade-off between data transfer rate and power consumption [6].

B. DATA LINK LAYER

The second lowest layer in the OSI model, the DLL, is primarily responsible for the multiplexing of data streams, medium access control, error control, and data frame detection. As was stated in Chapter I, the DLL is divided into two sub-layers—a higher LLC sub-layer and lower MAC sub-layer. The subdivision of the DLL into two sub-layers is necessary to accommodate the logic required to manage access to a shared communications medium. This inherent complexity suggests the choice of a MAC protocol is crucial to the performance of a WSN [1]. There are many options for MAC protocols, and the best choice is often application dependent. Here we discuss the key features and benefits of the IEEE 802.15.4 MAC protocol as a reference framework for MAC operation.

The MAC sub-layer, which appears just above the PHY layer of the OSI model, is responsible for managing transmissions, access to the channel and association/disassociation to the network. The MAC protocol of the IEEE 802.15.4 standard supports two modes of transmission [6].

- **Beacon-enabled:** In this mode, periodic beacons are transmitted by a designated coordinating node (i.e., the gateway) in order to maintain synchronization and exchange network information between the nodes. These beacons constitute the first slot of a 16 slot superframe. The exchange of data takes place during the superframe. A superframe is a collection of timeslots that repeat and provide a format for scheduling transmissions. The structure of the superframe is defined by the coordinating node, and the remaining nodes of the WSN are informed about the superframe structure through beacons. The superframe structure is shown in Figure 3. It can be seen that the superframe is bounded by the

beacon frames. The entire superframe may be occupied by the contention access period (CAP). During the CAP, if a device wishes to communicate, it has to contend with other devices using a slotted carrier-sense multiple access with collision avoidance (CSMA/CA) mechanism; however, if some devices (or applications) require low latency communication, then a contention free period (CFP) is introduced, which consists of guaranteed time slots (GTS) [6]. The inactive period allows for nodes to shut off for a portion of the communications cycle to conserve battery power and elongate the lifetime of the WSN.

- **Non Beacon-enabled**: When the WSN operates in this mode, there are no beacons broadcast by the coordinator and there are no superframes. Access to the channel in the network takes place using an unslotted CSMA/CA mechanism [6].

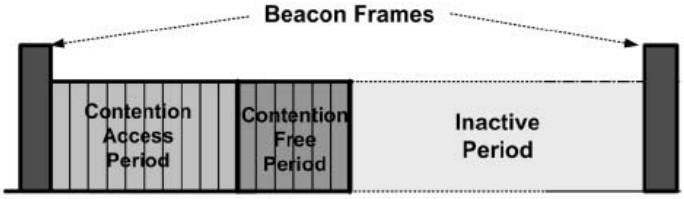

Figure 3. IEEE 802.15.4 superframe structure from [6].

Based on the 802.15.4 example, it can be seen that the purpose of the MAC sub-layer is to provide coordinated access to the communications channel for all nodes in a WSN in an effort to reduce delay, prevent packet collisions, and minimize required re-transmissions. The effectiveness of a specific MAC implementation plays a key role in the power consumption of the nodes and is, therefore, a critical component in WSN design.

C. NETWORK LAYER

The 802.15.4 standard does not address the stack implementation above the DLL and instead leaves the functional design of this layer to the network engineer. In this section, we discuss the purpose of the network layer and the considerations that need to be studied when designing the best routing algorithm. The network layer defines how interconnected networks interact with one another. The network layer is the first layer in the OSI model that is concerned with sending data to a remote network in contrast to the DLL, which functions to send data across a local network only (i.e., communications at the DLL are point-to-point rather than end-to-end). Specific functions normally performed by the network layer include:

- **Logical Addressing**: Every node that communicates within a WSN must be assigned an address. A familiar example of addressing is the Internet protocol (IP) address for the network layer protocol. Network layer logical addresses are independent of particular hardware and must be unique across an entire interconnected network [2].

- **Routing**: Moving data across a series of interconnected networks is the defining function of the network layer. It is the job of the devices and software routines that function at the network layer to handle incoming packets from various sources, determine their final destination, and then determine the best method to forward them.

- **Datagram Encapsulation**: The network layer encapsulates messages from higher layers by placing them into datagrams, called packets, with a network layer header.

- **Fragmentation and Reassembly**: The network layer must send messages down to the DLL for transmission. If the DLL has a design limit on the packet length that differs from the packet length used at the network layer, then the outgoing packets have to be broken into fragments. This process, known as fragmentation, must be un-done when packets are received at the destination node. This is known as reassembly.

- **Error Handling and Diagnostics**: Special protocols are used at the network layer to allow devices that are logically connected, or that are routing traffic, to exchange information about the status of hosts on the network or the devices themselves [2].

Network layer protocols may offer either connection-oriented or connectionless services for delivering packets across the network. Connectionless protocols are the most common at the network layer. In many protocol suites, the network layer protocol is connectionless, and connection-oriented services are provided by the upper layers [2].

WSN specific routing requirements and energy limitations require the development of routing protocols that are designed for the efficient and effective operation of WSNs. WSN routing protocols can be divided into data-centric, location based and hierarchical types of routing protocols. In data-centric routing, the gateway sends queries to certain regions of the network and waits for data from the sensors located in the selected regions. Sensor Protocols for Information via Negotiation was the first data-centric protocol. It considers data negotiation between nodes in order to eliminate redundant data and save energy [7]. Other examples of notable data-centric WSN routing protocols are gradient-based routing [8], constrained anisotropic diffusion routing [9], and active query forwarding in sensor networks [10]. Location-based routing protocols focus on delay optimization by greedily forwarding packets towards sensors near the destination. Location-based protocols include minimum energy communication network [11], geographic adaptive fidelity [12], and geographic and energy aware routing [13]. Hierarchical WSN routing protocols exploit rigid topology constraints to provide energy-efficient and scalable routing [14]. Popular hierarchical routing protocols include the low-energy adaptive clustering hierarchy [15] and power-efficient gathering in sensor information systems [16].

D. UPPER LAYERS

The functions of the layers above the network layer in the OSI model include connection establishment, data acknowledgement and retransmission, data multiplexing

and de-multiplexing, and software application issues [2]. The functionality of the upper layers is not discussed in this thesis. Our focus is on the PHY, DLL, and network layers of the WSN.

III. INTERFERENCE CONSIDERATIONS IN WIRELESS COMMUNICATIONS SYSTEMS

One of the key aspects that must be considered when modeling interference in wireless communications is how interference disturbs the reception of a given desired signal [4]. Early random-access models assumed that if any two transmitters had temporally overlapping transmissions then the transmissions would be collisions and all data would have to be re-sent. This assumption resulted in a view that a signal would always be lost in the presence of interference, which necessitated the use of medium access protocols that provided either time or frequency exclusivity. Medium access of this variety provides collision-free operation but does not maximize use of resources by allowing transmissions from nodes with sufficient spatial separation. The introduction of the radio capture concept allows for the successful reception of a desired signal in an interference environment and, thus, provides for the possibility of multiple simultaneous transmissions. These transmissions potentially interfere but are controlled in order to allow for successful reception by placing design limits on the level of interference. There are many radio capture models available in the literature, which can be broadly grouped together as examples of protocol or physical interference models. In this chapter we develop the conceptual basis for the PrIM and the PhIM in order to demonstrate the superiority of the PhIM and associated trade-offs of using the more accurate model.

A. INTERFERENCE MODEL DEFINITIONS

Consider a WSN $G(V,L)$ consisting of n wireless nodes $V=\{\mathrm{v}_1,\mathrm{v}_2,...,\mathrm{v}_n\}$, which are distributed in a two-dimensional planar grid. Each node is wirelessly connected to its neighboring nodes over one of m communications links $L=\{l_1,l_2,...,l_m\}$. In the development of the PrIM and PhIM in [17], arbitrary and random network settings are provided as analysis options. In the arbitrary network setting, the location of nodes and traffic patterns can be chosen by the network designer. Since any traffic pattern and node placement can be used, the bounds for this scenario are applicable to any network; therefore, the arbitrary network setting may be viewed as the best case bounds on

network performance [18]. In the random setting node location, traffic patterns and transmission power for each node are all independently selected using a uniform random distribution. For all subsequent analysis in this thesis, we assume arbitrary node placement, node transmission power, and traffic pattern settings for simplicity of analysis.

B. PROTOCOL INTERFERENCE MODELS

Under the protocol model, the impact of interference from neighboring nodes is binary and is solely determined by whether or not the node falls within the interference range of non-intended transmitters [19]. The conceptual basis for the protocol interference model was initially introduced in [17] in an effort to analyze the capacity of multi-hop wireless networks. The protocol interference model has been the focus of rigorous study and application since being introduced and is the basis for numerous MAC protocol implementations. The basis of the protocol interference model is the vulnerability capture model, which states that the receiving node that is closest to the transmitting node in a given wireless network has the greatest received power.

The PrIM is an implicit interference model that describes the effects of interference based on the communications channel conditions [4]. Given the WSN $G(V,L)$, the PrIM asserts that a transmission from v_i to v_j, located at Euclidean distance $d(i,j)$, over some link $l_{i,j} \in L$, is successful if for every other node v_k that is simultaneously transmitting is located $d(k,j)$ away. This is stated as

$$d(k,j) \geq (1+\Delta)d(i,j), \Delta > 0 \tag{3.1}$$

where Δ is a guard parameter to ensure that all conflicting transmitting nodes are sufficiently far away from the receiver to allow successful decoding of the intended signal [18]. In order to relate the Δ parameter to a physically significant quantity, the PrIM introduces the concept of the maximum transmission range and the maximum interference range where $R_T^{\max}$ and $R_I^{\max}$ represent the maximum transmission and maximum interference ranges of each node, respectively. The transmission range

represents the maximum distance up to which a packet can be received, while the interference range represents the maximum distance up to which simultaneous transmissions interfere. In the literature, the interference range is usually chosen to be twice as large as the transmission range, which has been shown to not necessarily be a practical assumption [5]. In reality, the actual values of the transmission and interference ranges depend on the transmission power used by the nodes and are a function of the SIR threshold [5]. Optimized and more realistic values for $R_I^{\max}$ can be obtained using a reality check mechanism as shown in [19], which helps to obtain a more realistic agreement between the protocol model behavior and real-world physical interference behavior.

Within the construct of the PrIM, there are two different types of interference that have been studied in the literature, namely primary interference and secondary interference. Primary interference occurs when a node transmits and receives packets at the same time. Secondary interference occurs when a node receives two or more separate transmissions [20].

The inadequacies of the PrIM become apparent when analyzing the secondary interference problem. The assumption that concurrent transmissions do not interfere as long as the nodes that are exchanging information are outside of one another's interference radius does not accurately account for aggregate interference. According to the PrIM, the aggregate interference stems from the fact that multiple simultaneous transmissions with sufficient spatial separation can combine to constructively interfere and prevent successful reception.

C. PHYSICAL INTERFERENCE MODELS

The PhIM is based on the power capture model that assumes a threshold based channel model. In other words, the transmission data rate is a function of the signal-to-interference ratio (SIR). If the SIR is sufficiently large, the data rate is guaranteed and constant. Conversely, if the SIR is below a pre-defined threshold, then the transmission data rate is zero [4]. The SIR is calculated from

$$SIR = \frac{P_j(i)}{\sum_{k \in V', k \neq i} P_j(k)} \geq \beta \tag{3.2}$$

where β is the SIR threshold above which data transmissions are successful from v_i to v_j in the presence of all concurrent secondary interference events, V' is the subset of nodes in the WSN that are assigned to transmit simultaneously and $P_j(i)$ is the received power at node *j* due to node *i*. When calculating the SIR, $P_j(i)$ is calculated from

$$P_j(i) = d(i, j)^{-\alpha}, \alpha \geq 2 \tag{3.3}$$

where $d(i, j)$ is the Euclidian distance between nodes and α is the path loss exponent, which is a function of the propagation loss model used to account for free-space power loss during radio transmission.

As can be seen from the above equations, the PhIM does not suffer from the same shortcomings as the PrIM. Namely, the error introduced by neglecting aggregate interference effects and the unrealistic binary channel condition utilized in the PrIM are not concerns when using the PhIM. These advantages in accuracy are countered by significantly increased complexity in simulation analysis. Additionally, the complexity of PhIM based MAC protocol implementations is greatly increased as compared to available PrIM solutions.

D. GRAPH BASED INTERFERENCE MODELS

Graph theory is an important analytical tool in the field of wireless network design and modeling. Graph theory provides the ability to describe a network with a set of vertices and edges that connect the vertices. A graph G is generally defined as a function of its vertices and edges (i.e., $G(V, E)$). In a WSN, the sensor nodes are the vertices, and the wireless links between nodes are the edges. An example connectivity graph is shown in Figure 4 with a set of vertices $V = \{A, B, C, D, E\}$ connected by a set of edges $E = \{AB, AE, BC, BD, BE, CD, DE\}$. The graph G is said to be connected if there

is a path from any vertex to any other vertex. Graphs are either undirected, in which edges are represented by a line, or directed (also called a digraph), in which an edge is represented by an arrow drawn from the tail of a vertex to the head of a vertex [4].

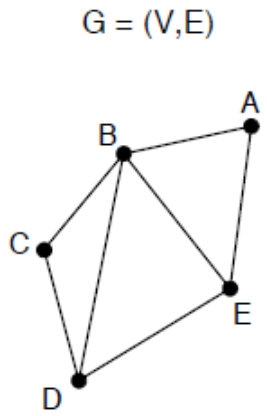

Figure 4. Example of an undirected graph from [8].

A basic aspect of an ad-hoc wireless network that can be modeled by means of a graph is the radio connection between any two nodes. A typical rule to determine if a given node v_i can communicate with another node v_j is that the SIR of the received signal at v_j must be equal to or larger than a given threshold β [8]. If this connectivity criterion is satisfied, then an edge is assigned from v_i to v_j. Thus, a connectivity graph representing a wireless network, such as a WSN, can be created with knowledge of node location and node transmit power. The network connectivity described by a connectivity graph can be misleading, and it should be noted that this connectivity does not account for interference due to concurrent transmissions. The interference encountered in networks that utilize a packet forwarding scheme, allowing simultaneous transmissions, is usually accounted for by constructing a structure known as a conflict graph [5]. In a conflict graph, denoted as $G_I(V_I, E_I)$, a vertex is introduced for each link in the original network $G(V, E)$. An edge in $G_I(V_I, E_I)$ connects two vertices in the conflict graph if these two links interfere in the corresponding connectivity graph (as in the case of simultaneous transmissions).

The appropriate parameter of a wireless link to characterize the interference relationship between two links is the SIR tolerated at the receiver. At this point, we distinguish two approaches for assigning values to the links in $G_I(V_I, E_I)$: (1) in its aggregate form, as in the PhIM, or (2) by considering interference from each interferer individually, as in the PrIM [4], [5]. As stated in the previous discussion, the PrIM has a $R_T^{\max}$ and $R_I^{\max}$ associated with each node that can be determined by using a free space path-loss model and the node transmit power. In the case of a PrIM based conflict graph, an edge is added to the graph in the event of secondary interference by a simultaneously transmitting node. Note that the interference between two links, according to this rule, is not a reciprocal relationship, and interference graphs are in general directed [4].

Recalling that the PrIM is a binary communications model, for every instance that a link exists in the conflict graph, we see that there is a need for exclusivity in channel access for the conflicting nodes. Thus, the number of nodes permitted to transmit at the same time (i.e., the amount of link reuse) is a direct output of the conflict graph. The amount of link reuse allowed is, therefore, a function of the loss propagation model, transmission power, and interference range model used. In contrast to the PrIM, the PhIM is an aggregate interference model; thus, the conflict graph link weights consist of aggregate SIR values for all simultaneously transmitting nodes. The PhIM takes into account all active links regardless of the distance between the links, which results in a distinctly different conflict graph as compared to the PrIM for a given network topology. The PhIM conflict graph presents a more accurate representation of the interference in the physical environment as compared to the PrIM. Unfortunately, in general it is difficult to obtain accurate and continuously updated SIR information; additionally, there is an overhead incurred to share that SIR information for dynamically assigning conflict graph link weights. This extra computation and the associated power and bandwidth consumption results in the choice of a PrIM approach to interference modeling for many models. In this thesis we propose a simplified approach to assigning SIR values to a PhIM conflict graph in an effort to bridge the gap between the simplicity of the PrIM and the accuracy of the PhIM.

IV. MEDIUM ACCESS CONTROL

In this chapter we shift our focus from the accuracy and computational ease with which interference can be modeled in a WSN to how we use our interference analysis results to best share the communications resources amongst the WSN nodes. The MAC protocol is primarily responsible for orderly and efficient resource management in a wireless networking system. The medium access mechanism can be implemented in using either a distributed coordination function or a centralized coordination function. Generally speaking, a distributed coordination function consists of a contention based medium access method where nodes compete with their neighbors for transmission time and channel resources. Central coordination functionality, on the other hand, involves the execution of a carefully synchronized transmission schedule, which is determined by a central entity. An example of a coordinated MAC protocol is the time-division multiple access (TDMA) protocol, which allocates network resources, node by node, using time slots. There are inherent complexities associated with using TDMA including timing synchronization, message exchange for slot assignment, and slot schedule optimization to account for various network dynamics. In this chapter we present a detailed explanation of the TDMA protocol and a discussion of the advantages of using TDMA for a WSN as opposed to a contention based MAC protocol.

An upgrade to TDMA is the STDMA protocol, which takes advantage of the spatial separation of nodes in a WSN and allows more than one node to transmit in the same time slot (slot reuse). STDMA is superior to TDMA and can decrease delay and increase network capacity. In order for an STDMA protocol to operate properly, there must be a thorough understanding of the interference environment created by allowing the time slot reuse. In this chapter we explain the operation of the TDMA and STDMA protocols, compare and contrast the two protocols, demonstrate the superiority of STDMA to TDMA, and discuss the effects of accurate interference modeling on the performance of an STDMA protocol implementation.

A. TIME-DIVISION MULTIPLE ACCESS

In a TDMA access scheme, a conflict-free access schedule is achieved by assigning a time slot to each node and only one node can transmit per time slot. A frame is formed with the concatenation of all time slots and the required non-data overhead timeslots. In a traditional TDMA protocol, each node is assigned one time slot per frame, and the slots are assigned by a controlling entity, such as the gateway in a WSN. The time slot schedule can be predefined to aid in simplicity of network design or it can be periodically updated to reflect changes in network traffic patterns to increase performance. In this thesis, we use the simplified TDMA protocol with a pre-defined slot assignment schedule as our reference TDMA model.

In a TDMA scheme the time axis is divided into M time slots (assuming a WSN consisting of M nodes) that are pre-assigned to the different WSN nodes. Each node is permitted to transmit using the entire channel bandwidth during its assigned time slot. The slot assignments follow a predetermined pattern that repeats itself periodically, and each such period is called a frame. In the most basic TDMA scheme, every user has exactly one slot in every frame as indicated by the numerical indices in Figure 5.

In our analysis of TDMA, we are interested primarily in the throughput and delay characteristics, which depend on frame length, slot assignment, and slot assignment ordering. We consider the throughput to be the average aggregate data that can be transported through the channel per unit time. We quantify the throughput as the fraction of time during which node data is being successfully transferred. The delay is the time from message generation until message delivery to the next hop. We infer that a lower per-hop delay indicates a lower overall system wide delay when sending information from an arbitrary node to the gateway.

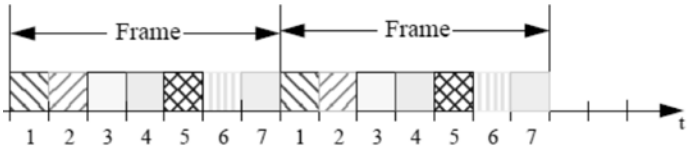

Figure 5. Example of a TDMA frame structure from [21].

To analyze the performance of a TDMA scheme, consider a system composed of M nodes with each node transmitting equally long packets of P bits each. If the total rate of transmission is R bits/sec, then the packet transmission time is

$$T = \frac{P}{R} \text{sec}. \tag{4.1}$$

This is taken to be the slot size. The duration of the entire cycle is, therefore, equal to T_c where

$$T_c = MT \text{ sec}. \tag{4.2}$$

Assuming that the packet generation processes of new packets by each WSN node are independent, it follows that the queuing behaviors of each nodes' queue are independent. This assumption that the queues are independent is reasonable because each node transmits a packet every T_c seconds, independently of any event in any of the other queues of other users. Consequently, in the following analysis, we concentrate on the characteristics of one node. Without loss of generality, we assume that the node transmits a packet if there is a packet in the node's queue during its assigned slot of every frame.

Consider a typical packet generated by a node. The delay suffered by this packet has three components: (1) the time between its generation and the end of the current frame, (2) the queuing time to allow all the packets already queued to be transmitted and, (3) the packet transmission time itself. Of these components, the first and the third are readily known since all frames are of equal length, and the average time between the packet generation time and the end of the current frame is $0.5T_c$. Here we assume that the average packet is generated halfway through an arbitrary frame period (i.e., 0.5 of T_c). By restricting our packet generation to Poisson arrivals and using the deterministic servicing of TDMA, our queuing time for each node can be described by an M/D/1 queue

$$W_q = \frac{\rho}{2(1-\rho)}\overline{x} \tag{4.3}$$

where $\overline{x} = T_c$ is the deterministic service time and $\rho_i = \lambda_i T_c$ is the fraction of the total network load arriving at node i [22]. The packet generation at the node is λ_i packets per second. Note that the queue notation M/D/1 denotes a single server queue with Poisson arrivals and a deterministic service time. Substituting T_c for $\overline{x}$ and recalling that $T_c = MT$, we obtain the expected queuing time

$$W_q = \frac{\rho}{2(1-\rho)} MT \text{ sec}. \tag{4.4}$$

Combining Eqs. (4.2) and (4.3), for an average packet generated at time $0.5T_c$ within a frame and with a transmission time of T sec, we obtain the total packet delay [22]

$$D = \frac{1}{2} T_c + W_q + T = \frac{MT}{2} + \frac{\rho}{2(1-\rho)} MT + T = T\left[1 + \frac{M}{2(1-\rho)}\right] \text{sec}. \tag{4.5}$$

From [22] the relationship between throughput S and utilization ρ for a M/D/1 queue is shown to be $S = \rho$. Substituting S for ρ in Eq. 4.5, we get

$$\frac{D}{T} = 1 + \frac{M}{2(1-S)} \ . \tag{4.6}$$

Equation 4.6 is the TDMA delay throughput characteristic where M is the number of nodes in the network, S is the network throughput, and D is the delay in the network. A family of graphs of the expected delay versus throughput for various network sizes (i.e., different values of M) is shown in Figure 6.

Though TDMA provides for simple implementation as well as closed form equations to be used in design and analysis, it does not provide for optimal utilization of resources on a per node basis. This drawback is particularly harmful in the context of an operating WSN where battery power is limited and care must be taken to maximize resource utilization. In the following section, we discuss how resource utilization can be improved by allowing simultaneous transmissions for multiple nodes while mitigating inter-node interference [23].

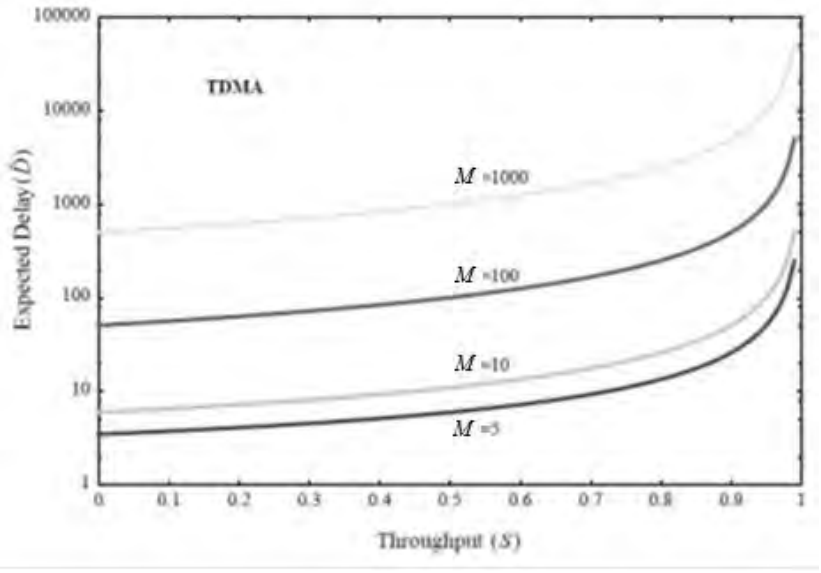

Figure 6. TDMA throughput versus expected delay curve for various network sizes from [22].

B. TIME-DIVISION MULTIPLE ACCESS WITH SPATIAL LINK REUSE

Access schemes based on the controlled simultaneous transmissions for nodes with sufficient spatial separations are known as STDMA protocols. STDMA operates by allowing multiple nodes to transmit at once as long as the nodes are sufficiently separated in space. In this case a given node may have the opportunity to transmit more than once per frame, thereby decreasing the average delay at each node. In order for this delay reduction to be useful, it is important that there is a thorough understanding of the interference environment created by using STDMA. An adequate model must be used to represent network interference in order to predict network performance under STDMA.

Similar to TDMA, STDMA is a synchronized protocol that uses a centralized coordination approach. STDMA requires an algorithm for generating the transmission schedule [23]. Transmission scheduling using STDMA is a complex problem that requires optimization in order to maximize resource utilization. Ideal characteristics of an STDMA scheduling algorithm are as follows: (1) the algorithm must be able to adapt to changes in network topology and (2) the algorithm must be simple enough to allow for fast, on-line computation. In addition to computing transmissions schedules that allow

for operation in the presence of interference, the STDMA schedule may be further optimized to dynamically re-order time slots in reaction to changing traffic patterns [24]. In practice there is no optimal STDMA algorithm due to the computational complexity of the STDMA implementation; therefore, the common approach is to develop fast heuristic algorithms to solve the STDMA problem [23].

In the remainder of this section, our goal is to show the derivation of the maximum throughput for an STDMA protocol assuming knowledge of the interference environment and that there is no time constraint for scheduling computation. It is important to note that when scheduling a transmission in an STDMA protocol, the time slots can be assigned to either nodes or links. In a node oriented assignment, a node that is scheduled to transmit may transmit to any other node, which allows for broadcast and multicast transmissions. For link-based transmission schedules, each node is only permitted to transmit over a specified point-to-point link to another node. Clearly, the broadcast and multicast capability afforded by node-based scheduling results in a higher number of available active links per node as compared to a link-based schedule. This characteristic of node-based STDMA serves as motivation for us to use a node-based STDMA schedule in our presentation of the comparison of STDMA to TDMA throughput.

For the purposes of the following discussion, our network model consists of a connectivity graph $G(V,E)$ and utilizes the PrIM for describing connectivity and interference. We assume that there are directional links between nodes, there are no primary interference conflicts allowed, and each link is error free as long as the $SIR \geq \beta$. We assume all nodes use the same transmit power, and we use frame and timing notation consistent with our earlier discussions on TDMA and SIR.

Recall that λ_i is the average traffic load arriving at node i and let h_i denote the number of slots in which node i is scheduled using a node-oriented schedule [23]. The throughput is the maximum λ_i for which the delay remains finite, which requires the following inequality to hold:

$$\lambda_i \le \frac{h_i}{T_c} \ . \tag{4.7}$$

The throughput is the largest λ_i that satisfies Eq. 4.7. This is represented as

$$\lambda_N^T = \min_{i \in N} \frac{\frac{h_i}{T_c}}{\rho_i} \ packets \Big/ \sec \tag{4.8}$$

where $\rho_i = \frac{\lambda_i}{\lambda}$ [23]. Having provided this formal definition of throughput, our goal is now to maximize the throughput for a given node assignment schedule. We proceed by introducing the concept of a transmission group, which is a group of nodes that can share a time slot. Let L_N denote the sets of transmission groups and s_{il} serve as a binary indication parameter defined as [23]

$$s_{il} = \begin{cases} 1, & \text{node } i \in l \\ 0, & otherwise \end{cases} \quad , \quad \forall \ l \in L_N \ . \tag{4.9}$$

Let x_l be the number of time slots assigned to transmission group l. The maximization of the throughput z is

$$z \le \frac{\sum_{l \in L_N} s_{il} x_l}{\rho_i T_c}, \forall l \in L_N \tag{4.10}$$

subject to the following constraints [23]:

$$\sum_{l \in L_N} x_l = T_c \tag{4.11}$$

$$x_l \ge 0, x_l \in \square, \forall l \in L_N \tag{4.12}$$

where $\square$ is the set of integers.

There exists a sizeable field of study dedicated to providing optimal and fast solutions to this optimization problem. The solution of this optimization requires the use

of linear programming numerical methods. The results of the optimization are detailed in [23] and show that the throughput for STDMA can be as much as 35 times the throughput in a traditional TDMA system.

V. INTERFERENCE AWARE LEAST-COST ROUTING

In this chapter we present our proposed method of improving upon the performance of traditional TDMA by implementing a physical interference based network routing algorithm. Our method consists of a SIR threshold based PHY layer reception model, STDMA DLL, and shortest path routing at the network layer. The discussions on interference modeling, graph theory and STDMA in the preceding chapters provide the background in explaining our proposed WSN communications stack.

There are various algorithms for finding the shortest path if the edges in a network are restricted to non-negative values. The most popular shortest path algorithm in use is Dijkstra's algorithm [25]. The operation of Dijkstra's algorithm in the context of using SIR values as the link-cost metric associated with transmission groups in an STDMA MAC protocol is explained in this chapter. We propose to use the PhIM to develop a network conflict graph in order to quantify the interference encountered at a given network node during data exchange over a wireless link. Once these interference values are obtained and link weights assigned, we use Dijkstra's algorithm to find the shortest path across a WSN using the interference weights as the routing metric. In Chapter VI, we simulate this routing method in conjunction with a STDMA MAC protocol and demonstrate that using interference-aware least cost routing is an effective means of reducing network delay when compared to interference aware routing using the traditional TDMA MAC protocol.

A. DIJKSTRA'S ALGORITHM

The problem of finding shortest paths plays a central role in the design and analysis of networks. Most routing problems can be solved as shortest path problems once an appropriate cost is assigned to each link. For example, cost metrics can reflect available bandwidth, delay or interference [25]. In a packet switched communications network such as a WSN, the function of routing data across a network using the shortest path is accomplished at the network layer.

Dijkstra's algorithm finds the least-cost distance from any node to any other node in a directed graph with non-negative weights. Dijkstra's algorithm has been shown to work in $\Theta\left(n^2\right)$ time for a network with n nodes with reduced complexity in the instance of sparse connectivity (i.e., non-dense networks). The inputs to the algorithm are the graph $G\left(V,E\right)$, weights $c:E\left(G\right)\rightarrow\square_+$, and a starting node $s\in V\left(G\right)$ where $E\left(G\right)$ and $V\left(G\right)$ denote the node and link weights of $G\left(V,E\right)$, respectively. The output consists of the shortest path from s to all $v\in V\left(G\right)$, and the associated path lengths are denoted as the sets $l\left(v\right)$ and $p\left(v\right)$, respectively.

The steps of the algorithm are as follows [26]:

1) Set $l\left(s\right)\square 0$, set $l\left(v\right)\square\infty$ for all $v\in V(G)\setminus\left\{s\right\}$, set $R:=\varnothing$

2) Find a vertex $\left\{v\in V\left(G\right)\setminus R:l\left(v\right)=\min\limits_{w\in V(G)\setminus R}\left[l\left(w\right)\right]\right\}$

3) Set $R:=R\cup\left\{v\right\}$

4) **For** all $\left\{w\in V\left(G\right)\setminus R:\left(v,w\right)\in E\left(G\right)\right\}$ **do**:

If $l\left(w\right)>l\left(v\right)+c\left(\left(v,w\right)\right)$ **then**

set $l\left(w\right):=l\left(v\right)+c\left(\left(v,w\right)\right)$ and $p\left(w\right):=v$.

5) **If** $R\neq V\left(G\right)$ **then go to** 2)

Step one in the algorithm sets node s as the starting node and assigns a cost of zero to node s while assigning an infinite initial cost to all other nodes. Additionally, the set of shortest paths is initialized as empty. In step two, the shortest path to all nodes from the current node is found and this path is added to the set R in step three. In step four all paths are compared to the newest visited node and the current path is verified as the shortest path. If this current path is not the shortest path, then the shortest path is

calculated based on the current cost value. This shortest path overwrites the current path. Verification that all nodes have been visited and termination of the algorithm once all nodes have been visited takes place in step 5.

B. INTERFERENCE AWARE LEAST-COST ROUTING ALGORITHM

Dijkstra's algorithm has been proven to find the shortest path among all nodes in a network and is, therefore, a logical choice for finding the shortest path through a WSN. In order for us to integrate Dijkstra's algorithm with our overall task of finding least interfering paths, we must first construct the connectivity and cost graphs for our STDMA WSN scenario. A depiction of the physical layout of a 21-node WSN is shown in Figure 7. Each node is placed in a Cartesian coordinate system in 10-meter increments. We created the network shown in Figure 7 using MATLAB [27]. The gateway for this WSN is designated as node 21, and all other nodes are considered sensing nodes that send information to the gateway. The set of nodes corresponding to this specific WSN model is $V(G) = \{1, 2, 3, ..., 21\}$.

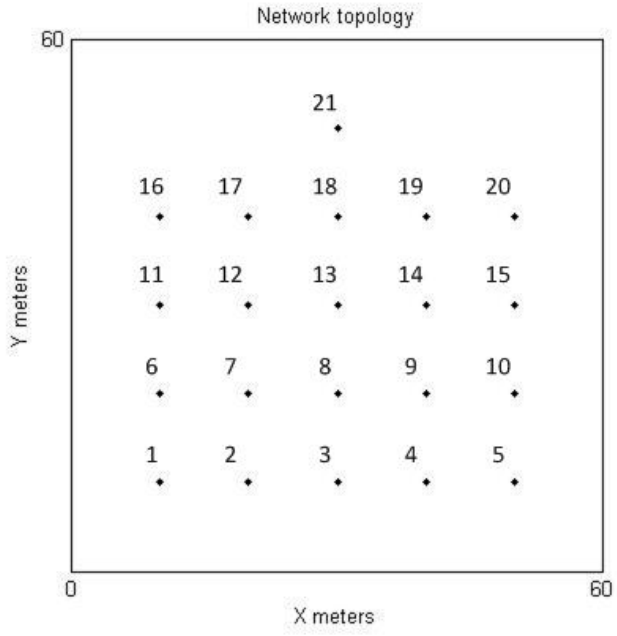

Figure 7. 21-Node WSN topology generated using MATLAB, where node 21 is the designated gateway.

Given this topology, we construct the connectivity graph starting with the allowed connections shown in Figure 8. Allowed connections are defined as connections that forward information toward the gateway.

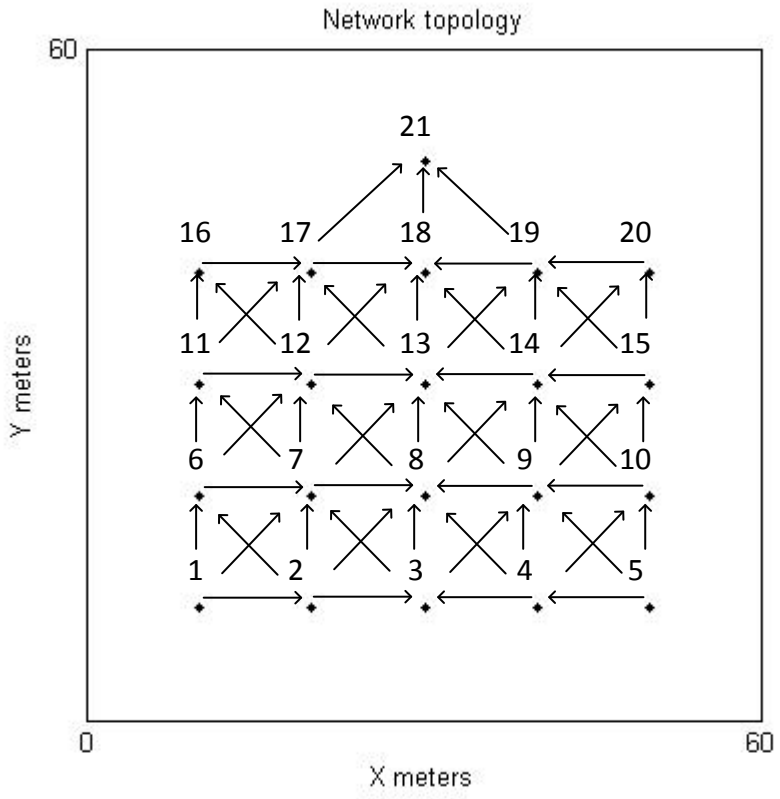

Figure 8. Connectivity graph for the 21-node WSN.

For our analysis, the term "links" refers to the directed edges that belong to the set of allowed connections. The set of allowed connections is denoted as $L(G)$. The links for this connection scenario are $L(G) = \{l_{1-2}, l_{1-6}, l_{1-7}, ..., l_{17-21}, l_{18-21}, l_{19-21}\}$. Now that we have obtained $V(G)$ and $L(G)$, we use these two sets to compose graph $G(V, L)$.

The next step is to take our knowledge of the topology and compute the power received at each node assuming 10 mW transmission from a node that corresponds to the maximum transmit power of a Mica2 mote, which is a common commercial sensor node [28]. We compute this received power from

$$P_1(j) = P_1^T \left(d(1,j)\right)^{-\alpha} \tag{5.1}$$

where $P_1(j)$ represents the power received at node j due to a transmission by node 1, $P_1^T = 10$ mW, $d(1,j)$ is the distance from node 1 to node *j*, and $\alpha = 2$ is the path loss factor. This calculation must be repeated for all nodes where $j \neq 1$ and repeated again for nodes 1 through 20. Since node 21 is the gateway, there are no transmissions from this node; thus, there is no requirement to compute the received power at the other network nodes due to transmissions from node 21. These received power values are used to compute the aggregate interference in our PhIM conflict graph assignments.

Given our goal of maximizing spatial reuse and controlling interference, we do not allow one-hop neighbors to simultaneously transmit because of the creation of excess interference. We allow for two-hop neighbors to transmit, which results in the four color-coded transmission groups shown in Figure 9. The colors denote nodes that are permitted to simultaneously transmit in our STDMA schedule. For example, all blue nodes may transmit simultaneously in the same time slot. This holds for all the colored node groups.

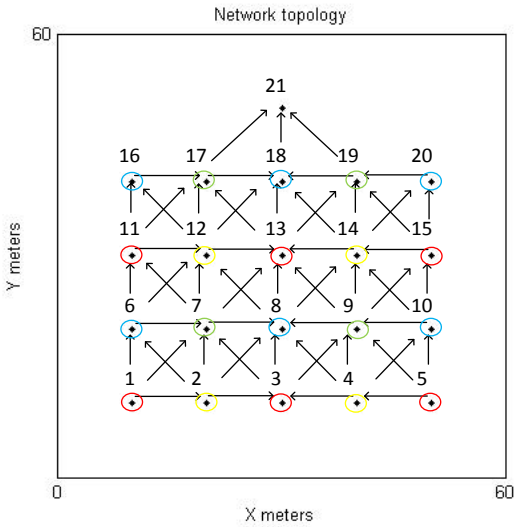

Figure 9. Transmission groups in the 21-node WSN to indicate simultaneous transmissions in the STDMA transmission schedule.

The next step is to compute the interference experienced during each reception due to simultaneous transmissions throughout the transmission group. For example, what interference does node 2 experience when listening to node 1 with the interference due to transmissions from nodes 3, 5, 11, 13, and 15? Note that nodes 1, 3, 5, 11, 13, and 15 all belong to the same transmission group. For this calculation we assume that all nodes in the transmission group have data to send, which provides the maximum possible interference. The calculations for the SIR at each node are calculated from

$$SIR_i(j) = \frac{P_i(j)}{\sum_{k \in TG_j} P_i(k)} \tag{5.2}$$

where TG_j denotes the transmission group of node j. This was also given in Eq. (3.2). This states that the interference received at node i during reception from transmitting node j (i.e., node 2 from node 1) is the aggregate of the transmitted power from all other nodes in the transmission group of node j. Equation (5.2) maintains the assumption that all nodes in the transmission group send data simultaneously and that the interference is at its maximum possible value (upper bound on experienced interference). This

calculation is repeated for every combination of receiving nodes and applicable transmission groups. This enables us to assign an interference metric (i.e., the calculated SIR value) to each edge in our interference graph.

To create the conflict graph, we treat the SIR values from the previous section as link weights such that higher SIR values equate to a lower cost for transmission. Specifically, we assign to each link in $L(G)$ the value of the inverse SIR for the receiving node of the link. For example, the link weight for l_{1-2} is assigned related to the SIR value corresponding to

$$l_{1-2} = \left[SIR_2(1)\right]^{-1} = \left[\frac{P_2(1)}{\sum_{k \in TG_1} P_2(k)}\right]^{-1} . \tag{5.3}$$

This process is repeated for each link in Figure 9 using MATLAB. The graphical result of the link weight calculations and assignments are shown in Figure 10.

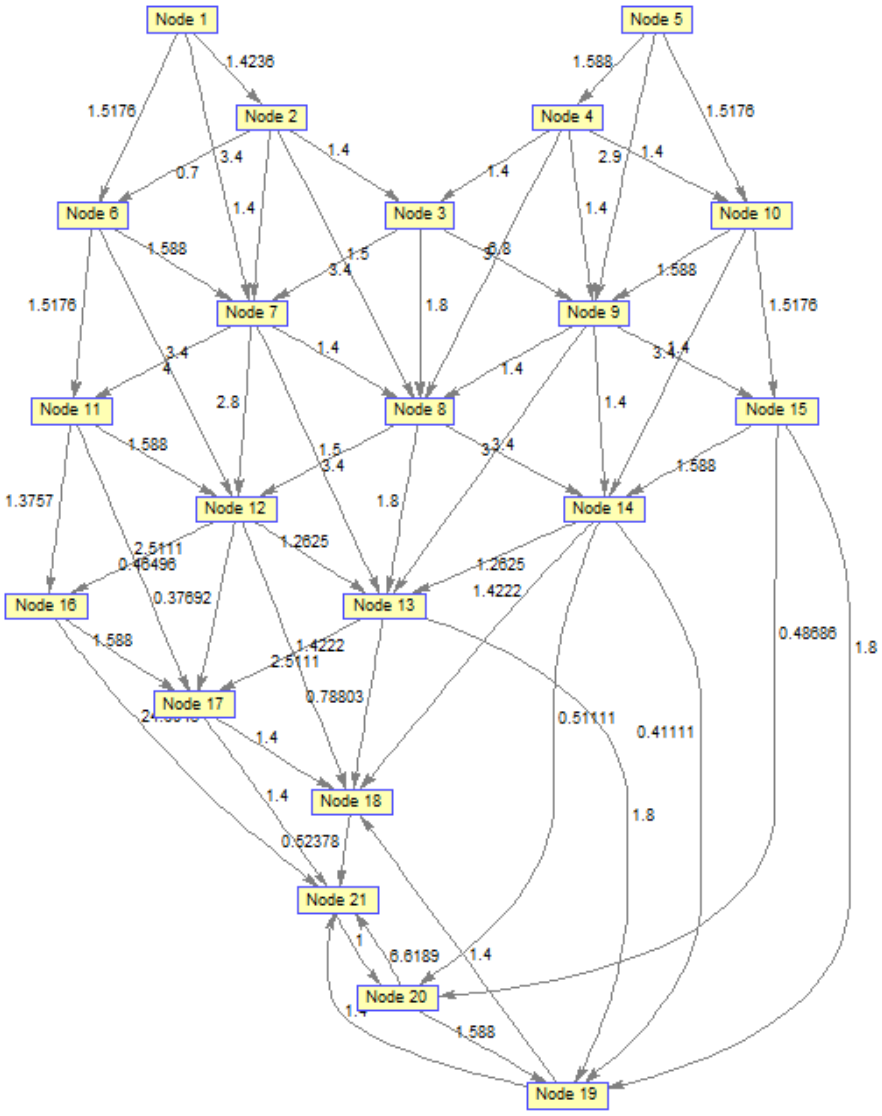

Figure 10. Weighted conflict interference graph for the network shown in Figure 8.

Next, we use the link weights and connectivity graphs developed above as inputs to Dijkstra's algorithm to calculate the shortest path based on interference. Once the shortest path has been determined, we construct the corresponding routing table and run simulations to compare the performance of TDMA against our implementation of STDMA.

VI. SIMULATIONS, EXPERIMENTS, AND PERFORMANCE EVALUATION

As discussed in the preceding chapter, our interference aware routing method requires evaluation at the PHY, DLL, and network layers. As such, we use two different computing tools to assist in the task of simulating our system. The first step is to calculate SIR values and develop a weighted conflict graph based on interference. This step is accomplished using MATLAB, a common engineering and mathematics programming language. The second step is to setup a WSN and use least-cost routing with both TDMA and STDMA so that comparisons can be made on the performance of our WSN routing technique. For the second set of tasks, we use QualNet [24], which is a comprehensive suite of tools for modeling wired and wireless networks including WSNs.

A. LEAST COST PATH GENERATION IN MATLAB

An overview of the steps accomplished in MATLAB, detailed in Chapter V, can be seen in Figure 11. The specific MATLAB scripts used to implement each block can be found in Appendix A through Appendix F.

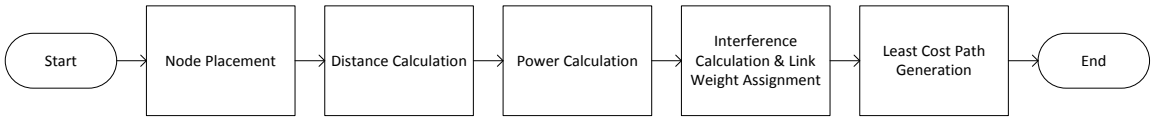

Figure 11. MATLAB flowchart of least-cost path generation.

We simulate two network configurations, one with 10 nodes and another with 21 nodes. The source and destination node were arbitrarily chosen to be node 1 and node 10 in the ten-node network, respectively. Similarly, the source and destination node were chosen to be node 1 and node 21 in the 21-node network, respectively. We assume in these cases that nodes 1 and 21 are the gateways for their respective networks. This allows data to traverse as much of the network as possible and allows a more thorough and realistic evaluation of the effects of interference on network performance.

In order to obtain the least-cost routing paths based on interference, we followed the MATLAB flowchart as shown in Figure 11. First we built a ten-node and 21-node network and calculated the distance between all nodes using Eq. (3.1). Next, we calculated the received power at each node based on transmissions from every other node and used this information to calculate the SIR values for each node per Eqs. (5.1) and (5.2), respectively. Finally, we used the SIR information to calculate the link metric cost using Eq. (5.3). We used all the link metric costs and network topology as inputs to the shortest path routing function of MATLAB. The results of the ten-node and 21-node shortest path determinations are shown in Figure 12 and Figure 13.

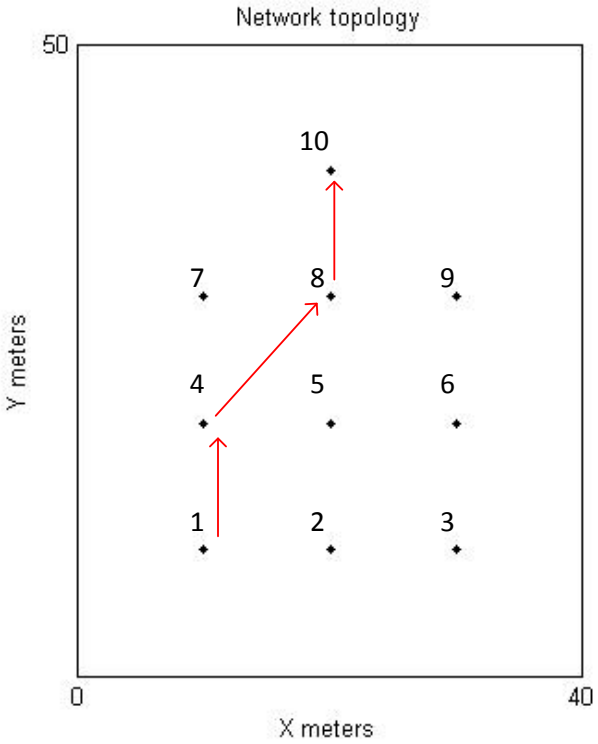

Figure 12. Ten-node interference-based shortest paths.

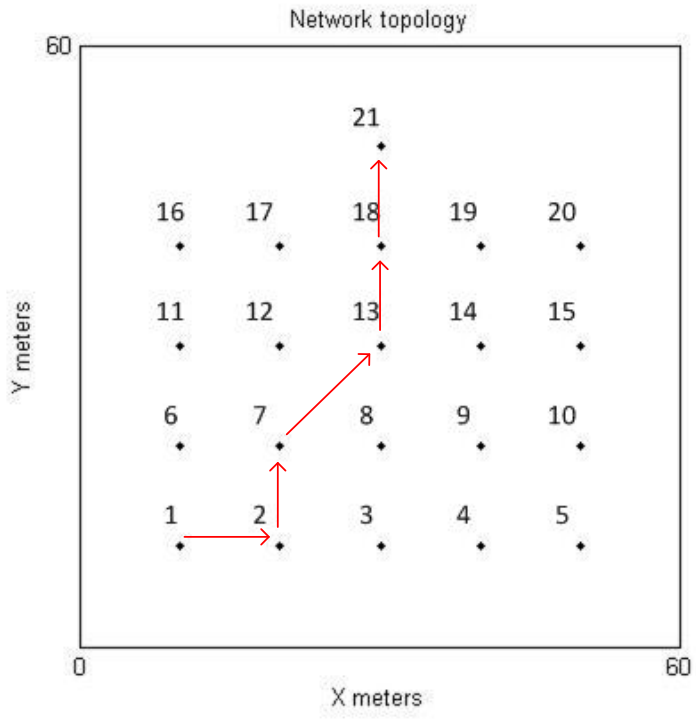

Figure 13. 21-node interference based shortest paths.

Note that in the ten-node case the least cost route is also the least hop route, unlike the 21-node case where the least cost route has more hops than the least hop route.

B. ROUTING SIMULATION IN QUALNET

QualNet [29] allows for scenario based simulation of various network types of different sizes and topology configurations. We used QualNet to establish the routes generated by our MATLAB results and added in a variety of variables to observe the effect of fading, traffic distribution type, and MAC layer protocol selection on WSN performance.

QualNet is composed of the following tools:

- QualNet Architect—A graphical experiment design and visualization tool. Architect has two modes: design mode for designing experiments, and visualize mode for running and visualizing experiments.
- QualNet Analyzer—A graphical statistics analyzing tool.

- QualNet Packet Tracer—A graphical tool to display and analyze packet traces.
- QualNet File Editor—A text editing tool.
- QualNet Command Line Interface—Command line access to the simulator.

In order to ensure that our results correspond to the scenario that was built in MATLAB, we built the equivalent scenario in QualNet, which is depicted in Figure 14 for the 21-node scenario. In Figure 14, the cloud icon and blue dashed lines represent a wireless subnet connection for all nodes in the WSN. In addition to node placement, all wireless properties of the network were assigned to match the MATLAB parameters (i.e., transmit power and path-loss model).

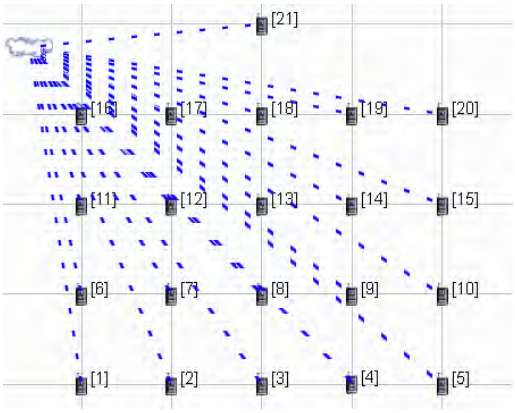

Figure 14. 21-node scenario built in QualNet.

1. Simulation Parameters

The simulation parameters common to all baseline experiments are enumerated in Tables 1 through 3, for the PHY, MAC, and Network layers, respectively.

Table 1. PHY layer simulation parameters.

Packet Reception Model	SIR Threshold
SIR Threshold	10
Transmission Power	10 dBm
Data Rate	2 Mbps
Antenna Gain	0 dB
Antenna Noise Temperature	290 K

The PHY parameters that are pertinent to the operation of our network simulation are presented in Table 1. These specifications require a received SIR greater than 10 in order for a packet to be considered successfully received (note that the SIR threshold β is unitless). We assign 10 mW (10 dBm) transmission power for all nodes and a data rate of R = 2 Mbps. Our antenna parameters are a 0 dB gain and 290 K antenna temperature, which is the standard antenna temperature for terrestrial antenna operation.

Table 2. MAC layer simulation parameters.

MAC Protocol	TDMA or STDMA
Slot Duration	10 ms
Guard Time	0 ms
Inter-frame Time	1 ms
Slots per frame	9 or 20

Our MAC parameters specify a 10 ms slot per node with no guard time between node time slots and a 1 ms guard between successive frames. The nine or 20 time slots specify one slot per node per frame in the 10- and 21-node scenarios, respectively.

Table 3. Network layer simulation parameters.

Routing Protocol	Static Routes
Networking Protocol	Internet Protocol version four

Our network routing protocol consists of using the shortest paths created using the Dijkstra algorithm in MATLAB. The static routes specified correspond to the routes in Figure 12 and Figure 13. All nodes utilize standard internet protocol addressing.

In addition to these general parameters, there are additional parameters that must be specified depending on the scenario. In some scenarios we include Rayliegh fading, in which case the fading parameters are shown in Table 4.

Table 4. Fading channel parameters used in the simulations.

Fading Model	Rayleigh
Signal Propagation Speed	3×10^8 m/s
Propagation Limit	−111 dBm

The Rayleigh fading model is a statistical model to represent the fast variation of signal amplitude at the receiver. In wireless propagation, Rayleigh fading occurs when there is no line-of-sight between the transmitter and receiver [29].

2. Experimental Results

The results of the experiments for various combinations of the MAC protocol, fading channel, and traffic distribution are shown in Table 5 for the 10-node scenario. The results of interest are the delay and end-to-end throughput from source to destination for constant bit rate (CBR) and variable bit rate (VBR) traffic applications. The VBR traffic model uses an exponential distribution. The last two columns of Table 5 indicate the percent improvement from using STDMA versus TDMA for the throughput and delay results shown.

Table 5. 10-node network. Percent improvement in delay and throughput for the 10-node network when using STDMA versus STDMA for varying scenarios.

MAC Protocol	Fading Model	Traffic Model	Throughput (kbps)	Delay (sec)	Throughput Improvement using STDMA (%)	Delay Improvement using STDMA (%)
TDMA	None	CBR	4283	0.1237	−0.4	24.3
STDMA	None	CBR	4266	0.0937		
TDMA	Rayleigh	CBR	4105	0.1194	3.9	21.5
STDMA	Rayleigh	CBR	4266	0.0937		
TDMA	None	VBR	5936	0.1493	3.4	31.0
STDMA	None	VBR	5733	0.1030		
TDMA	Rayleigh	VBR	5576	0.1488	−0.004	29.8
STDMA	Rayleigh	VBR	5554	0.1044		

The same simulation combinations were repeated for the 21-node network and the results are shown in Table 6.

Table 6. Results comparing delay and throughput when using STDMA versus TDMA under different fading and traffic models for a 21-node network.

MAC Protocol	Fading Model	Traffic Model	Throughput (kbps)	Delay (sec)	Throughput Improvement using STDMA (%)	Delay Improvement using STDMA (%)
TDMA	None	CBR	4388	0.2057	−6.7	25.9
STDMA	None	CBR	4095	0.1524		
TDMA	Rayleigh	CBR	4388	0.2057	−10.7	25.9
STDMA	Rayleigh	CBR	3917	0.1524		
TDMA	None	VBR	6088	0.3919	−20.3	53.1
STDMA	None	VBR	4850	0.1839		
TDMA	Rayleigh	VBR	6088	0.3919	−20.3	53.1
STDMA	Rayleigh	VBR	4850	0.1836		

In general, the performance trends showed no improvement in throughput and, in some cases, a significant decrease in throughput for the STDMA scenario as compared with the TDMA scenario. This is particularly true for the 21-node case (Table 6). However, in all cases there were significant decreases in end-to-end delay, which

indicates there is value in the approach taken. Our interpretation of the results is that slot re-use allows slots to matriculate faster across the network but at the expense of interference causing the received SIR to drop below the required threshold. This loss of packets results in a decreased throughput.

In an attempt to improve the throughput performance, we re-attempted the 21-node CBR no fading scenario with varying values of SIR threshold. We simulated two sets of cases: (1) small changes around our initial threshold value of 10 and (2) threshold changes over a larger scale. The results when the SIR is varied on a small scale are shown in Table 7, whereas the results when the SIR is varied on a large scale are shown in Table 8. The values in the Tables 7 and 8 indicate a decreasing trend in throughput as the SIR threshold becomes more discriminatory. In addition, we did not observe any increase in throughput as compared to our throughput for the STDMA case with no fading and CBR traffic when the SIR threshold was lowered below ten. These results confirm that the STDMA approach is not beneficial in terms of throughput but is useful in reducing transmission delay.

Table 7. Small variations in SIR threshold for throughput analysis of the 21-node network.

SIR Threshold	Throughput (kbps)
1	4095
2	4095
3	4095
4	4095
5	4095
6	3917
7	4095
8	3917
9	3739
10	3917
11	3739
12	3739
13	3917
14	3739
15	3561

Table 8. Large variations in SIR Threshold for throughput analysis of the 21-node network.

SIR Threshold	Throughput (kbps)
5	4095
10	3917
15	3561
20	3561
25	3561
30	3383
35	2493
40	1638
45	0
50	0

In this chapter we implemented the interference aware STDMA and least-cost routing method proposed in Chapter V. The interference introduced by the transmission groups caused poor network throughput performance, particularly in the 21-node network. Additionally, we performed simulations to determine if we could gain a throughput increase by manipulating the SIR threshold but did not see any improvement for less discriminant choices of SIR threshold. We saw a sizeable improvement in network delay by utilizing transmission group scheduling and interference aware least cost routing.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. CONCLUSIONS

A. CONCLUDING REMARKS

Throughout the literature there is ample proof of the value of using a STDMA MAC protocol for improved network performance. In a WSN, this can be exploited to better utilize precious and limited network resources and increase the operating network lifetime; however, many STDMA implementations incur a costly computational penalty, which can mitigate the savings benefits of the optimized schedules they provide.

In this thesis, we presented the idea of a simplified scheduling philosophy using STDMA. As interference is a bottleneck to network performance, we studied the effect of interference when using STDMA integrated with a PhIM. The implementation of the PhIM allowed us to capture link interference metrics. We used the concept of spatial reuse to schedule simultaneous transmissions by implementing a conflict graph. We implemented Dijkstra's algorithm, where the link costs are obtained from SIR calculations determined from the conflict graph. Using these links, we determined the shortest path from a sensor node to the gateway. Via our simulations using MATLAB and QualNet, we showed that our proof-of-concept improves delay in the network. An advantage to our proposed scheme is that it does not require heuristic algorithms or messaging overhead for proper system operation; however, a tradeoff is that our method did not help improve throughput. Nonetheless, we see the results obtained as a proof-of-concept that a deterministic PhIM can provide value in network design and performance despite its algorithmic computational complexity.

B. FUTURE WORK

1. Network Generation Automation and QualNet Integration

In order to create a more flexible and robust scenario generation environment, we suggest the implementation of automated network generation. This should include randomized node placement and calculation of power and interference values for

arbitrary node distances. This generalized approach would permit more interesting and less restrictive scenario development and testing.

The QualNet simulation environment provides a comprehensive simulation environment that is designed to be customizable and allow high-level development at all network layers. In order to establish a more generalized network topology and integrate the PhIM functionality, significant modification to the PHY layer functions included in QualNet is required. By accomplishing the necessary integration steps, very large scenarios can be developed with varying node density to obtain data to verify the usefulness of our approach.

2. Development of a Formal Protocol

In order for a network to operate, there must be a set of rules for the constituent devices to operate under. This requires the formalization of inter-layer communications to pass SIR information from the PHY layer to the network layer to be used for routing. Additionally, the development of messaging components including message header and internal packet structures are needed to disseminate information for network scheduling among the nodes.

3. Testing in a Real-World WSN

Finally, building and testing a WSN using the scheduling considerations of the thesis and verifying delay improvement shown in the simulation results obtained in QualNet would provide practical analysis to our proof-of-concept developed in this thesis. The information gained by a real-world test can be used to determine if the interference calculations are overly optimistic or pessimistic and perhaps lead to a relaxation of some of our simplifying assumptions.

APPENDIX

The appendices that follow document the MATLAB code utilized for creation of the interference based least cost network path.

A. NODE TOPOLOGY FUNCTION FOR 10-NODE NETWORK

This function serves to populate a ten-node network with all nodes separated by 10 meters. The maxx and maxy parameters are used to scale the distance and the node array is used for manual entry of node coordinates. There is a drawFigure argument that can be used to produce a graphical depiction of the network topology.

```
function [node] = topo10(maxx, maxy, drawFigure);
% Generate network topology
% Maxx = maximum x distance for a node to be placed
% Maxy = maximum y distance for a node to be placed
% drawFigure = boolean input 1 for plot 0 for no plot

node = [1 1;2 1;3 1;1 2;2 2;3 2;1 3;2 3;3 3;2 4].*0.1;  %Enter
Coordinates of nodes here
node(:,1) = node(:,1)*maxx;
node(:,2) = node(:,2)*maxy;


if drawFigure >= 1
    colordef none,  whitebg
    figure(1);
    axis equal
    hold on;
    box on;
    plot(node(:, 1), node(:, 2), 'k.', 'MarkerSize', 5);
    title('Network topology');
    xlabel('X meters');
    ylabel('Y meters');
    axis([0, 40, 0, 50]);
    set(gca, 'XTick', [0; 40]);
    set(gca, 'YTick', [50]);
end
return;
```

B. NODE TOPOLOGY FUNCTION FOR 21-NODE NETWORK

This function serves to populate a 21-node network with all nodes separated by 10 meters. The maxx and maxy parameters are used to scale the distance and the node array

is used for manual entry of node coordinates. There is a drawFigure argument that can be used to produce a graphical depiction of the network topology.

```
function [node] = topo21(maxx, maxy, drawFigure);
% Generate network topology
% Maxx = maximum x distance for a node to be placed
% Maxy = maximum y distance for a node to be placed
% drawFigure = boolean input 1 for plot 0 for no plot

node = [1 1;2 1;3 1;4 1;5 1;1 2;2 2;3 2;4 2;5 2;1 3;2 3;3 3;4 3;5 3;1
4;...
        2 4;3 4;4 4;5 4;3 5].*0.1;  %Enter Coordinates of nodes here
node(:,1) = node(:,1)*maxx;
node(:,2) = node(:,2)*maxy;

if drawFigure >= 1
    colordef none,  whitebg
    figure(1);
    axis equal
    hold on;
    box on;
    plot(node(:, 1), node(:, 2), 'k.', 'MarkerSize', 5);
    title('Network topology');
    xlabel('X meters');
    ylabel('Y meters');
    axis([0, 60, 0, 60]);
    set(gca, 'XTick', [0; 60]);
    set(gca, 'YTick', [60]);
end
return;
```

C. DISTANCE CALCULATION FUNCTION

This function is called to calculate the Euclidean distance between nodes.

```
function d = node_dist(x1,y1,x2,y2);
%determines the distance beween nodes
d = sqrt((x2-x1)^2+(y2-y1)^2);
return;
```

D. RECEIVED POWER CALCULATION FUNCTION

This function is called to calculate the received power based on distance between nodes using the path-loss model.

```
function Prx = pathlosscalc(Ptx,d);
%Determines change in power due to distance losses
Prx = Ptx*d^(-2);  %2 is the path loss factor which is typical
%Prx = log10(Prx)
return
```

E. MATLAB SCRIPT TO DETERMINE THE LEAST COST PATH FROM NODE 1 TO NODE 10 IN THE TEN-NODE NETWORK.

This script calculates the SIR and link values as well as the connectivity and conflict graphs for the ten-node network. The function then uses the built in shortestpath() function in MATLAB to determine the shortest path and associated path cost through the network.

```
%5 node SINR link-weight generation

clear

%setup constansts

Ptx = 10*10^(-3);  %Transmit Power of 10mW


%Intiate Topology
nodes = topo10(100,100,0);  %returns coordinates of nodes in order

%calculate distances between nodes

x1 = nodes (1,1);
y1 = nodes (1,2);
x2 = nodes (2,1);
y2 = nodes (2,2);
x3 = nodes (3,1);
y3 = nodes (3,2);
x4 = nodes (4,1);
y4 = nodes (4,2);
x5 = nodes (5,1);
y5 = nodes (5,2);
x6 = nodes (6,1);
y6 = nodes (6,2);
x7 = nodes (7,1);
y7 = nodes (7,2);
x8 = nodes (8,1);
y8 = nodes (8,2);
x9 = nodes (9,1);
y9 = nodes (9,2);
x10 = nodes (10,1);
y10 = nodes (10,2);

%distance from each node to each other node

d12 = node_dist(x1,y1,x2,y2);
d13 = node_dist(x1,y1,x3,y3);
d14 = node_dist(x1,y1,x4,y4);
d15 = node_dist(x1,y1,x5,y5);
d16 = node_dist(x1,y1,x6,y6);
```

```
d17 = node_dist(x1,y1,x7,y7);
d18 = node_dist(x1,y1,x8,y8);
d19 = node_dist(x1,y1,x9,y9);
d110 = node_dist(x1,y1,x10,y10);

d21 = d12;
d23 = node_dist(x2,y2,x3,y3);
d24 = node_dist(x2,y2,x4,y4);
d25 = node_dist(x2,y2,x5,y5);
d26 = node_dist(x2,y2,x6,y6);
d27 = node_dist(x2,y2,x7,y7);
d28 = node_dist(x2,y2,x8,y8);
d29 = node_dist(x2,y2,x9,y9);
d210  = node_dist(x2,y2,x10,y10);

d31 = d13;
d32 = d23;
d34 = node_dist(x3,y3,x4,y4);
d35 = node_dist(x3,y3,x5,y5);
d36 = node_dist(x3,y3,x6,y6);
d37 = node_dist(x3,y3,x7,y7);
d38 = node_dist(x3,y3,x8,y8);
d39 = node_dist(x3,y9,x5,y9);
d310 = node_dist(x3,y3,x10,y10);

d41 = d14;
d42 = d24;
d43 = d34;
d45 = node_dist(x4,y4,x5,y5);
d46 = node_dist(x4,y4,x6,y6);
d47 = node_dist(x4,y4,x7,y7);
d48 = node_dist(x4,y4,x8,y8);
d49 = node_dist(x4,y4,x9,y9);
d410 = node_dist(x4,y4,x10,y10);

d51 = d15;
d52 = d25;
d53 = d35;
d54 = d45;
d56 = node_dist(x5,y5,x6,y6);
d57 = node_dist(x5,y5,x7,y7);
d58 = node_dist(x5,y5,x8,y8);
d59 = node_dist(x5,y5,x9,y9);
d510 = node_dist(x5,y5,x10,y10);

d61 = d16;
d62 = d26;
d63 = d36;
d64 = d46;
d65 = d56;
d67 = node_dist(x6,y6,x7,y7);
d68 = node_dist(x6,y6,x8,y8);
d69 = node_dist(x6,y6,x9,y9);
```

```
d610 = node_dist(x6,y6,x10,y10);

d71 = d17;
d72 = d27;
d73 = d37;
d74 = d47;
d75 = d57;
d76 = d67;
d78 = node_dist(x7,y7,x8,y8);
d79 = node_dist(x7,y7,x9,y9);
d710 = node_dist(x7,y7,x10,y10);

d81 = d18;
d82 = d28;
d83 = d38;
d84 = d48;
d85 = d58;
d86 = d68;
d87 = d78;
d89 = node_dist(x8,y8,x9,y9);
d810 = node_dist(x8,y8,x10,y10);

d91 = d19;
d92 = d29;
d93 = d39;
d94 = d49;
d95 = d59;
d96 = d69;
d97 = d79;
d98 = d89;
d910 = node_dist(x9,y9,x10,y10);

d101 = d110;
d102 = d210;
d103 = d310;
d104 = d410;
d105 = d510;
d106 = d610;
d107 = d710;
d108 = d810;
d109 = d910;
%Calculate the recieved power due to freespace losses

Prx12 = pathlosscalc(Ptx,d12);
Prx13 = pathlosscalc(Ptx,d13);
Prx14 = pathlosscalc(Ptx,d14);
Prx15 = pathlosscalc(Ptx,d15);
Prx16 = pathlosscalc(Ptx,d16);
Prx17 = pathlosscalc(Ptx,d17);
Prx18 = pathlosscalc(Ptx,d18);
Prx19 = pathlosscalc(Ptx,d19);
Prx110 = pathlosscalc(Ptx,d110);
```

```
Prx21 = pathlosscalc(Ptx,d21);
Prx23 = pathlosscalc(Ptx,d23);
Prx24 = pathlosscalc(Ptx,d24);
Prx25 = pathlosscalc(Ptx,d25);
Prx26 = pathlosscalc(Ptx,d26);
Prx27 = pathlosscalc(Ptx,d27);
Prx28 = pathlosscalc(Ptx,d28);
Prx29 = pathlosscalc(Ptx,d29);
Prx210 = pathlosscalc(Ptx,d210);

Prx31 = pathlosscalc(Ptx,d31);
Prx32 = pathlosscalc(Ptx,d32);
Prx34 = pathlosscalc(Ptx,d34);
Prx35 = pathlosscalc(Ptx,d35);
Prx36 = pathlosscalc(Ptx,d36);
Prx37 = pathlosscalc(Ptx,d37);
Prx38 = pathlosscalc(Ptx,d38);
Prx39 = pathlosscalc(Ptx,d39);
Prx310 = pathlosscalc(Ptx,d310);

Prx41 = pathlosscalc(Ptx,d41);
Prx42 = pathlosscalc(Ptx,d42);
Prx43 = pathlosscalc(Ptx,d43);
Prx45 = pathlosscalc(Ptx,d45);
Prx46 = pathlosscalc(Ptx,d46);
Prx47 = pathlosscalc(Ptx,d47);
Prx48 = pathlosscalc(Ptx,d48);
Prx49 = pathlosscalc(Ptx,d49);
Prx410 = pathlosscalc(Ptx,d410);


Prx51 = pathlosscalc(Ptx,d51);
Prx52 = pathlosscalc(Ptx,d52);
Prx53 = pathlosscalc(Ptx,d53);
Prx54 = pathlosscalc(Ptx,d54);
Prx56 = pathlosscalc(Ptx,d56);
Prx57 = pathlosscalc(Ptx,d57);
Prx58 = pathlosscalc(Ptx,d58);
Prx59 = pathlosscalc(Ptx,d59);
Prx510 = pathlosscalc(Ptx,d510);

Prx61 = Prx16;
Prx62 = Prx26;
Prx63 = Prx36;
Prx64 = Prx46;
Prx65 = Prx56;
Prx67 = pathlosscalc(Ptx,d67);
Prx68 = pathlosscalc(Ptx,d68);
Prx69 = pathlosscalc(Ptx,d69);
Prx610 = pathlosscalc(Ptx,d610);

Prx71 = Prx17;
Prx72 = Prx27;
```

```
Prx73 = Prx37;
Prx74 = Prx47;
Prx75 = Prx57;
Prx76 = Prx67;
Prx78 = pathlosscalc(Ptx,d78);
Prx79 = pathlosscalc(Ptx,d79);
Prx710 = pathlosscalc(Ptx,d710);

Prx81 = Prx18;
Prx82 = Prx28;
Prx83 = Prx38;
Prx84 = Prx48;
Prx85 = Prx58;
Prx86 = Prx68;
Prx87 = Prx78;
Prx89 = pathlosscalc(Ptx,d89);
Prx810 = pathlosscalc(Ptx,d810);

Prx91 = Prx19;
Prx92 = Prx29;
Prx93 = Prx39;
Prx94 = Prx49;
Prx95 = Prx59;
Prx96 = Prx69;
Prx97 = Prx79;
Prx98 = Prx89;
Prx910 = pathlosscalc(Ptx,d910);

Prx101 = Prx110;
Prx102 = Prx210;
Prx103 = Prx310;
Prx104 = Prx410;
Prx105 = Prx510;
Prx106 = Prx610;
Prx107 = Prx710;
Prx108 = Prx810;
Prx109 = Prx910;

%Calculate Signal to Interference Ratios

%SIR for transmissions from 1 to 2,4, or 5

SIR2from1 = (Prx21)/(Prx32+Prx72+Prx92);
SIR4from1 = (Prx41)/(Prx34+Prx74+Prx94);
SIR5from1 = (Prx51)/(Prx35+Prx75+Prx95);


%SIR for transmissions from 2 to 1,3,4,5,6

SIR1from2 = Prx12/Prx18;
SIR3from2 = Prx32/Prx38;
SIR4from2 = Prx42/Prx48;
```

```
SIR5from2 = Prx52/Prx58;
SIR6from2 = Prx62/Prx68;

%SIR for transmissions from 3 to 2,5,6

SIR2from3 = Prx23/(Prx21+Prx27+Prx29);
SIR5from3 = Prx53/(Prx51+Prx57+Prx59);
SIR6from3 = Prx63/(Prx61+Prx67+Prx69);

%SIR for transmissions from 4 to 1,2,5,7,8

SIR1from4 = Prx41/Prx61;
SIR2from4 = Prx42/Prx62;
SIR5from4 = Prx45/Prx65;
SIR7from4 = Prx47/Prx67;
SIR8from4 = Prx48/Prx68;

%SIR for transmissions from 5 to 1,2,3,4,6,7,8,9
%Note, 5 has no two hop neighbors
SIR1from5 = Prx51;
SIR2from5 = Prx52;
SIR3from5 = Prx53;
SIR4from5 = Prx54;
SIR6from5 = Prx56;
SIR7from5 = Prx57;
SIR8from5 = Prx58;
SIR9from5 = Prx59;

%SIR for transmissions from 6 to 2,3,5,8,9
SIR2from6 = Prx62/Prx42;
SIR3from6 = Prx63/Prx43;
SIR5from6 = Prx65/Prx45;
SIR8from6 = Prx68/Prx48;
SIR9from6 = Prx69/Prx49;

%SIR for transmissions from 7 to 4,5,8,10
SIR4from7 = Prx74/(Prx14+Prx34+Prx94);
SIR5from7 = Prx75/(Prx15+Prx35+Prx95);
SIR8from7 = Prx87/(Prx18+Prx38+Prx98);
SIR10from7 = Prx710/(Prx110+Prx310+Prx910);

%SIR for transmissions from 8 to 4,5,6,7,9,10
SIR4from8 = Prx84/Prx24;
SIR5from8 = Prx85/Prx25;
SIR6from8 = Prx86/Prx26;
SIR7from8 = Prx87/Prx27;
SIR9from8 = Prx89/Prx29;
SIR10from8 = Prx810/Prx210;

%SIR for transmissions from 9 to 5,6,8,10
SIR5from9 = Prx95/(Prx15+Prx35+Prx75);
SIR6from9 = Prx96/(Prx16+Prx36+Prx76);
```

```
SIR8from9 = Prx98/(Prx18+Prx38+Prx78);
SIR10from9 = Prx910/(Prx110+Prx310+Prx710);

%Fill out the Link Weight matrix for shortest path input
%Note nodes beyond one hop are considered disconnected and assigned a
link
%weight of


W = ([SIR4from1 SIR2from1 SIR5from1 SIR4from2 SIR5from2 SIR6from2 ...
     SIR2from3 SIR5from3 SIR6from3 SIR5from4 SIR5from6 SIR8from4 ...
     SIR8from6 SIR7from4 SIR9from6 SIR8from7 SIR8from9 SIR10from7 ...
     SIR10from8 SIR10from9 1]).^(-1);


DG = sparse([1 1 1 2 2 2 3 3 3 4 6 4 6 4 6 7 9 7 8 9 10],[4 2 5 4 5 6 2
5 6 5 5 8 8 7 9 8 8 10 10 10 9],W);

%h = view(biograph(DG,[],'ShowWeights','on'))

[dist, path, pred] = graphshortestpath(DG, 1, 10)
```

F. MATLAB SCRIPT TO DETERMINE THE LEAST COST PATH FROM NODE 1 TO NODE 21 IN THE 21-NODE NETWORK.

This script calculates the SIR and link values as well as the connectivity and conflict graphs for the 21-node network. The function then uses the built in shortestpath() function in MATLAB to determine the shortest path and associated path cost through the network.

```
%5 node SINR link-weight generation

clear

%setup constansts

Ptx = 10*10^(-3);  %Transmit Power of 10mW


%Intiate Topology
nodes = topo21(100,100,0);  %returns coordinates of nodes in order

%calculate distances between nodes

x1 = nodes (1,1);
y1 = nodes (1,2);
x2 = nodes (2,1);
```

```
y2 = nodes (2,2);
x3 = nodes (3,1);
y3 = nodes (3,2);
x4 = nodes (4,1);
y4 = nodes (4,2);
x5 = nodes (5,1);
y5 = nodes (5,2);
x6 = nodes (6,1);
y6 = nodes (6,2);
x7 = nodes (7,1);
y7 = nodes (7,2);
x8 = nodes (8,1);
y8 = nodes (8,2);
x9 = nodes (9,1);
y9 = nodes (9,2);
x10 = nodes (10,1);
y10 = nodes (10,2);
x11 = nodes (11,1);
y11 = nodes (11,2);
x12 = nodes (12,1);
y12 = nodes (12,2);
x13 = nodes (13,1);
y13 = nodes (13,2);
x14 = nodes (14,1);
y14 = nodes (14,2);
x15 = nodes (15,1);
y15 = nodes (15,2);
x16 = nodes (16,1);
y16 = nodes (16,2);
x17 = nodes (17,1);
y17 = nodes (17,2);
x18 = nodes (18,1);
y18 = nodes (18,2);
x19 = nodes (19,1);
y19 = nodes (19,2);
x20 = nodes (20,1);
y20 = nodes (20,2);
x21 = nodes (21,1);
y21 = nodes (21,2);

%distance from each node to each other node

d12 = node_dist(x1,y1,x2,y2);
d13 = node_dist(x1,y1,x3,y3);
d14 = node_dist(x1,y1,x4,y4);
d15 = node_dist(x1,y1,x5,y5);
d16 = node_dist(x1,y1,x6,y6);
d17 = node_dist(x1,y1,x7,y7);
d18 = node_dist(x1,y1,x8,y8);
d19 = node_dist(x1,y1,x9,y9);
d110 = node_dist(x1,y1,x10,y10);
d111 = node_dist(x1,y1,x11,y11);
d112 = node_dist(x1,y1,x12,y12);
d113 = node_dist(x1,y1,x13,y13);
```

```
d114 = node_dist(x1,y1,x14,y14);
d115 = node_dist(x1,y1,x15,y15);
d116 = node_dist(x1,y1,x16,y16);
d117 = node_dist(x1,y1,x17,y17);
d118 = node_dist(x1,y1,x18,y18);
d119 = node_dist(x1,y1,x19,y19);
d120 = node_dist(x1,y1,x20,y20);
d121 = node_dist(x1,y1,x21,y21);

d21 = d12;
d23 = node_dist(x2,y2,x3,y3);
d24 = node_dist(x2,y2,x4,y4);
d25 = node_dist(x2,y2,x5,y5);
d26 = node_dist(x2,y2,x6,y6);
d27 = node_dist(x2,y2,x7,y7);
d28 = node_dist(x2,y2,x8,y8);
d29 = node_dist(x2,y2,x9,y9);
d210  = node_dist(x2,y2,x10,y10);
d211  = node_dist(x2,y2,x11,y11);
d212  = node_dist(x2,y2,x12,y12);
d213  = node_dist(x2,y2,x13,y13);
d214  = node_dist(x2,y2,x14,y14);
d215  = node_dist(x2,y2,x15,y15);
d216  = node_dist(x2,y2,x16,y16);
d217  = node_dist(x2,y2,x17,y17);
d218  = node_dist(x2,y2,x18,y18);
d219  = node_dist(x2,y2,x19,y19);
d220  = node_dist(x2,y2,x20,y20);
d221  = node_dist(x2,y2,x21,y21);


d31 = d13;
d32 = d23;
d34 = node_dist(x3,y3,x4,y4);
d35 = node_dist(x3,y3,x5,y5);
d36 = node_dist(x3,y3,x6,y6);
d37 = node_dist(x3,y3,x7,y7);
d38 = node_dist(x3,y3,x8,y8);
d39 = node_dist(x3,y9,x5,y9);
d310 = node_dist(x3,y3,x10,y10);
d311 = node_dist(x3,y3,x11,y11);
d312 = node_dist(x3,y3,x12,y12);
d313 = node_dist(x3,y3,x13,y13);
d314 = node_dist(x3,y3,x14,y14);
d315 = node_dist(x3,y3,x15,y15);
d316 = node_dist(x3,y3,x16,y16);
d317 = node_dist(x3,y3,x17,y17);
d318 = node_dist(x3,y3,x18,y18);
d319 = node_dist(x3,y3,x19,y19);
d320 = node_dist(x3,y3,x20,y20);
d321 = node_dist(x3,y3,x21,y21);

d41 = d13;
```

```
d42 = d24;
d43 = d34;
d45 = node_dist(x4,y4,x5,y5);
d46 = node_dist(x4,y4,x6,y6);
d47 = node_dist(x4,y4,x7,y7);
d48 = node_dist(x4,y4,x8,y8);
d49 = node_dist(x4,y4,x9,y9);
d410 = node_dist(x4,y4,x10,y10);
d411 = node_dist(x4,y4,x11,y11);
d412 = node_dist(x4,y4,x12,y12);
d413 = node_dist(x4,y4,x13,y13);
d414 = node_dist(x4,y4,x14,y14);
d415 = node_dist(x4,y4,x15,y15);
d416 = node_dist(x4,y4,x16,y16);
d417 = node_dist(x4,y4,x17,y17);
d418 = node_dist(x4,y4,x18,y18);
d419 = node_dist(x4,y4,x19,y19);
d420 = node_dist(x4,y4,x20,y20);
d421 = node_dist(x4,y4,x21,y21);

d51 = d15;
d52 = d25;
d53 = d35;
d54 = d45;
d56 = node_dist(x5,y5,x6,y6);
d57 = node_dist(x5,y5,x7,y7);
d58 = node_dist(x5,y5,x8,y8);
d59 = node_dist(x5,y5,x9,y9);
d510 = node_dist(x5,y5,x10,y10);
d511 = node_dist(x5,y5,x11,y11);
d512 = node_dist(x5,y5,x12,y12);
d513 = node_dist(x5,y5,x13,y13);
d514 = node_dist(x5,y5,x14,y14);
d515 = node_dist(x5,y5,x15,y15);
d516 = node_dist(x5,y5,x16,y16);
d517 = node_dist(x5,y5,x17,y17);
d518 = node_dist(x5,y5,x18,y18);
d519 = node_dist(x5,y5,x19,y19);
d520 = node_dist(x5,y5,x20,y20);
d521 = node_dist(x5,y5,x21,y21);

d61 = d16;
d62 = d26;
d63 = d36;
d64 = d46;
d65 = d56;
d67 = node_dist(x6,y6,x7,y7);
d68 = node_dist(x6,y6,x8,y8);
d69 = node_dist(x6,y6,x9,y9);
d610 = node_dist(x6,y6,x10,y10);
d611 = node_dist(x6,y6,x11,y11);
d612 = node_dist(x6,y6,x12,y12);
d613 = node_dist(x6,y6,x13,y13);
d614 = node_dist(x6,y6,x14,y14);
```

```
d615 = node_dist(x6,y6,x15,y15);
d616 = node_dist(x6,y6,x16,y16);
d617 = node_dist(x6,y6,x17,y17);
d618 = node_dist(x6,y6,x18,y18);
d619 = node_dist(x6,y6,x19,y19);
d620 = node_dist(x6,y6,x20,y20);
d621 = node_dist(x6,y6,x21,y21);

d71 = d17;
d72 = d27;
d73 = d37;
d74 = d47;
d75 = d57;
d76 = d67;
d78 = node_dist(x7,y7,x8,y8);
d79 = node_dist(x7,y7,x9,y9);
d710 = node_dist(x7,y7,x10,y10);
d711 = node_dist(x7,y7,x11,y11);
d712 = node_dist(x7,y7,x12,y12);
d713 = node_dist(x7,y7,x13,y13);
d714 = node_dist(x7,y7,x14,y14);
d715 = node_dist(x7,y7,x15,y15);
d716 = node_dist(x7,y7,x16,y16);
d717 = node_dist(x7,y7,x17,y17);
d718 = node_dist(x7,y7,x18,y18);
d719 = node_dist(x7,y7,x19,y19);
d720 = node_dist(x7,y7,x20,y20);
d721 = node_dist(x7,y7,x21,y21);

d81 = d18;
d82 = d28;
d83 = d38;
d84 = d48;
d85 = d58;
d86 = d68;
d87 = d78;
d89 = node_dist(x8,y8,x9,y9);
d810 = node_dist(x8,y8,x10,y10);
d811 = node_dist(x8,y8,x11,y11);
d812 = node_dist(x8,y8,x12,y12);
d813 = node_dist(x8,y8,x13,y13);
d814 = node_dist(x8,y8,x14,y14);
d815 = node_dist(x8,y8,x15,y15);
d816 = node_dist(x8,y8,x16,y16);
d817 = node_dist(x8,y8,x17,y17);
d818 = node_dist(x8,y8,x18,y18);
d819 = node_dist(x8,y8,x19,y19);
d820 = node_dist(x8,y8,x20,y20);
d821 = node_dist(x8,y8,x21,y21);

d91 = d19;
d92 = d29;
d93 = d39;
d94 = d49;
```

```
d95 = d59;
d96 = d69;
d97 = d79;
d98 = d89;
d910 = node_dist(x9,y9,x10,y10);
d911 = node_dist(x9,y9,x11,y11);
d912 = node_dist(x9,y9,x12,y12);
d913 = node_dist(x9,y9,x13,y13);
d914 = node_dist(x9,y9,x14,y14);
d915 = node_dist(x9,y9,x15,y15);
d916 = node_dist(x9,y9,x16,y16);
d917 = node_dist(x9,y9,x17,y17);
d918 = node_dist(x9,y9,x18,y18);
d919 = node_dist(x9,y9,x19,y19);
d920 = node_dist(x9,y9,x20,y20);
d921 = node_dist(x9,y9,x21,y21);

d101 = d110;
d102 = d210;
d103 = d310;
d104 = d410;
d105 = d510;
d106 = d610;
d107 = d710;
d108 = d810;
d109 = d910;
d1011 = node_dist(x10,y10,x11,y11);
d1012 = node_dist(x10,y10,x12,y12);
d1013 = node_dist(x10,y10,x13,y13);
d1014 = node_dist(x10,y10,x14,y14);
d1015 = node_dist(x10,y10,x15,y15);
d1016 = node_dist(x10,y10,x16,y16);
d1017 = node_dist(x10,y10,x17,y17);
d1018 = node_dist(x10,y10,x18,y18);
d1019 = node_dist(x10,y10,x19,y19);
d1020 = node_dist(x10,y10,x20,y20);
d1021 = node_dist(x10,y10,x21,y21);

d111 = d111;
d112 = d211;
d113 = d311;
d114 = d411;
d115 = d511;
d116 = d611;
d117 = d711;
d118 = d811;
d119 = d911;
d1110 = d1011;
d1112 = node_dist(x11,y11,x12,y12);
d1113 = node_dist(x11,y11,x13,y13);
d1114 = node_dist(x11,y11,x14,y14);
d1115 = node_dist(x11,y11,x15,y15);
d1116 = node_dist(x11,y11,x16,y16);
d1117 = node_dist(x11,y11,x17,y17);
```

```
d1118 = node_dist(x11,y11,x18,y18);
d1119 = node_dist(x11,y11,x19,y19);
d1120 = node_dist(x11,y11,x20,y20);
d1121 = node_dist(x11,y11,x21,y21);

d121 = d112;
d122 = d212;
d123 = d312;
d124 = d412;
d125 = d512;
d126 = d612;
d127 = d712;
d128 = d812;
d129 = d912;
d1210 = d1012;
d1211 = d1112;
d1213 = node_dist(x12,y12,x13,y13);
d1214 = node_dist(x12,y12,x14,y14);
d1215 = node_dist(x12,y12,x15,y15);
d1216 = node_dist(x12,y12,x16,y16);
d1217 = node_dist(x12,y12,x17,y17);
d1218 = node_dist(x12,y12,x18,y18);
d1219 = node_dist(x12,y12,x19,y19);
d1220 = node_dist(x12,y12,x20,y20);
d1221 = node_dist(x12,y12,x21,y21);

d131 = d113;
d132 = d213;
d133 = d313;
d134 = d413;
d135 = d513;
d136 = d613;
d137 = d713;
d138 = d813;
d139 = d913;
d1310 = d1013;
d1311 = d1113;
d1312 = d1213;
d1314 = node_dist(x13,y13,x14,y14);
d1315 = node_dist(x13,y13,x15,y15);
d1316 = node_dist(x13,y13,x16,y16);
d1317 = node_dist(x13,y13,x17,y17);
d1318 = node_dist(x13,y13,x18,y18);
d1319 = node_dist(x13,y13,x19,y19);
d1320 = node_dist(x13,y13,x20,y20);
d1321 = node_dist(x13,y13,x21,y21);

d141 = d114;
d142 = d214;
d143 = d314;
d144 = d414;
d145 = d514;
d146 = d614;
d147 = d714;
```

```
d148 = d814;
d149 = d914;
d1410 = d1014;
d1411 = d1114;
d1412 = d1214;
d1413 = d1314;
d1415 = node_dist(x14,y14,x15,y15);
d1416 = node_dist(x14,y14,x16,y16);
d1417 = node_dist(x14,y14,x17,y17);
d1418 = node_dist(x14,y14,x18,y18);
d1419 = node_dist(x14,y14,x19,y19);
d1420 = node_dist(x14,y14,x20,y20);
d1421 = node_dist(x14,y14,x21,y21);

d151 = d115;
d152 = d215;
d153 = d315;
d154 = d415;
d156 = d615;
d157 = d715;
d158 = d815;
d159 = d915;
d1510 = d1015;
d1511 = d1115;
d1512 = d1215;
d1513 = d1315;
d1514 = d1415;
d1516 = node_dist(x15,y15,x16,y16);
d1517 = node_dist(x15,y15,x17,y17);
d1518 = node_dist(x15,y15,x18,y18);
d1519 = node_dist(x15,y15,x19,y19);
d1520 = node_dist(x15,y15,x20,y20);
d1521 = node_dist(x15,y15,x21,y21);

d161 = d116;
d162 = d216;
d163 = d316;
d164 = d416;
d165 = d516;
d166 = d616;
d167 = d716;
d168 = d816;
d169 = d916;
d1610 = d1016;
d1611 = d1116;
d1612 = d1216;
d1613 = d1316;
d1614 = d1416;
d1615 = d1516;
d1617 = node_dist(x16,y16,x17,y17);
d1618 = node_dist(x16,y16,x18,y18);
d1619 = node_dist(x16,y16,x19,y19);
d1620 = node_dist(x16,y16,x20,y20);
d1621 = node_dist(x16,y16,x20,y21);
```

```
d171 = d117;
d172 = d217;
d173 = d317;
d174 = d417;
d175 = d517;
d176 = d617;
d177 = d717;
d178 = d817;
d179 = d917;
d1710 = d1017;
d1711 = d1117;
d1712 = d1217;
d1713 = d1317;
d1714 = d1417;
d1715 = d1517;
d1716 = d1617;
d1718 = node_dist(x17,y17,x18,y18);
d1719 = node_dist(x17,y17,x19,y19);
d1720 = node_dist(x17,y17,x20,y20);
d1721 = node_dist(x17,y17,x21,y21);

d181 = d118;
d182 = d218;
d183 = d318;
d184 = d418;
d185 = d518;
d186 = d618;
d187 = d718;
d188 = d818;
d189 = d918;
d1810 = d1018;
d1811 = d1118;
d1812 = d1218;
d1813 = d1318;
d1814 = d1418;
d1815 = d1518;
d1816 = d1618;
d1817 = d1718;
d1819 = node_dist(x18,y18,x19,y19);
d1820 = node_dist(x18,y18,x20,y20);
d1821 = node_dist(x18,y18,x21,y21);

d191 = d119;
d192 = d219;
d193 = d319;
d194 = d419;
d195 = d519;
d196 = d619;
d197 = d719;
d198 = d819;
d199 = d919;
d1910 = d1019;
d1911 = d1119;
```

```
d1912 = d1219;
d1913 = d1319;
d1914 = d1419;
d1915 = d1519;
d1916 = d1619;
d1917 = d1719;
d1918 = d1819;
d1920 = node_dist(x19,y19,x20,y20);
d1921 = node_dist(x19,y19,x21,y21);

d201 = d120;
d202 = d220;
d203 = d320;
d204 = d420;
d205 = d520;
d206 = d620;
d207 = d720;
d208 = d820;
d209 = d920;
d2010 = d1020;
d2011 = d1120;
d2012 = d1220;
d2013 = d1320;
d2014 = d1420;
d2015 = d1520;
d2016 = d1620;
d2017 = d1720;
d2018 = d1820;
d2019 = d1920;
d2021 = node_dist(x20,y20,x21,y21);

d211 = d121;
d212 = d221;
d213 = d321;
d214 = d421;
d215 = d521;
d216 = d621;
d217 = d721;
d218 = d821;
d219 = d921;
d2110 = d1021;
d2111 = d1121;
d2112 = d1221;
d2113 = d1321;
d2114 = d1421;
d2115 = d1521;
d2116 = d1621;
d2117 = d1721;
d2118 = d1821;
d2119 = d1921;
d2120 = d2021;

%Calculate the recieved power due to freespace losses
```

```
Prx12 = pathlosscalc(Ptx,d12);
Prx13 = pathlosscalc(Ptx,d13);
Prx14 = pathlosscalc(Ptx,d14);
Prx15 = pathlosscalc(Ptx,d15);
Prx16 = pathlosscalc(Ptx,d16);
Prx17 = pathlosscalc(Ptx,d17);
Prx18 = pathlosscalc(Ptx,d18);
Prx19 = pathlosscalc(Ptx,d19);
Prx110 = pathlosscalc(Ptx,d110);
Prx111 = pathlosscalc(Ptx,d111);
Prx112 = pathlosscalc(Ptx,d112);
Prx113 = pathlosscalc(Ptx,d113);
Prx114 = pathlosscalc(Ptx,d114);
Prx115 = pathlosscalc(Ptx,d115);
Prx116 = pathlosscalc(Ptx,d116);
Prx117 = pathlosscalc(Ptx,d117);
Prx118 = pathlosscalc(Ptx,d118);
Prx119 = pathlosscalc(Ptx,d119);
Prx120 = pathlosscalc(Ptx,d120);
Prx121 = pathlosscalc(Ptx,d121);

Prx21 = pathlosscalc(Ptx,d21);
Prx23 = pathlosscalc(Ptx,d23);
Prx24 = pathlosscalc(Ptx,d24);
Prx25 = pathlosscalc(Ptx,d25);
Prx26 = pathlosscalc(Ptx,d26);
Prx27 = pathlosscalc(Ptx,d27);
Prx28 = pathlosscalc(Ptx,d28);
Prx29 = pathlosscalc(Ptx,d29);
Prx210 = pathlosscalc(Ptx,d210);
Prx211 = pathlosscalc(Ptx,d211);
Prx212 = pathlosscalc(Ptx,d212);
Prx213 = pathlosscalc(Ptx,d213);
Prx214 = pathlosscalc(Ptx,d214);
Prx215 = pathlosscalc(Ptx,d215);
Prx216 = pathlosscalc(Ptx,d216);
Prx217 = pathlosscalc(Ptx,d217);
Prx218 = pathlosscalc(Ptx,d218);
Prx219 = pathlosscalc(Ptx,d219);
Prx220 = pathlosscalc(Ptx,d220);
Prx221 = pathlosscalc(Ptx,d221);

Prx31 = pathlosscalc(Ptx,d31);
Prx32 = pathlosscalc(Ptx,d32);
Prx34 = pathlosscalc(Ptx,d34);
Prx35 = pathlosscalc(Ptx,d35);
Prx36 = pathlosscalc(Ptx,d36);
Prx37 = pathlosscalc(Ptx,d37);
Prx38 = pathlosscalc(Ptx,d38);
Prx39 = pathlosscalc(Ptx,d39);
Prx310 = pathlosscalc(Ptx,d310);
Prx311 = pathlosscalc(Ptx,d311);
Prx312 = pathlosscalc(Ptx,d312);
```

```
Prx313 = pathlosscalc(Ptx,d313);
Prx314 = pathlosscalc(Ptx,d314);
Prx315 = pathlosscalc(Ptx,d315);
Prx316 = pathlosscalc(Ptx,d316);
Prx317 = pathlosscalc(Ptx,d317);
Prx318 = pathlosscalc(Ptx,d318);
Prx319 = pathlosscalc(Ptx,d319);
Prx320 = pathlosscalc(Ptx,d320);
Prx321 = pathlosscalc(Ptx,d321);

Prx41 = pathlosscalc(Ptx,d41);
Prx42 = pathlosscalc(Ptx,d42);
Prx43 = pathlosscalc(Ptx,d43);
Prx45 = pathlosscalc(Ptx,d45);
Prx46 = pathlosscalc(Ptx,d46);
Prx47 = pathlosscalc(Ptx,d47);
Prx48 = pathlosscalc(Ptx,d48);
Prx49 = pathlosscalc(Ptx,d49);
Prx410 = pathlosscalc(Ptx,d410);
Prx411 = pathlosscalc(Ptx,d411);
Prx412 = pathlosscalc(Ptx,d412);
Prx413 = pathlosscalc(Ptx,d413);
Prx414 = pathlosscalc(Ptx,d414);
Prx415 = pathlosscalc(Ptx,d415);
Prx416 = pathlosscalc(Ptx,d416);
Prx417 = pathlosscalc(Ptx,d417);
Prx418 = pathlosscalc(Ptx,d418);
Prx419 = pathlosscalc(Ptx,d419);
Prx420 = pathlosscalc(Ptx,d420);
Prx421 = pathlosscalc(Ptx,d421);

Prx51 = pathlosscalc(Ptx,d51);
Prx52 = pathlosscalc(Ptx,d52);
Prx53 = pathlosscalc(Ptx,d53);
Prx54 = pathlosscalc(Ptx,d54);
Prx56 = pathlosscalc(Ptx,d56);
Prx57 = pathlosscalc(Ptx,d57);
Prx58 = pathlosscalc(Ptx,d58);
Prx59 = pathlosscalc(Ptx,d59);
Prx510 = pathlosscalc(Ptx,d510);
Prx511 = pathlosscalc(Ptx,d511);
Prx512 = pathlosscalc(Ptx,d512);
Prx513 = pathlosscalc(Ptx,d513);
Prx514 = pathlosscalc(Ptx,d514);
Prx515 = pathlosscalc(Ptx,d515);
Prx516 = pathlosscalc(Ptx,d516);
Prx517 = pathlosscalc(Ptx,d517);
Prx518 = pathlosscalc(Ptx,d518);
Prx519 = pathlosscalc(Ptx,d519);
Prx520 = pathlosscalc(Ptx,d520);
Prx521 = pathlosscalc(Ptx,d521);

Prx61 = Prx16;
Prx62 = Prx26;
```

```
Prx63 = Prx36;
Prx64 = Prx46;
Prx65 = Prx56;
Prx67 = pathlosscalc(Ptx,d67);
Prx68 = pathlosscalc(Ptx,d68);
Prx69 = pathlosscalc(Ptx,d69);
Prx610 = pathlosscalc(Ptx,d610);
Prx611 = pathlosscalc(Ptx,d611);
Prx612 = pathlosscalc(Ptx,d612);
Prx613 = pathlosscalc(Ptx,d613);
Prx614 = pathlosscalc(Ptx,d614);
Prx615 = pathlosscalc(Ptx,d615);
Prx616 = pathlosscalc(Ptx,d616);
Prx617 = pathlosscalc(Ptx,d617);
Prx618 = pathlosscalc(Ptx,d618);
Prx619 = pathlosscalc(Ptx,d619);
Prx620 = pathlosscalc(Ptx,d620);
Prx621 = pathlosscalc(Ptx,d621);

Prx71 = Prx17;
Prx72 = Prx27;
Prx73 = Prx37;
Prx74 = Prx47;
Prx75 = Prx57;
Prx76 = Prx67;
Prx78 = pathlosscalc(Ptx,d78);
Prx79 = pathlosscalc(Ptx,d79);
Prx710 = pathlosscalc(Ptx,d710);
Prx711 = pathlosscalc(Ptx,d711);
Prx712 = pathlosscalc(Ptx,d712);
Prx713 = pathlosscalc(Ptx,d713);
Prx714 = pathlosscalc(Ptx,d714);
Prx715 = pathlosscalc(Ptx,d715);
Prx716 = pathlosscalc(Ptx,d716);
Prx717 = pathlosscalc(Ptx,d717);
Prx718 = pathlosscalc(Ptx,d718);
Prx719 = pathlosscalc(Ptx,d719);
Prx720 = pathlosscalc(Ptx,d720);
Prx721 = pathlosscalc(Ptx,d721);

Prx81 = Prx18;
Prx82 = Prx28;
Prx83 = Prx38;
Prx84 = Prx48;
Prx85 = Prx58;
Prx86 = Prx68;
Prx87 = Prx78;
Prx89 = pathlosscalc(Ptx,d89);
Prx810 = pathlosscalc(Ptx,d810);
Prx811 = pathlosscalc(Ptx,d811);
Prx812 = pathlosscalc(Ptx,d812);
Prx813 = pathlosscalc(Ptx,d813);
Prx814 = pathlosscalc(Ptx,d814);
Prx815 = pathlosscalc(Ptx,d815);
```

```
Prx816 = pathlosscalc(Ptx,d816);
Prx817 = pathlosscalc(Ptx,d817);
Prx818 = pathlosscalc(Ptx,d818);
Prx819 = pathlosscalc(Ptx,d819);
Prx820 = pathlosscalc(Ptx,d820);
Prx821 = pathlosscalc(Ptx,d821);

Prx91 = Prx19;
Prx92 = Prx29;
Prx93 = Prx39;
Prx94 = Prx49;
Prx95 = Prx59;
Prx96 = Prx69;
Prx97 = Prx79;
Prx98 = Prx89;
Prx910 = pathlosscalc(Ptx,d910);
Prx911 = pathlosscalc(Ptx,d911);
Prx912 = pathlosscalc(Ptx,d912);
Prx913 = pathlosscalc(Ptx,d913);
Prx914 = pathlosscalc(Ptx,d914);
Prx915 = pathlosscalc(Ptx,d915);
Prx916 = pathlosscalc(Ptx,d916);
Prx917 = pathlosscalc(Ptx,d917);
Prx918 = pathlosscalc(Ptx,d918);
Prx919 = pathlosscalc(Ptx,d919);
Prx920 = pathlosscalc(Ptx,d920);
Prx921 = pathlosscalc(Ptx,d921);

Prx101 = Prx110;
Prx102 = Prx210;
Prx103 = Prx310;
Prx104 = Prx410;
Prx105 = Prx510;
Prx106 = Prx610;
Prx107 = Prx710;
Prx108 = Prx810;
Prx109 = Prx910;
Prx1011 = pathlosscalc(Ptx,d1011);
Prx1012 = pathlosscalc(Ptx,d1012);
Prx1013 = pathlosscalc(Ptx,d1013);
Prx1014 = pathlosscalc(Ptx,d1014);
Prx1015 = pathlosscalc(Ptx,d1015);
Prx1016 = pathlosscalc(Ptx,d1016);
Prx1017 = pathlosscalc(Ptx,d1017);
Prx1018 = pathlosscalc(Ptx,d1018);
Prx1019 = pathlosscalc(Ptx,d1019);
Prx1020 = pathlosscalc(Ptx,d1020);
Prx1021 = pathlosscalc(Ptx,d1021);

Prx111 = Prx111;
Prx112 = Prx211;
Prx113 = Prx311;
Prx114 = Prx411;
Prx115 = Prx511;
```

```
Prx116 = Prx611;
Prx117 = Prx711;
Prx118 = Prx811;
Prx119 = Prx911;
Prx1110 = Prx1011;
Prx1112 = pathlosscalc(Ptx,d1112);
Prx1113 = pathlosscalc(Ptx,d1113);
Prx1114 = pathlosscalc(Ptx,d1114);
Prx1115 = pathlosscalc(Ptx,d1115);
Prx1116 = pathlosscalc(Ptx,d1116);
Prx1117 = pathlosscalc(Ptx,d1117);
Prx1118 = pathlosscalc(Ptx,d1118);
Prx1119 = pathlosscalc(Ptx,d1119);
Prx1120 = pathlosscalc(Ptx,d1120);
Prx1121 = pathlosscalc(Ptx,d1121);

Prx121 = Prx112;
Prx122 = Prx212;
Prx123 = Prx312;
Prx124 = Prx412;
Prx125 = Prx512;
Prx126 = Prx612;
Prx127 = Prx712;
Prx128 = Prx812;
Prx129 = Prx912;
Prx1210 = Prx1012;
Prx1211 = Prx1112;
Prx1213 = pathlosscalc(Ptx,d1213);
Prx1214 = pathlosscalc(Ptx,d1214);
Prx1215 = pathlosscalc(Ptx,d1215);
Prx1216 = pathlosscalc(Ptx,d1216);
Prx1217 = pathlosscalc(Ptx,d1217);
Prx1218 = pathlosscalc(Ptx,d1218);
Prx1219 = pathlosscalc(Ptx,d1219);
Prx1220 = pathlosscalc(Ptx,d1220);
Prx1221 = pathlosscalc(Ptx,d1221);

Prx131 = Prx113;
Prx132 = Prx213;
Prx133 = Prx313;
Prx134 = Prx413;
Prx135 = Prx513;
Prx136 = Prx613;
Prx137 = Prx713;
Prx138 = Prx813;
Prx139 = Prx913;
Prx1310 = Prx1013;
Prx1311 = Prx1113;
Prx1312 = Prx1213;
Prx1314 = pathlosscalc(Ptx,d1314);
Prx1315 = pathlosscalc(Ptx,d1315);
Prx1316 = pathlosscalc(Ptx,d1316);
Prx1317 = pathlosscalc(Ptx,d1317);
Prx1318 = pathlosscalc(Ptx,d1318);
```

```
Prx1319 = pathlosscalc(Ptx,d1319);
Prx1320 = pathlosscalc(Ptx,d1320);
Prx1321 = pathlosscalc(Ptx,d1321);

Prx141 = Prx114;
Prx142 = Prx214;
Prx143 = Prx314;
Prx144 = Prx414;
Prx145 = Prx514;
Prx146 = Prx614;
Prx147 = Prx714;
Prx148 = Prx814;
Prx149 = Prx914;
Prx1410 = Prx1014;
Prx1411 = Prx1114;
Prx1412 = Prx1214;
Prx1413 = Prx1314;
Prx1415 = pathlosscalc(Ptx,d1415);
Prx1416 = pathlosscalc(Ptx,d1416);
Prx1417 = pathlosscalc(Ptx,d1417);
Prx1418 = pathlosscalc(Ptx,d1418);
Prx1419 = pathlosscalc(Ptx,d1419);
Prx1420 = pathlosscalc(Ptx,d1420);
Prx1421 = pathlosscalc(Ptx,d1421);

Prx151 = Prx115;
Prx152 = Prx215;
Prx153 = Prx315;
Prx154 = Prx415;
Prx155 = Prx515;
Prx156 = Prx615;
Prx157 = Prx715;
Prx158 = Prx815;
Prx159 = Prx915;
Prx1510 = Prx1015;
Prx1511 = Prx1115;
Prx1512 = Prx1215;
Prx1513 = Prx1315;
Prx1514 = Prx1415;
Prx1516 = pathlosscalc(Ptx,d1516);
Prx1517 = pathlosscalc(Ptx,d1517);
Prx1518 = pathlosscalc(Ptx,d1518);
Prx1519 = pathlosscalc(Ptx,d1519);
Prx1520 = pathlosscalc(Ptx,d1520);
Prx1521 = pathlosscalc(Ptx,d1521);

Prx161 = Prx116;
Prx162 = Prx216;
Prx163 = Prx316;
Prx164 = Prx416;
Prx165 = Prx516;
Prx166 = Prx616;
Prx167 = Prx716;
Prx168 = Prx816;
```

```
Prx169 = Prx916;
Prx1610 = Prx1016;
Prx1611 = Prx1116;
Prx1612 = Prx1216;
Prx1613 = Prx1316;
Prx1614 = Prx1416;
Prx1615 = Prx1516;
Prx1617 = pathlosscalc(Ptx,d1617);
Prx1618 = pathlosscalc(Ptx,d1618);
Prx1619 = pathlosscalc(Ptx,d1619);
Prx1620 = pathlosscalc(Ptx,d1620);
Prx1621 = pathlosscalc(Ptx,d1621);

Prx171 = Prx117;
Prx172 = Prx217;
Prx173 = Prx317;
Prx174 = Prx417;
Prx175 = Prx517;
Prx176 = Prx617;
Prx177 = Prx717;
Prx178 = Prx817;
Prx179 = Prx917;
Prx1710 = Prx1017;
Prx1711 = Prx1117;
Prx1712 = Prx1217;
Prx1713 = Prx1317;
Prx1714 = Prx1417;
Prx1715 = Prx1517;
Prx1716 = Prx1617;
Prx1718 = pathlosscalc(Ptx,d1718);
Prx1719 = pathlosscalc(Ptx,d1719);
Prx1720 = pathlosscalc(Ptx,d1720);
Prx1721 = pathlosscalc(Ptx,d1721);

Prx181 = Prx118;
Prx182 = Prx218;
Prx183 = Prx318;
Prx184 = Prx418;
Prx185 = Prx518;
Prx186 = Prx618;
Prx187 = Prx718;
Prx188 = Prx818;
Prx189 = Prx918;
Prx1810 = Prx1018;
Prx1811 = Prx1118;
Prx1812 = Prx1218;
Prx1813 = Prx1318;
Prx1814 = Prx1418;
Prx1815 = Prx1518;
Prx1816 = Prx1618;
Prx1817 = Prx1718;
Prx1819 = pathlosscalc(Ptx,d1819);
Prx1820 = pathlosscalc(Ptx,d1820);
Prx1821 = pathlosscalc(Ptx,d1821);
```

```
Prx191 = Prx119;
Prx192 = Prx219;
Prx193 = Prx319;
Prx194 = Prx419;
Prx195 = Prx519;
Prx196 = Prx619;
Prx197 = Prx719;
Prx198 = Prx819;
Prx199 = Prx919;
Prx1910 = Prx1019;
Prx1911 = Prx1119;
Prx1912 = Prx1219;
Prx1913 = Prx1319;
Prx1914 = Prx1419;
Prx1915 = Prx1519;
Prx1916 = Prx1619;
Prx1917 = Prx1719;
Prx1918 = Prx1819;
Prx1920 = pathlosscalc(Ptx,d1920);
Prx1921 = pathlosscalc(Ptx,d1921);

Prx201 = Prx120;
Prx202 = Prx220;
Prx203 = Prx320;
Prx204 = Prx420;
Prx205 = Prx520;
Prx206 = Prx620;
Prx207 = Prx720;
Prx208 = Prx820;
Prx209 = Prx920;
Prx2010 = Prx1020;
Prx2011 = Prx1120;
Prx2012 = Prx1220;
Prx2013 = Prx1320;
Prx2014 = Prx1420;
Prx2015 = Prx1520;
Prx2016 = Prx1620;
Prx2017 = Prx1720;
Prx2018 = Prx1820;
Prx2019 = Prx1920;
Prx2021 = pathlosscalc(Ptx,d2021);

Prx211 = Prx121;
Prx212 = Prx221;
Prx213 = Prx321;
Prx214 = Prx421;
Prx215 = Prx521;
Prx216 = Prx621;
Prx217 = Prx721;
Prx218 = Prx821;
Prx219 = Prx921;
Prx2110 = Prx1021;
Prx2111 = Prx1121;
```

```
Prx2112 = Prx1221;
Prx2113 = Prx1321;
Prx2114 = Prx1421;
Prx2115 = Prx1521;
Prx2116 = Prx1621;
Prx2117 = Prx1721;
Prx2118 = Prx1821;
Prx2119 = Prx1921;
Prx2120 = Prx2021;

%Calculate Signal to Interference Ratios

%SIR for transmissions from 1 to 2,6,7

SIR2from1 = Prx12/(Prx32+Prx52+Prx112+Prx132+Prx152);
SIR6from1 = Prx16/(Prx36+Prx56+Prx116+Prx136+Prx156);
SIR7from1 = Prx17/(Prx37+Prx57+Prx117+Prx137+Prx157);

%SIR for transmissions from 2 to 3,6,7,8

SIR3from2 = Prx23/(Prx123+Prx143+Prx43);
SIR6from2 = Prx23/(Prx126+Prx146+Prx46);
SIR7from2 = Prx23/(Prx127+Prx147+Prx47);
SIR8from2 = Prx23/(Prx128+Prx148+Prx48);

%SIR for transmissions from 3 to 7,8,9

SIR7from3 = Prx37/(Prx17+Prx57+Prx117+Prx137+Prx157);
SIR8from3 = Prx38/(Prx18+Prx58+Prx118+Prx138+Prx158);
SIR9from3 = Prx39/(Prx19+Prx59+Prx119+Prx139+Prx159);

%SIR for transmissions from 4 to 3,5,8,9,10

SIR3from4 = Prx43/(Prx23+Prx123+Prx143);
SIR5from4 = Prx45/(Prx25+Prx125+Prx145);
SIR8from4 = Prx48/(Prx28+Prx128+Prx148);
SIR9from4 = Prx49/(Prx29+Prx129+Prx149);
SIR10from4 = Prx410/(Prx210+Prx1210+Prx1410);

%SIR for transmissions from 5 to 4,9,10

SIR4from5 = Prx54/(Prx34+Prx14+Prx114+Prx134+Prx154);
SIR9from5 = Prx59/(Prx39+Prx19+Prx119+Prx139+Prx159);
SIR10from5 = Prx510/(Prx310+Prx110+Prx1110+Prx1310+Prx1510);

%SIR for transmissions from 6 to 7,11,12

SIR7from6 = Prx67/(Prx87+Prx107+Prx167+Prx187+Prx207);
SIR11from6 = Prx611/(Prx811+Prx1011+Prx1611+Prx1811+Prx2011);
SIR12from6 = Prx612/(Prx812+Prx1012+Prx1612+Prx1812+Prx2012);
```

```
%SIR for transmissions from 7 to 8,11,12,13

SIR8from7 = Prx87/(Prx98+Prx178+Prx198);
SIR11from7 = Prx811/(Prx118+Prx1711+Prx1911);
SIR12from7 = Prx812/(Prx912+Prx1712+Prx1912);
SIR13from7 = Prx813/(Prx913+Prx1713+Prx1913);

%SIR for transmissions from 8 to 12,13,14

SIR12from8 = Prx812/(Prx612+Prx1012+Prx1612+Prx1812+Prx2012);
SIR13from8 = Prx813/(Prx613+Prx1013+Prx1613+Prx1813+Prx2013);
SIR14from8 = Prx814/(Prx614+Prx1014+Prx1614+Prx1814+Prx2014);

%SIR for transmissions from 9 to 8,13,14,15

SIR8from9 = Prx98/(Prx78+Prx178+Prx198);
SIR13from9 = Prx913/(Prx713+Prx1713+Prx1913);
SIR14from9 = Prx914/(Prx714+Prx1714+Prx1914);
SIR15from9 = Prx915/(Prx715+Prx1715+Prx1915);

%SIR for transmissions from 10 to 9,14,15

SIR9from10 = Prx109/(Prx89+Prx69+Prx169+Prx189+Prx209);
SIR14from10 = Prx1014/(Prx814+Prx614+Prx1614+Prx1814+Prx2014);
SIR15from10 = Prx1015/(Prx815+Prx615+Prx1615+Prx1815+Prx2015);

%SIR for transmissions from 11 to 12,16,17

SIR12from11 = Prx1112/(Prx112+Prx312+Prx512+Prx1312+Prx1512);
SIR16from11 = Prx1116/(Prx116+Prx316+Prx516+Prx1316+Prx1516);
SIR17from11 = Prx1117/(Prx117+Prx317+Prx517+Prx1317+Prx1517);

%SIR for transmissions from 12 to 13,16,17,18

SIR13from12 = Prx1213/(Prx213+Prx413+Prx1413);
SIR16from12 = Prx1216/(Prx216+Prx416+Prx1416);
SIR17from12 = Prx1217/(Prx217+Prx417+Prx1417);
SIR18from12 = Prx1218/(Prx218+Prx418+Prx1418);

%SIR for transmissions from 13 to 17,18,19

SIR17from13 = Prx1317/(Prx117+Prx317+Prx517+Prx1117+Prx1517);
SIR18from13 = Prx1318/(Prx118+Prx318+Prx518+Prx1118+Prx1518);
SIR19from13 = Prx1319/(Prx119+Prx319+Prx519+Prx1119+Prx1519);

%SIR for transmissions from 14 to 13,18,19,20

SIR13from14 = Prx1413/(Prx213+Prx413+Prx1213);
SIR18from14 = Prx1418/(Prx218+Prx418+Prx1218);
SIR19from14 = Prx1419/(Prx219+Prx419+Prx1219);
SIR20from14 = Prx1420/(Prx220+Prx420+Prx1220);
```

```
%SIR for transmissions from 15 to 14,19,20

SIR14from15 = Prx1514/(Prx114+Prx314+Prx514+Prx1114+Prx1314);
SIR19from15 = Prx1519/(Prx119+Prx319+Prx519+Prx1119+Prx1319);
SIR20from15 = Prx1520/(Prx120+Prx320+Prx520+Prx1120+Prx1320);

%SIR for transmissions from 16 to 17,21

SIR17from16 = Prx1617/(Prx617+Prx817+Prx1017+Prx1817+Prx2017);
SIR21from16 = Prx1621/(Prx621+Prx821+Prx1021+Prx1821+Prx2021);

%SIR for transmissions from 17 to 18,21

SIR18from17 = Prx1718/(Prx718+Prx918+Prx1918);
SIR21from17 = Prx1721/(Prx721+Prx921+Prx1921);

%SIR for transmissions from 18 to 21

SIR21from18 = Prx1821/(Prx1621+Prx2021+Prx621+Prx821+Prx1021);

%SIR for transmissions from 19 to 18,21

SIR18from19 = Prx1918/(Prx1718+Prx718+Prx918)
SIR21from19 = Prx1921/(Prx1721+Prx721+Prx921)

%SIR for transmissions from 20 to 19,21

SIR19from20 = Prx2019/(Prx1819+Prx1619+Prx619+Prx819+Prx1019)
SIR21from20 = Prx2021/(Prx1821+Prx1621+Prx621+Prx821+Prx1021)

W = ([SIR2from1 SIR6from1 SIR7from1 SIR6from2 SIR7from2 SIR8from2 ...
     SIR3from2 SIR7from3 SIR8from3 SIR9from3 SIR8from4 SIR9from4 ...
     SIR10from4 SIR3from4 SIR4from5 SIR9from5 SIR10from5 SIR7from6 ...
     SIR11from6 SIR12from6 SIR11from7 SIR12from7 SIR13from7 ...
     SIR8from7 SIR12from8 SIR13from8 SIR14from8 SIR8from9 SIR13from9
...
     SIR14from9 SIR15from9 SIR9from10 SIR14from10 SIR15from10 ...
     SIR16from11 SIR17from11 SIR12from11 SIR16from12 SIR17from12 ...
     SIR18from12 SIR13from12 SIR17from13 SIR18from13 SIR19from13 ...
     SIR18from14 SIR19from14 SIR20from14 SIR13from14 SIR14from15 ...
     SIR19from15 SIR20from15 SIR21from16 SIR17from16 SIR21from17 ...
     SIR18from17 SIR21from18 SIR18from19 SIR21from19 SIR21from20 ...
     SIR19from20 1]).^(-1);

S = [1 1 1 2 2 2 2 3 3 3 4 4 4 4 5 5 5 6 6 6 7 7 7 7 8 8 8 ...
    9 9 9 9 10 10 10 11 11 11 12 12 12 12 13 13 13 14 14 14 14 ...
    15 15 15 16 16 17 17 18 19 19 20 20 21];

D = [2 6 7 6 7 8 3 7 8 9 8 9 10 3 4 9 10 7 11 12 11 12 13 8 12 13 14
...
```

```
    8 13 14 15 9 14 15 16 17 12 16 17 18 13 17 18 19 18 19 20 13 ...
    14 19 20 21 17 21 18 21 18 21 21 19 20];

DG = sparse(S,D,W);

h = view(biograph(DG,[],'ShowWeights','on'))

[dist, path, pred] = graphshortestpath(DG, 1, 21) ;

dist

path
```

LIST OF REFERENCES

[1] I. J. Sliva, "Technologies used in wireless sensor networks," *Proc. of IEEE International Conference on Systems, Signals and Image Processing*, 2008, pp.77–80.

[2] C. Kozierok, *The TCP/IP Guide*, No Starch Press: San Francisco, CA, pp. 148–185, 2005.

[3] L. Yadong, W. Cui and R. Zhang, "Research based on OSI model," *Proc. of IEEE International Conference on Communication Software and Networks (ICCSN)*, 2011, pp. 554–557.

[4] P. Cardieri, "Modeling Interference in Wireless Ad Hoc Networks," *IEEE Communications Surveys & Tutorials,* vol.12, no.4, 2010, pp.551–572.

[5] P. Thulasiraman and X. Shen," Interference Aware Resource Allocation for Hybrid Hierarchical Wireless Networks," *Computer Networks,* vol. 54, no.13, pp. 2271–2280, 2010.

[6] N. Salman, I. Rasool and A.H. Kemp, "Overview of the IEEE 802.15.4 standards family for Low Rate Wireless Personal Area Networks," *Proc. of IEEE International Symposium on Wireless Communication Systems (ISWCS)*, 2010, pp. 701–705.

[7] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, 1997, pp. 1405–1413.

[8] D. Gao and H. Wang, "The research of WSN routing technology based on the gradient and the residual energy ant algorithm," *Proc. of IEEE International Conference on Computer and Automation Engineering (ICCAE)*, 2010, pp. 552–556.

[9] M. Chu, H. Haussecker and F. Zhao, "Scalable Information-Driven Sensor Querying and Routing for ad hoc Heterogeneous Sensor Networks," *The International Journal of High Performance Computing Applications,* vol. 16, no.3, pp.293–313, 2002.

[10] N. Sadagopan, B. Krishnamachari and A. Helmy, "The ACQUIRE mechanism for efficient querying in sensor networks," *Proc. of IEEE International Workshop on Sensor Network Protocols and Applications,* 2003, pp.149–155.

[11] V. Rodoplu and T. H. Meng, "Minimum Energy Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications,* vol. 17, no. 8, pp. 1333–1344, 1999.

[12] Y. Xu, J. Heidemann and D. Estrin, "Geography-informed Energy Conservation for Ad-hoc Routing," in *Proc. of ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 2001, pp. 70–84.

[13] Y. Yu, R. Govindan and D. Estrin, "Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks," UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023, May 2001.

[14] K. Beydoun and V. Felea, "WSN hierarchical routing protocol taxonomy," *Proc. of IEEE International Conference on Telecommunications (ICT),* 2012, pp. 1–6.

[15] W. Heinzelman, A, Chandrakasan and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," *Proc. of the Hawaii International Conference on System Sciences (HICSS)*, 2000, pp. 1–10.

[16] S. Lindsey and C. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems," *Proc. of IEEE Aerospace Conference*, 2002, pp. 1125–1130.

[17] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[18] P. Kyasanur and N. F. Vaidya, "Capacity of Multichannel Wireless Networks Under the Protocol Model," *IEEE/ACM Transactions on Networking,* vol.17, no. 2, pp. 515–527, 2009.

[19] Y. Shi, T. Hou, J. Liu and S. Kompella, "Bridging the Gap between Protocol and Physical Models for Wireless Networks*," IEEE Transactions on Mobile Computing,* vol.12, no.7, pp. 1404–1416, 2013.

[20] Y. Wang, W. Wang, X-Y. Li and W-Z. Song, "Interference-Aware Joint Routing and TDMA Link Scheduling for Static Wireless Networks," *IEEE Transactions on Parallel and Distributed* vol.19, no.12, pp. 1709–1726, 2008.

[21] S. Lam, "Delay Analysis of a Time Division Multiple Access (TDMA) Channel," *IEEE Transactions on Communications,* vol.25, no.12, pp. 1489–1494, 1977.

[22] R. Cooper, "Birth-and-Death Queuing Models," *Introduction to Queuing Theory*, 2nd ed., New York: Oxford, ch. 3, pp. 73–116, 1972.

[23] P. Varbrand and D. Yuan, “Resource Allocation of Spatial Time Division Multiple Access in Multi-hop Radio Networks,” *Resource Management in Wireless Networking*, vol.16, pp. 198–222, 2005.

[24] Z. Guo and C. Yong-guang, “An optimal scheduling algorithm in spatial TDMA mobile ad hoc network,” *Proc. of IEEE International Conference on Microwave Radar and Wireless Communications (MIKON),* 2010, pp. 1–5.

[25] M. Tommiska and S. Jorma, “Dijkstra’s Shortest Path Routing Algorithm in Reconfigurable Hardware,” *Field-Programmable Logic and Applications*, Springer Berlin Heidelberg, pp. 653–657, 2001.

[26] B. Korte and J. Vygen, “Shortest Paths,” *Combinatorial Optimization Theory and Algorithms*, Springer Berlin Heidelberg, Chapter 7, pp. 145–146, 2005.

[27] *MATLAB version 8.0.0 Reference Manual,* The Mathworks Inc., Natick, MA, 2012.

[28] J. C. Lim and K. D. Wong, “Exploring Possibilities for RSS-Adaptive Power Control in MICA2-based Wireless Sensor Networks,” *Proc. IEEE International Conference on Control, Automation, Robotics and Vision,* 2006, pp. 1–6.

[29] *QualNet 6.1 User’s Guide*. Scalable Network Technologies, Los Angeles, CA, 2012.

[30] D. Torrieri, “Fading of Wireless Communications,” *Principles of Spread Spectrum Communications Systems*. Boston MA: Springer US, 2005, pp. 231–292.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California