Stony Brook University

SDN-based WSN Routing Protocol

A Thesis presented

by

Divith Aruni Babu

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Electrical Engineering

Stony Brook University

December 2016

Stony Brook University

The Graduate School

Divith Aruni Babu

We, the thesis committee for the above candidate for the

Master of Science degree, hereby recommend

acceptance of this thesis

K. Wendy Tang - Thesis Advisor
Associate Professor, Department of Electrical and Computer Engineering

Eric Noel - Thesis Co-Advisor
Adjunct Professor, Department of Electrical and Computer Engineering

This thesis is accepted by the Graduate School

Charles Taber
Dean of the Graduate School

Abstract of the Thesis

SDN-based WSN Routing Protocol

by

Divith Aruni Babu

Master of Science

in

Electrical Engineering

Stony Brook University

2016

This thesis aims to find out whether a centralized routing protocol with a dynamic routing table could perform as well as a decentralized routing protocol for Wireless Sensor Networks (WSNs). Towards this goal, a Software-defined Networking (SDN) based protocol is proposed and its performance is measured against a popular decentralized routing protocol, Control Tree Protocol (CTP). The goal is to investigate whether an implementation of SDN in a sensor network would offer WSN applications an advantage by compromising on autonomy to possibly increase efficiency and application diversity. This thesis shows that, although the performance of the SDN-based protocol is inferior than that of a decentralized protocol, it offers more flexibility in terms dynamic variations to node function by updating a routing table as opposed to manually reprogramming a node to perform a different task as in the case of a traditional decentralized routing protocol. The proposed SDN-based implementation is a novel idea to the extent of our knowledge. We did not find any other such implementation in our literature survey.

Contents

List of Figures

List of Tables

Abbreviations

SDN	**S**oftware **D**efined **N**etworking
WSN	**W**ireless **S**ensor **N**etwork
CTP	**C**ontrol **T**ree **P**rotocol

Dedicated to my mother and father

Chapter 1

Introduction

Wireless Sensor Networks (WSNs) gained prominence at the beginning of this century. Small sized, low cost hardware accelerated their growth and enabled them to be used in large networks for a variety of applications [1, 2]. The deployment of large decentralized networks of sensors and actuators has been fundamental to the development and integration of the Internet of Things (IoT) in our day-to-day lives [3]. Further development in WSNs could also have a positive effect on IoT applications[3].

Unlike WSNs, the concept of Software-defined Networking (SDN) can be traced back to the 1980's, [4] but the technology has only recently been deployed on a large scale [4]. Offering a centralized approach to routing internet traffic, SDN provides many advantages over traditional routing where switching units route traffic based on their immediate surroundings.

Naturally, combining the two concepts can be considered the logical next step. This thesis aims to find out whether an implementation of SDN in a sensor network would offer WSN applications an advantage by compromising on autonomy to possibly increase efficiency and application diversity.

1.1 Wireless Sensor Networks

A node is the basic hardware unit in a Wireless Sensor Network. Each node is composed of a processor, a transceiver, memory, sensors and an energy supply. These nodes are deployed in huge numbers to form a large interconnected network.

The primary function of the network is to obtain one or more environmental factors like temperature, pressure, acidity, intensity of light, chemical composition to mention a few. The communication protocol and the sensors that are used depend on the application the network was designed for and the environment they are placed in [5]. Each node in the network acts as an autonomous unit performing the following functions independently:

- sensing its surroundings
- processing the sensed information
- transmitting the processed information
- forwarding transmitted data of other nodes

The messages usually are directed towards and collected by one of the nodes called the sink which acts as a gateway between the sensor network and the user. The sink generally has access to a continuous source of power along with a large memory storage unit to reliably collect all the messages sent by the nodes in the network. The collected data can then be used by the user as described in [6].

1.1.1 Routing

Each node is made autonomous by programming a routing protocol which dictates the behaviour of individual nodes creating a decentralized network. The communication system, rather than the sensing system, is the most power consuming function of a WSN. Therefore, the routing protocol used plays a major role in reducing the power consumed by the network as a whole. There are a variety of routing protocols available and they are selected according to a number of factors. There are various ways of classifying routing protocols and various attempts have been made by Akkaya and Younis [6], Al-Karaki and Kamal [5], Villalba et al. [7], and Singh et al. [8]. Based on these classifications, a combined classification of routing protocols has been made in Chapter 2.

1.1.1.1 Design Issues

The routing protocol closely influences the success of the architecture considered and so routing protocols play an important role in the success of the network.

There are several constraints of WSN's that limit the design of a routing protocol. These issues, as described in [8], are briefly discussed in this section.

- ***Energy Capacity:*** Wireless communication and computations that the nodes perform, have a detrimental effect on the limited energy supplies of the nodes.Replacing energy sources may prove infeasible due to the harsh environments the nodes are deployed in. Hence, efficient routing protocols could help prolong the life of the nodes by reducing consumption of energy.

- ***Autonomy:*** Each node in the network is capable of transmitting, receiving, sensing and performing calculations. Routing protocols are designed to increase this independence. This autonomy allows nodes to self-organize and form adhoc networks easily. Autonomy introduces security issues since any random node could possibly enter the network and have access to network data or also could possibly cause network disruptions.

- ***Scalability:*** Sensor networks have hundreds or even thousands of nodes in the network. Hence, the routing protocol must handle huge amounts of data generated by a large number of nodes.

- ***Resilience:*** Situations where nodes stop working abruptly are common considering the harsh environments these networks are deployed in. Routing protocols must be resilient enough to adapt to the changes in topology.

- ***Data Delivery Models:*** Depending on the application, the number of data samples could change. Data could be continuous, event-driven, query-driven or a hybrid of both. In continuous data delivery, the nodes send updates periodically while in event-driven delivery, nodes send update only when an event occurs. In query-driven data delivery, nodes update their sensed values when a query is sent by the sink while, a hybrid model may be a combination of a couple or all the above methods.

- ***Quality of Service:*** In some applications, the delay from when a node's sensed data reaches the sink is very important. This bounded latency is an important consideration in the case of data delivery. The delay may however be increased to reduce quality but increase the life of the network. The trade-off must be studied and the routing protocol should be implemented based on the effect it has on the application.

- ***Data Aggregation:*** Sensor nodes are tasked with the measurement of ambient environmental features. These normally do not change by much over long periods of time, leading to nodes generating large amounts of redundant data. To reduce redundancy, data aggregation has been used as a solution. Since computation utilizes significantly less power than radio communication, including aggregation could result in optimizing traffic and increasing energy efficiency.

1.2 Software-defined Networking

The concept of a centralized network architecture was first proposed in the 1980s by AT&T. The proposed architecture was called Network Control Point (NCP) [9] but didn't gain importance because its implementation proved to be difficult. The network-wide vantage point the architecture offered while also providing a path for the evolution of the network without having to rely on changing the infrastructure simultaneously offered an insight into the advantages a centralized network could provide. Software-defined Networking provides a practical implementation of the network architecture proposed AT&T's Central Network Control.

The modern implementation of a Software-defined Network (SDN) provides the advantages as AT&T's NCP along with a few others. SDN is implemented in today's networks by separating the whole network into two planes, the Control Plane and the Data Plane, each plane is responsible for different functions. This separation is made possible by using Flow Tables designed by the Control Plane and implemented by the Data Plane. The separation of responsibilities simplifies the routing decisions to be made by each router, essentially making them just forwarders of data rather than making actual routing decisions. Thus, the software running on routers, switches, and other components of the network involved in forwarding of packets make up the Data Plane while the Control Plane is composed of the specific controller tasked with the responsibility of making routing decisions and transmitting Flow Tables to the respective infrastructure components like routers and switches [10].

1.2.1 Separating the Control and Data Plane

As explained previously, the separation of the Data Plane from the Control Plane is the path taken by SDN to develop independent of underlying hardware. In traditional networks, the Data Pane would be responsible for accepting packets, matching packet destination with Flow Table entries, modifying headers, and transmitting the packet through the appropriate port. In SDN [10], this layer is made programmable and thus, it is capable of performing operations on network traffic at a much higher complexity. It may be able to provide services such as

- *access control*
- *mapping header fields*
- *traffic monitoring*
- *buffering*
- *scheduling*
- *deep packet inspection*
- *forwarding of packets*

On the other hand the Control Plane is made responsible for implementing the logic of the network. The Controller at the Control Plane gathers information about switches in the network and produces Flow Tables as the output. The Flow Table is designed based on the unique bird's eye view the Controller has over the whole network.

The introduction of separate Control and Data Planes brings with it its own set of advantages and disadvantages. The advantages [11] will be discussed below.

- ***Unhindered Innovation:*** The decoupling of the Control Plane and the Data Plane allows routing decisions to be made at a much higher level than traditional networks where, routing decisions are made by almost every hardware component like a switch or a router in the infrastructure of the network. Each hardware component released by different companies could have different implementations and might operate differently than other hardware. The independence from such underlying hardware provides network software the

ability to grow and develop at a much higher rate unhindered by hardware performance and compatibility issues.

- ***Bird's Eye View:*** The Control Plane which includes the Controller in SDN operates above the physical network. The Controller accounts for the topology of the whole network to make routing decisions. This is in stark contrast to traditional network where routers and switches make routing decisions based on their immediate neighbours and network characteristics. This would allow the development of more efficient network protocols and also allow researchers to have more information on how a network operates under various conditions.

- ***Flexibility:*** In traditional networks, the routing protocol is developed along with the hardware. Each hardware component has software that specifically designed for that particular piece of hardware. This makes it hard to develop better software since it would have to be designed for each component based on its specific hardware configuration. Also, each component in the network would have to be individually updated with new software. In SDN, since the Data Plane is basically reduced to components mainly tasked with forwarding data and the responsibility of making routing decisions is left to the Control Plane, any new routing protocol could be implemented only by changing the software at the Controller, specifically the Controller. This selection could be made according the many factors and the change would not be as tedious as in traditional networks.

Although SDN has its advantages, as with any physical implementation, it also has its disadvantages. The most important of which are [10]:

- ***Scalability:*** Any centrally controlled system faces the problem of exponentially increasing load on the control unit. This unit will have to be extensively tested and verified. The Controller in SDN has several responsibilities such as monitoring, storing, and processing information like the state and address of every switch, router or any network component. As the network size increases, the load on the Controller to fulfil its responsibilities also increases. This could mean that network size might be limited by the performance of the Controller.

- ***Security:*** Since all routing decisions are made at possibly a single Controller, the failure of the Controller to perform its duties may result in hindering the operation of the network. Single point failures could have a deep impact on the performance of the network. There would have to be reliable backups to the Controller to make the system more reliable.

1.2.2 Match/Action Routing

Match/Action Routing can be discussed using an example. [10] provides a description of one of the implementations of SDN i.e. OpenFlow. According to the paper, OpenFlow switches operate based on Flow Tables designed by the Controller and sent to the switch via a Secure Channel.

In an OpenFlow [10] switch, each entry has the following three fields:

- *Packet Header*, defining flow
- *Action*, defining how the packet should be processed
- *Statistics*, to keep track of basic information such as number of packets, flow size in bytes, and the time elapsed since the flow was matched with a packet

The Flow Table allows the Data Plane to perform certain programmed actions when a packet matches an entry in the table. Based on the number of actions the OpenFlow switch provides, it is classified under different categories. The lowest category, Type 0, would be a switch which supports a 10-tuple header and four basic actions. The four basic actions the switch would provide are:

- *Forward* the matched packets to a particular port or ports
- *Encapsulate* and forward matched packets to the controller. This would be primarily done in case of a new flow
- *Drop* all matched packets. This could be done with network security in mind and in the case of a denial of service (DOS) attack. This could also be useful in order to reduce spurious traffic resulting from discovery broadcasts from end hosts
- *Forward Normally* through the normal processing pipeline of the switch

SDN allows the Data Plane to be programmed and thus, enables network programmers to include more actions or action sets to increase performance or efficiency of the network.

1.3 Problem Description

This thesis is aimed at determining whether incorporating a few fundamental concepts of Software-defined Networking would be beneficial to Wireless Sensor Networks. Nodes in a traditional decentralized Sensor Network enjoy a high degree of autonomy in terms of routing. A study aimed at incorporating a centralized architecture with one controller in charge of making comparatively more complex routing decisions, appears to be an idea worth exploring.

By assigning most nodes in a network to act as forwarding engines as in the SDN Data Plane, and having one or a few nodes perform the functions of a Control Plane Controller, a Sensor Network based on SDN concepts can be created. Such a network could solve a few problems in WSNs. These problems are discussed below.

- ***Resource Underutilization:*** Nodes in Sensor Networks are constrained for memory and processing power. This limits the applications they can be programmed with. As with TinyOS, a WSN operating system, there can only be one application a node can be programmed with. This makes the network rigid, where to change an application each node would have to be reprogrammed with the new application. Using Match/Action flow tables as in SDN could provide the nodes with more flexibility to change how they operate. Reducing most nodes to forwarding engines saves on memory and processing power which can be used to include more actions which the nodes can perform which in turn could help to diversify the applications one node can be used for.

- ***Load Balancing:*** The autonomy that nodes in a Sensor Network enjoys may also limit its effectiveness in finding the best path to the final destination. As the node is independent in making routing decisions, it would base this decision on its immediate one-hop neighbours and surroundings. Comparing this process with applying SDN's routing decisions which are based

on a bird's eye view of the network as a whole has a greater possibility of distributing load uniformly across the whole network. This would result in conserving power and extending battery life.

- ***Power Consumption:*** As the Controller would be making routing decisions taking into account the whole network's topology and the status of each node, the flow tables which are created may lead to more efficient network traffic. A more informed Controller would make a more efficient network.

1.4 Thesis Organization

The first chapter gives an introduction to WSNs and SDNs along with a brief explanation of the problem this thesis aims to tackle. Chapter 2 describes routing protocols currently in use for sensor networks. Each routing protocol is classified into different categories based on various factors. Around 60 routing protocols are classified in this chapter.

Chapter 3 provides a detailed explanation of the implementation of the SDN-based routing protocol while Chapter 4 provides simulation results obtained from the implementation. This chapter also compares the simulation results with a well known traditional protocol called Control Tree Protocol [12]. The last chapter provides the conclusion and offers some areas for future work.

Chapter 2

Background and Related Work

In existing implementations, inclusion of all the neighbours along with all the nodes within a given area is proposed by [13]. In [14], a method to create a number of routes and provide routing tables which include one entry for each route the corresponding node is a part of is proposed. To reduce the length of routing tables, we could send a routing table having entries for immediate neighbours only as proposed by [15]. The sink would have to create an accurate topology map of the system and determine routes based on the information sent by the nodes. It will then have to reduce this map into small routing tables consisting of entries of the neighbours of each node. This routing table would have to be sent to each node. Thus, the routing table size would be reduced to only the number of neighbours of the nodes in the network.

The routing table designed by the sink would not have access to the state of each node in the network. So, nodes would have to send information to the sink about their characteristics like battery level, link rate, replenishment rate if they have a renewable source of power among other values. To determine the most accurate routes, the sink needs as much information as possible while, to reduce energy consumed by the nodes, the nodes will have to reduce the information they send to the sink. This represents a trade off between an accurate topology map and energy consumption. While determining routes based on a list of neighbours might be the least energy consuming option as in [14], it might not lead to accurate routes leading to inefficiency in the network. Inclusion of more information about the node might be needed. There are a number of metrics used to determine routes in the survey provided in table 2.1 which includes

- [16] - Minimum Energy and Maximum Capacity
- [17] - Lifetime and QOS
- [18] - Hop Count
- [13] - Link Quality and Node Quality
- [15] - Velocity, PRR and Remaining Power

This thesis includes link rates, energy levels of the nodes, PRR (Packet Reception Rate), and replenishment rate to determine more accurate routes.

In the survey conducted listed in table 2.1, 5 of the 9 protocol proposals had simulation results and 1 was in TOSSIM. The nodes in the simulations were distributed in a grid. Only 1 simulation had a higher node count of 100 in the network. A simulation like this, in TOSSIM, would demand a huge amount of resources, especially RAM, and would not be convenient. The simulation in this thesis will be based on a 49 node network distributed in a grid.

TABLE 2.1: Survey of routing protocols using routing tables

Paper	Year	Sink	Simulation	Implementation	Performance Evaluation	Match/ Action Routing	Routing Table Length
[16]	2000	Yes	No	No	No	No	log(n), n -No. of organised groups
[17]	1999	Yes	No	Yes	Startup Time; Metric; Network Capacity	No	Not specified
[19]	2009	No	TOSSIM	No	Delivery Rate; Size of Routing Table; Long Links vs No. of Hops	No	Neighbours + Long link paths
[18]	2009	No	Matlab	No	Error of DV-Hop vs R (Range)	No	Neighbours + Received message destinations
[13]	2010	Yes	Parsec	No	Energy Usage; Packet Delivery; Detection of Malicious Node	No	Neighbours + Addresses within a limit
[20]	2012	Yes	No	Yes but, no detailed explanation provided	Energy Saved; Energy Efficiency; Energy Dissipated	Forwards based on destination	Number of clusters
[21]	2008	Yes	NS2	No	End-to-end Delay; Remaining Energy; Lifetime; First Death	No	Number of Child Nodes limited by Level Number
[15]	2009	Yes	NS2	No 12	Throughput and Dropping Traffic	Priority assigned by sink	Neighbours
[14]	2002	Yes	No	No	Protocol	No	One entry for each

TABLE 2.1: Survey of routing protocols using routing tables(*continued*)

Paper	Multihop	Multipath	Base Station	Routing Metrics	Application	Number of Nodes Used
[16]	Yes	No	Router Nodes	Minimum energy; Maximum capacity	Immobile Self-configurable networks	N/A
[17]	Yes	Yes	Yes	Lifetime and QOS	General routing protocol discussion	37 randomly distributed nodes
[19]	Yes	A few paths are chosen before-hand	No	Landmark and Geographic routing	General scalable geographic routing	1000 randomly distributed nodes
[18]	Yes	No	Routers	Hop count	General discussion	100 in 50*50 grid
[13]	Yes	Yes	Cluster Head	NHDF (Next Hop Determination Factor) = Weighted Link Quality and Node Quality	Heterogeneous networks in urban environments	100
[20]	Yes	Yes	Cluster Head	Divided sectors	Cellular WSN integrated network	6 cluster head with unknown normal nodes
[21]	Yes	No	Parent Nodes	Graph relationship; Energy	Mine safety	30 node in 1500*1500 grid
[15]	Yes	Yes	No	Velocity; PRR: Remaining power	Routing based on ant colony	25 nodes in 50 * 50 sq.m
[14]	Yes	Yes	No	Neighbors	Intrusion detection and network safety	100 in 1700 * 1700 sq.m

This thesis proposes that the sink would have the capability to change the routing protocol employed by the network even after the nodes have been deployed. Therefore, a survey of routing protocols used in WSNs has been conducted and the results tabulated in table 2.2. This is to determine a few routing protocols to compare and implement. The survey [5–8] classifies routing protocol based on a number of factors

- Query Based
- Hierarchical
- Location Based
- QoS
- Multipath
- Network Flow
- Data Aggregation
- Energy Efficiency
- Mobility
- Heterogeneity
- Flat Routing
- Negotiation Based
- Coherent Based

• ***Query Based:*** These are protocols where a destination node would have to send a query to a sensing node, to obtain information it requires. For example, if the destination node wanted to know if a proximity sensing node detected an intruder in the area it would need to request the sensing node to transmit that information. There would be no communication if the destination node does not require any information. An example of such a protocol would be [22], Directed Diffusion and [23], Rumor Routing.

• ***Hierarchical:*** These types of protocols generally organize the network into cluster so as to decrease the load on the gateway node. A single-tier network would

be overly dependant on the gateway which would act as the sole link between the user and the nodes. This greatly inhibits scalability while also increasing the chances of a single point failure by overloading the gateway. [24], LEACH, is one of the most popular protocols based on hierarchy.

• ***Location Based:*** Location awareness is important for WSN nodes as the information sensed and transmitted by these nodes are normally related to their immediate surroundings. As a consequence of the importance of location, there have been protocols based on the location of the nodes. [25–28], MECN (Minimum Energy Communication Network) and SMECN (Small Minimum Energy Communication Network), GAF (Geographic Adaptive Fidelity) ,and GEAR (Geographic and Energy Aware Routing) are some example of Location based routing protocols.

• ***QoS:*** The main trade off in WSNs is the one between accuracy of the system and energy conservation. The accuracy of the system translates to the amount of information being transmitted. By decreasing the number of transmissions, we can conserve energy to the nodes, thereby increasing lifetime of the network. The protocols that aim to balance this could be classified as QoS based protocols. [29], SAR (Sequential Assignment Routing), can be considered as a good example of QoS based routing.

• ***Multipath:*** In WSNs, the network topology keeps changing because nodes are usually following sleep cycles and some nodes just run out of energy. So, using multiple paths to transmit data becomes essential for most applications. Using multipath communication also increases resilience due to the fact that a node can use multiple other routes to reach its destination. Protocols mentioned above like SAR, and Directed Diffusion also employ Multipath communication.

• ***Network Flow:*** The network itself could be used to determine the routing protocol to be used. The traffic flow in the network and the state of each component in teh network could be used as factors to determine routes. [30], Maximum Lifetime Energy Routing, proposes to increase network lifetime by transmitting messages based on the link quality between nodes.

• ***Data Aggregation:*** Nodes in WSN usually respond to changes in its immediate environment. As the environment around us doesn't show significant changes in short periods of time, the nodes sensing this information tend to send redundant

data. This increases the number of transmission and therefore decreases the lifetime of the network. A number of the protocols use data aggregation to minimize the information transmitted by nodes and a few can be found in table 2.1.

- ***Energy Efficiency:*** These protocols aim to make the network as energy efficient as possible by choosing routes that maximize the lifetime of the network. Routes with node of high resource levels are preferred to other routes.

- ***Mobility:*** Mobility adds to the increase in complexity of WSN. A few protocols include mobility in their implementation like [31–35]

TABLE 2.2: Classification of routing protocols based on various characteristics

Protocol	Query Based	Hierarchical	Location Based	QoS	Multipath	Network Flow	Data Aggregation	Energy Efficiency	Mobility	Heterogeneity	Flat Routing	Negotiation Based	Coherent Based
[36]	✓						✓				✓	✓	
[22]	✓				✓		✓				✓		✓
[23]	✓		✓				✓				✓		
[37]	✓				✓								
[38]	✓						✓				✓		
[39]	✓									✓	✓		
[40]	✓						✓				✓		
[41]	✓												
[24]		✓				✓					✓		
[42]		✓					✓	✓					
[43]	✓	✓					✓	✓					
[44]	✓	✓					✓						
[45]		✓		✓			✓	✓					
[46]		✓	✓										
[25]		✓					✓						
[26]		✓	✓										
[27]		✓	✓										
[28]		✓	✓					✓					
[30]			✓										
[47]		✓			✓	✓							
[48]			✓			✓							
[29]		✓		✓									
[49]			✓	✓	✓								✓
[50]				✓									
[51]		✓											
[52]		✓		✓	✓			✓					
[53]					✓			✓					
[54]					✓			✓					
[55]			✓		✓			✓					
[56]			✓										✓
[57]			✓										
[58]			✓										
[59]			✓										
[60]	✓												
[61]	✓												
[62]	✓												
[63]	✓												
[64]	✓												
[65]	✓												
[31]		✓											
[32]									✓				
[33]									✓				
[34]									✓				
[35]									✓				
[66]									✓				
[67]					✓								
[68]					✓								
[69]					✓								
[70]										✓			
[71]										✓			
[72]											✓	✓	
[73]		✓			✓								
[74]		✓											
[75]					✓								
[76]		✓											
[77]		✓											
[78]		✓											
[79]			✓										
[80]			✓										
[81]			✓										
[82]			✓										
[83]		✓											
[84]	✓												

Chapter 3

Implementation

This chapter discusses the implementation of the network and the operating system on which the network was designed on. A flowchart of the algorithm used is also provided along with a detailed explanation of each component of the system. The system was implemented on TinyOS, an operating system designed specifically for Wireless Sensor Networks. TinyOS also provides an environment, TOSSIM, for simulation which was used to determine the performance of the system.

3.1 Implementation

When the concept of using the principles of SDN in WSN routing is looked at from a fundamental level, the implementation becomes rather simple. The functions of traditional WSN nodes which involve sensing, route calculation and forwarding would need to be reduced to just sensing and forwarding messages. The sink, acting as the controller, would need to calculate the most efficient route for each node to follow and inform each node about the route. The node would receive this information and forward the sensed data based on the route created by the sink.

For the sink to create routes, it would need to have accurate topology information of the network. To obtain a topology map, the sink would need the position of each node in the network and the relative position of each node with respect to other nodes in the network along with the state of each node. Since it is normally possible for a packet to take a number of routes, each node in the network might be able to determine a good amount of knowledge about the network by analysing

packets received by the nodes in the network. Transferring routing information could pose a serious issue in terms of the amount of information that could be exchanged between the sink and each node in the network if each node sent all the data it had about the network. Although this would help create the most accurate state of the network, it might not be the most efficient considering energy limitations of the WSN. One possible solution might be to limit the amount of information each node will have to send.

To reduce the amount of data to be transferred by the nodes, rather than providing all the information collected by each node, updating the sink about only the immediate neighbours of each node should give the sink most of the information necessary to generate a topology map. After collecting neighbour information sent by each node and generating the best route to be followed by each node, the sink would need to distribute the optimal route to each node. The route taken might involve multiple hops to reach the sink. We could assume that sending the best route to each node would result in routing tables whose length are proportional to the distance the node is from the sink. Sending a complete route could also pose an implementation problem as the data needed to be sent could become huge.

3.2 About TinyOS

TinyOS is an operating system designed for embedded systems with limited computational power and memory[85]. It is an open source platform which uses nesC, a C dialect to minimize resource consumption. This operating system is specifically designed to run on small, computationally restricted processors and power conservation plays an important role in increasing the longevity of the network.

The hardware and software of these embedded devices are designed with energy conservation as the main goal. A computational powerful 64-bit CPU capable of handling RAM of several gigabytes is replaced by either a 8 or a 16 bit microcontroller with RAM of a few kilobytes. A radio with a 802.11 protocol that has a transmission rate of megabytes is replaced with a low power radio that can transmit at a rate of tens to around hundreds of kilobytes a second.

TinyOS uses various mechanisms to aggressively conserve power while also providing abstractions such as timers, communication, storage and sensing. TinyOS

also can run on a number of different generic platforms at the same time TinyOS' structure allows us to easily port to other new platforms.

TinyOS is written in nesC, a C dialect based language. nesC allows for the operating system to reduce RAM usage, minimize code size, help with optimization, and help with detecting and preventing bugs like race conditions.

TinyOS provides a component model. This allows users to write pieces of code that can be reused. This allows the user to write components which can be made independent of underlying hardware on different platforms.

The execution model of TinyOS follows a concurrent model where various components can run at the same time while using the least amount of RAM. While traditional systems use a block until completion method for I/O calls, TinyOS uses split-phase completion. For split-phase calls, although a request immediately returns, the call function gets a callback after the completion of the I/O.

TinyOS also provides APIs for various functions such as reading sensor data, transmitting packets and also reacting to events. The Hardware Abstraction Architecture (HAA) defined in TinyOS provides a way to build components from low-level hardware components to hardware-independent high-level abstractions which allows us to develop new components and have cope with new hardware upgrades.

3.3 Node Functions

The nodes of the network can be classified into two types based on their functionality, the central data collection node called the Sink and network of smaller nodes tasked with the job of sensing the environment.

3.3.1 Sink

The Sink, in traditional WSN networks, acts only as the collector of the data processed and transmitted by the other smaller nodes in the network.

In the proposed architecture, the Sink also is used to be the one that collects the topology of the whole network and decides the routing path each node would have to take.

3.3.2 Sensor Node

The Sensor Nodes are the ones which actually sense changes in environmental factors, processes the sensed information and then transmits the processed information to the Sink.

3.4 Pseudocode

3.4.1 Neighbor Discovery Phase

- As soon as a node boots up, it broadcasts an REQ message to all its neighbors
- Upon reception of an REQ message, the node receiving the REQ message forwards an REP message back to the original node
- An REP message consists of the Node ID and the geometric location of the node
- When an REP message is received by a node, the node extracts all the relevant information from the message about its neighbor
- This information is added to a list of neighbor nodes
- A timer is used to determine the amount of time a mode spends in its neighbor discovery phase. When the timer counts down to zero, the node forwards the neighbor list towards the Sink. The timer period can be changed according to the amount of time needed by a node to collect information on as many neighbors as possible
- If a node other than the Sink receives the neighbor list, it simply forwards the list to the next closest node to the Sink

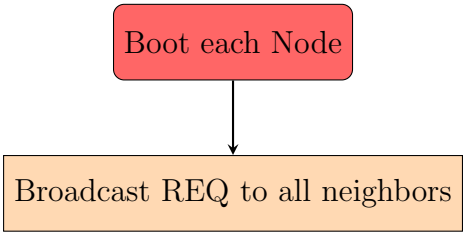


FIGURE 3.1: Each node broadcasts an REQ after initial bootup

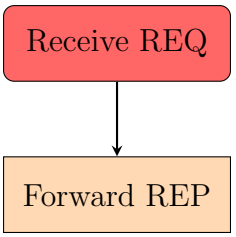


FIGURE 3.2: A node sends back an REP after receiving an REQ from a neighbor

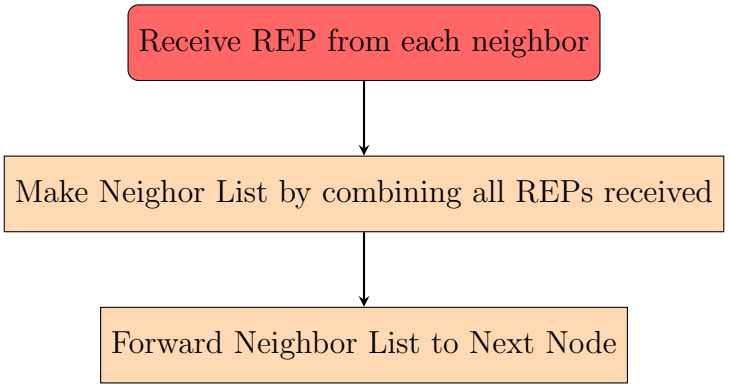


FIGURE 3.3: Each node Forwards its Neighbor List to the next hop with the Sink as the final destination

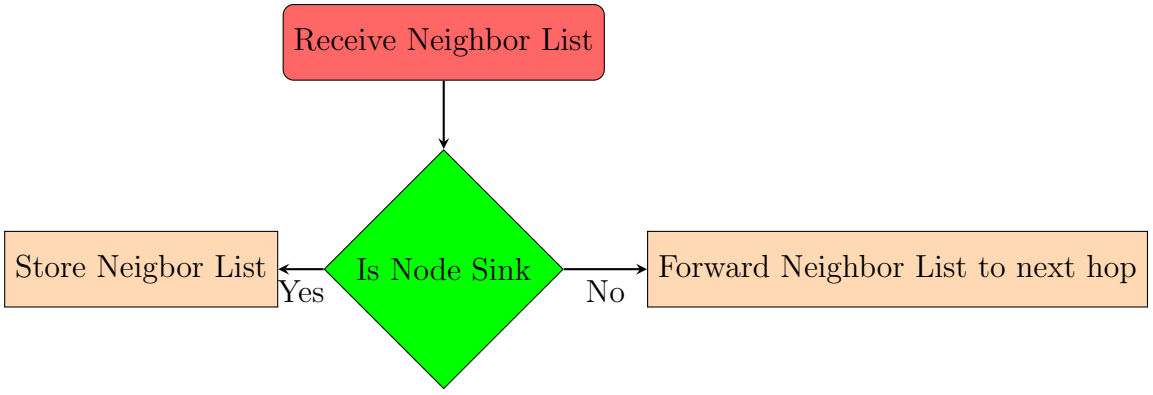


FIGURE 3.4: The neighbor list is forwarded until it reaches the sink

- When the neighbor list reaches the Sink, the Sink extracts all the information about the neighbors of that node
- When the Sink receives the neighbor lists of all the nodes in the network, it can then use this information to make a routing table for each of the nodes
- The routing tables are made using not just the information of one particular neighbor list but using the positions of all the nodes in the network to increase efficiency of communication
- The routing table could also be used to introduce new actions to perform
- These actions may be universal to all the nodes in the network, or may just modify the information sent out from one particular node
- When the Sink computes the routing table for each node, it broadcasts each routing table to each node
- Upon reception of the routing table, the nodes in the network know that the neighbor discovery phase is over and the routing phase is about to begin

3.4.2 Routing Phase

- The Routing Phase is initialized by the reception of the routing table from the Sink
- When a node receives the routing table, it stops neighbor discovery
- It then packs the sensed message in a packet
- The routing table defines the next hop for the node
- It then forwards the packet according to the next hop defined by the routing table
- After forwarding its sensed message, the node radio goes into reception mode and waits for one of its neighbors to forward packets to it
- When a node receives a packet during the Routing Phase, there are four types of actions it may take
- These actions are defined in the routing table by the Sink

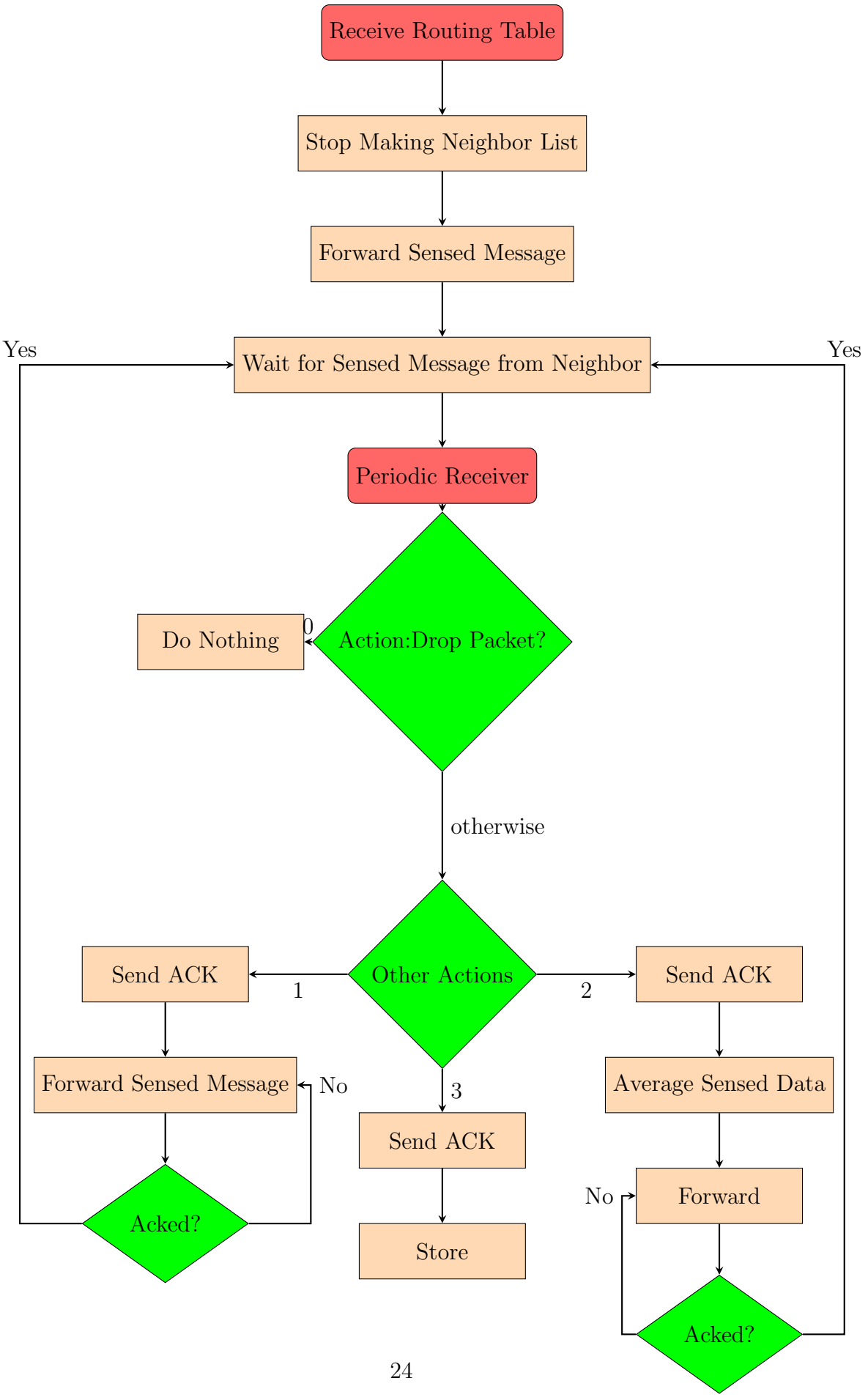


FIGURE 3.5: Routing Phase

- The routing table defines the action a node must take for each of its neighbors
- The four action defined by the Sink are
  - Action 0: Do Nothing - This action is a basic replacement for dropping a packet not intended for that particular node.
  - Action 1: Forward - This action receives a packet extracts the information inside the packet, packs the extracted information in another packet and transmits that packet to the next hop defined by its routing table
  - Action 2: Sum - This action build on the Forward Action. When a node receives a packet and is instructed to perform Action 2, the node extracts the information within the packet, determines the sensed message of the previous node, sums up the extracted sensed message and adds its own sensed message. The sum is then packed into another message and using the routing table, the message is transmitted to the next hop.
  - Action 3: Store - This action is almost exclusively used by the Sink. Upon reception of a packet, the Sink is instructed to perform Action 3. The Sink extracts the information from the packet, determines the sensed value and the corresponding node ID and saves the information to be evaluated by the user later.
- These actions are introduced to perform all basic requirements of a sensor network.
- More complex actions could be introduced to increase efficiency of the network or perform more complex operations using the routing table

Chapter 4

Simulation and Results

This chapter discusses the simulation of the proposed SDN-based network and its performance against a well established network routing protocol called CTP (Collection Tree Protocol) [12].

4.1 Network Variables

Network performance varies as the network changes in its structure or as the environment of the network changes. This section provides an explanation on how these parameters vary and the impact they can have on the performance of the network.

A number of network parameters have been changed to measure network performance over a range of environments and architectures. The following section provides an explanation about these parameters.

4.1.1 Network Topology

Three types of topology were used to test this network. They can be generally classified as a grid based topology or as a tree based topology. The grid based topology has 49 nodes whereas the tree based ones have 7 or 10 nodes based on which one is selected.

CTP does not take into account exact geographical location of each node but, relies on link quality between neighboring nodes to determine routes. Since link quality depends on geographical location, CTP indirectly depends on node location during our simulation. The proposed SDN-based network however, specifically requires node location to determine the next hop and the best path for a packet to reach the sink.

4.1.1.1 7 Node Test Topology

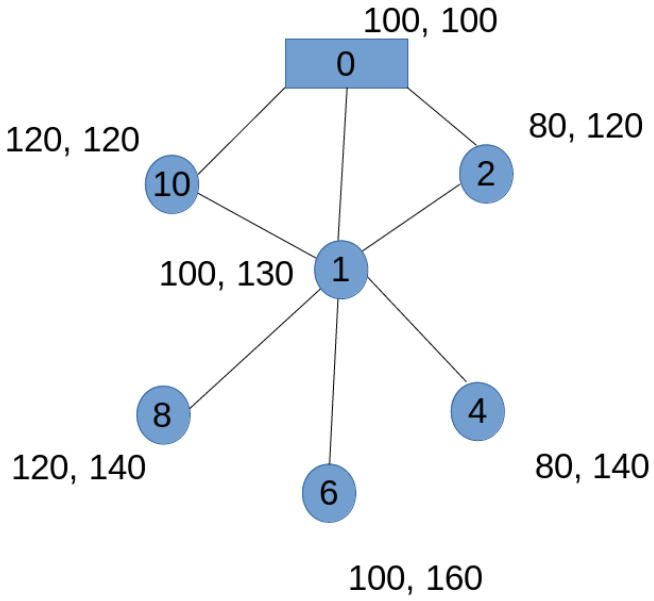


FIGURE 4.1: 7 Node Topology along with the coordinates for each node

Figure 4.1, shows the topology of the 7 node test network. This is a simple network which was designed primarily to test whether the algorithm works as expected. This topology also proved useful to debug erroneous code. This topology follows a tree based approach.

Here, node 1 connects all other nodes in the network while nodes 10 and 2 are the only nodes that connect with 1 and the sink, node 0. Nodes 8, 6, and 4 transmit packets to the sink through node 1.

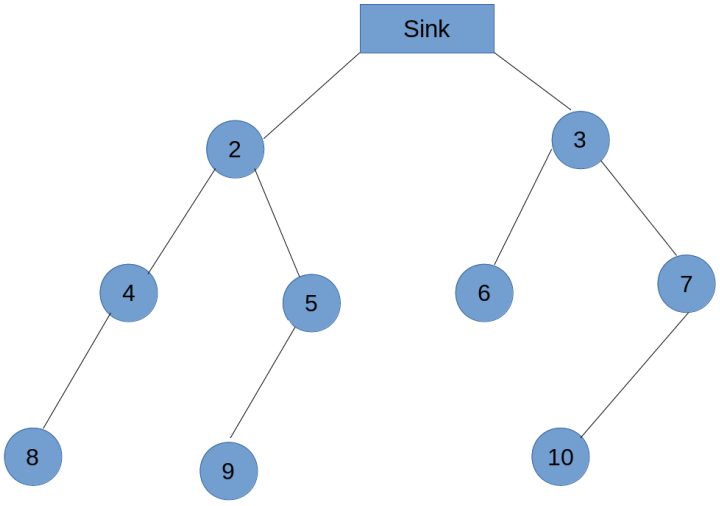


FIGURE 4.2: 10 Node Topology

4.1.1.2 10 Node Topology

The 10 node topology, in Figure 4.2 was used to compare the performance of the network with a well known routing algorithm called Collection Tree Protocol (CTP). This topology was used to measure the performance of CTP by [12] and so, this topology was used to compare the performance of the network with CTP. This topology also follows a tree based approach, identical to our 10 node test network.

In this network, the sink forms the root of the tree and node 2 and 3 form the first layer of nodes above the root. The leaf nodes 8, 9, and 10 transmit to the first layer of the tree through the second layer composed of nodes 4, 5, 6, and 7.

4.1.1.3 49 Node Topology

This topology has 49 nodes arranged in a square grid with 7 rows and 7 columns as shown in Figure 4.3. The sink is the node at the center. The nodes are arranged close together to test network performance in a crowded topology. Most of the results described in the following sections describes network performance for this particular topology.

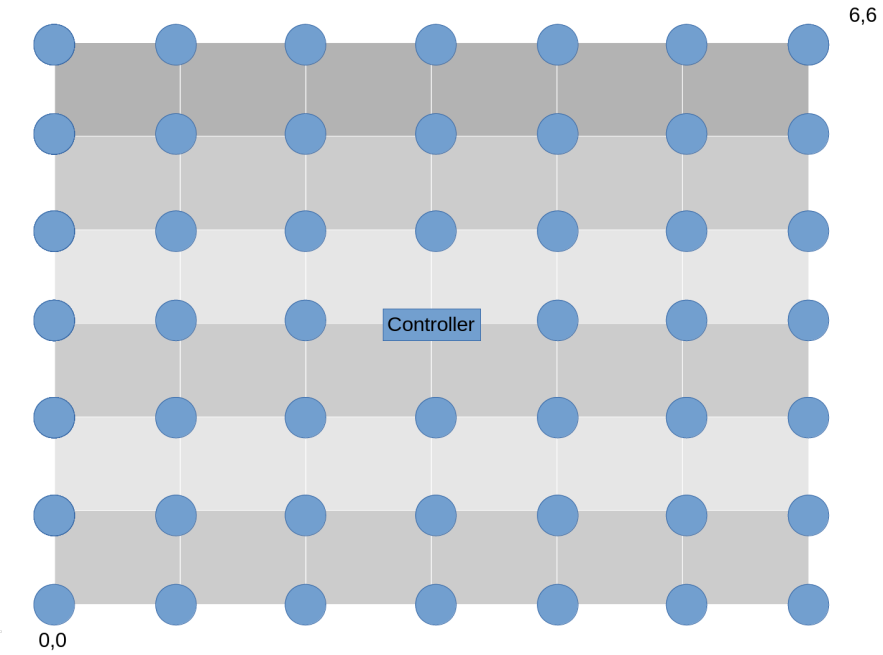


FIGURE 4.3: 49 Node Topology

Here, the nodes are arranged in a grid from coordinates (0,0) to (6,6). The four nodes at the corners connect to two other nodes next to them. Nodes on the outer boundary and not occupying the corners, connect to 3 other nodes while the ones inside connect to 4 neighboring nodes.

4.1.2 Sensing Interval

Sensing interval is the period between two consecutive sensing events where a node uses its sensor to capture data from its environment. Since the network designed in this thesis immediately transmits sensed information to the next hop, we can consider the sensing interval as the transmission interval also. But, practically, sensor node wait to transmit data so as to preserve as much power as possible at the same time decreasing the bandwidth it uses.

As sensing interval decreases the number of transmissions increases and so the amount of time the radio of the node is active increases. This in turn increases power consumption. Since communication and not sensing is the bigger drain on battery life, care is taken to increase the sensing interval. However, increasing sensing interval decreases the reliability of the data obtained from the network.

Thus, the trade-off between a low and a high sensing interval is decided based on the application and the resources available to the user.

A few sensing intervals are used to test this network but, all the nodes have the same period. This value may range from 2 seconds to a few thousand seconds.

4.1.3 Noise

For simulations in TOSSIM, a few noise traces are provided to simulate noise variations in the real world. The one used to test this network is the noise trace obtained from the Meyer library from Stanford [86]. This trace has a noise floor of about -98 dBm with interference spikes around -86 dBm to -87 dBm.

4.1.4 Message Types

The messages used by the network can be classified into two main categories based on which part of the algorithm the message is associated with. They can either belong to the Neighbor Discovery Phase or the Sensing Phase.

The Neighbor Discovery Phase has 4 types of messages varying between 2 bytes and can be upto 128 bytes.

- NeighborREQ : 2 bytes
- NeighorREP : 6 bytes
- NeighborList : 9 bytes
- NetworkMsg : 7 bytes upto 128 bytes

The Routing Phase has two main message types ranging from 10 bytes to 3 bytes. The message that contains sensed data is 10 bytes and the acknowledgement message is 3 bytes long.

4.2 Comparison with CTP

The Collection Tree Protocol, as the name suggests treats the network as a tree with the sink at the root and the nodes at the perimeter of the network as the leafs of a tree. This protocol is used for collection of data from a network by assigning one or a few nodes as a central node in charge of collecting information produced by all the other nodes in the network. Thus the root node acts as a gateway between the network and the user.

The CTP protocol implemented on TinyOS assumes that a few services provided by the data link layer like local addresses, acknowledgements, and source and destination fields are efficient.

CTP is implemented in TinyOS and is available, ready to use, for anyone installing the software [87]. There seems to be a software limitation to this implementation. TOSSIM simulation for CTP is designed to work only for Micaz motes [88] since TOSSIM simulates a radio stack that is identical to Micaz motes [89] and it does not simulate other radios yet. This limitation however does not seem to apply to actual hardware implementations since TinyOS was designed to support a number of hardware configurations [90].

CTP uses a metric called Expected Transmission Count (ETX) to determine the route a packet takes to reach the sink. ETX can be defined as the number of transmissions a node has to make in order for it to send a unicast packet which is acknowledged by the receiver. A node with an ETX of 'n' can be assumed to be able to transit a message to the sink after 'n' transmissions. ETX(sink) will be 0 since, the last node that receives data is the sink, it would not need to transmit data. ETX of any other node would be the ETX(parent) + ETX(that node to its parent). Each node keeps a list of its neighbors and their respective ETX values and the ETX of the link connecting the two. Thus by choosing the lowest ETX among their neighbors, nodes can communicate efficiently.

CTP has three main modules. These are described below:

- Routing Engine: This module helps maintain and update a routing table for each node consisting of its respective neighbors. The engine sends beacon messages to help the Link Estimator calculate link quality using the metric,

ETX. When data is being transmitted, the Routing Engine uses the data messages instead of the beacons to save bandwidth.

- Forwarding Engine: This module is responsible for transmission of data produced by the application layer. The packet produced may be from the same node or from another node in the network. This module also takes care of duplicate packets as well as repairing loops.
- Link Estimator: The link estimator determines the 1-hop link between neighbors. By collection information from beacon messages as well as data messages, the link estimator statistically determines the number of beacons a node must send to successfully transmit one.

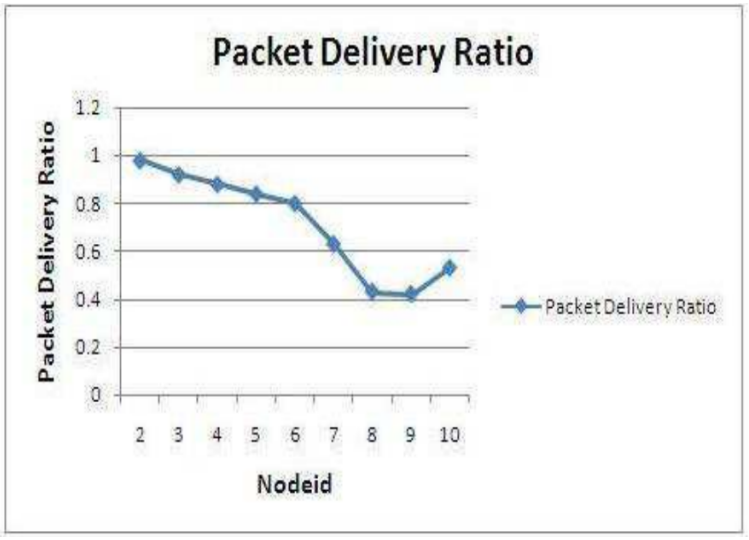
The following section attempts to recreate simulation results of [12]. This is done to make sure the comparison of CTP with the SDN-based network is valid.

4.2.1 CTP Network Verification

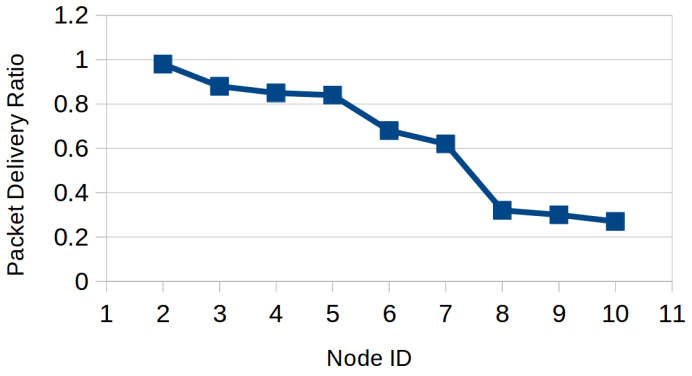
In [12] the performance of CTP on a 10 node network like the one in Figure 4.2 is described. The implementation is carried out again in TOSSIM to try to recreate the results published in the paper. The 10 nodes, which includes a root node that only collects data, have a counter that sends a integer count to the sink every 100 million ticks of a periodic counter. After the periodic counter counts down to 0, it packs the value in a packet, transmits it to the sink and then, increments the integer count.

1 tick of the timer is equivalent to 1 milli second. The Simulation is allowed to run for 5 billion ticks and around 600 messages sent by the nodes are received by the sink making the average number of messages for each node to be around 70 messages.

The paper uses a metric called Packet Delivery Ratio (PDR). Packet Delivery Ratio is defined by the paper as the ratio between the number of received packets and the number of packet transmitted by a node. The number of received packet is assumed to be the difference between the last integer count received by the sink for that particular node and the first count sent by the node. Figure 4.4 is taken from the paper and compared with the our implementation in Figure 4.4. Comparing



(a) Packet Delivery Ratio of CTP from [12]



(b) Packet Delivery Ratio of CTP from our implementation

FIGURE 4.4: Comparison of CTP simulation taken from [12] and our CTP simulation

the two plots, it can be concluded that the our implementation of CTP is similar to the implementation by [12]. There are a few varations which can be attributed to parameters that have not been mentioned in the paper like the noise used.

Power consumption as a metric is not considered in this analysis. This is because, TOSSIM does not provide a reliable way of calculating power consumption. TinyOS 1.x had a tool, PowerTOSSIM which provided power consumption estimates but, it had not been included for the TinyOS 2.x release. Hence, to calculate power, we would have to measure the time between when the radio is active and determine the power assuming energy consumed by the radio would be the same as the one mentioned in the data sheet. This calculation doesn't account for the power consumed due to computation. TOSSIM assumes there is no time

between two successive computation making it impossible to accurately determine the power consumed by a mote when performing computation.

Although CTP doesn't directly use location to determine ETX, it does use link quality to determine ETX. Since link quality is dependant on the distance between nodes, CTP indirectly depends on the location of nodes. This is true for a network deployed in the real world but in TOSSIM, since link quality is provided as an input to the network, the location of each node is not significant for simulation.

4.2.2 Performance Comparison

To compare the performance of the SDN-based network with CTP, another metric, Packet Loss Ratio (PLR), is used. This is because, in the implementation of CTP and the SDN-based network, the number of repeated or duplicate packets was not taken into account by the Packet Delivery Ratio metric used in [12]. This could be considered a significant problem since the number of messages in the network directly affects the efficiency of the network. The forwarding engine of CTP is tasked with suppressing duplicate packets but, in our simulations we found a surprising number of duplicate packets reaching the sink.

As the number of messages transmitted in the network increases, the number of collisions also increases, which can increase the number of retransmission. This vicious cycle would be detrimental to network performance and would lead to inefficient use of power unless the MAC protocol used, is good enough to significantly bring down the number of duplicate messages.

The number of messages received by the sink can also vary significantly because of multihop routing. The node that produces data will not likely be the only node to forward that particular data.

PLR can be defined as the ratio of the total number of messages received by the sink for each count value to the total number of messages transmitted by the node for that particular count value. This value is averaged over all the nodes in the network. PLR for the network is also calculated for a range of antenna gain from -75dB to -20 dB as shown in Figure 4.5. This test was conducted with the 10-node network used by [12].

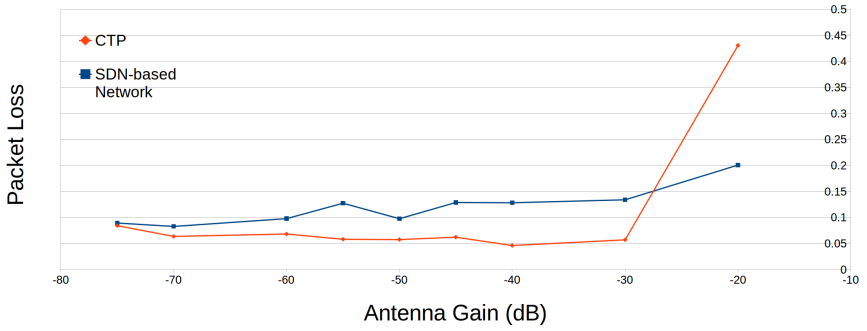


FIGURE 4.5: Packet Loss Ratio for CTP and SDN-based network

We can see that CTP outperforms the SDN-based network by about 2x. Figure 4.6 gives the reason behind the increased loss for the SDN-based network. The number of repeated messages is almost double for the SDN-based network compared to the CTP protocol. The performance deficit can be attributed to the increased overhead caused by maintaining an overview of the entire network by the sink. On the other hand, since CTP is a decentralized protocol, the overhead would be comparatively less.

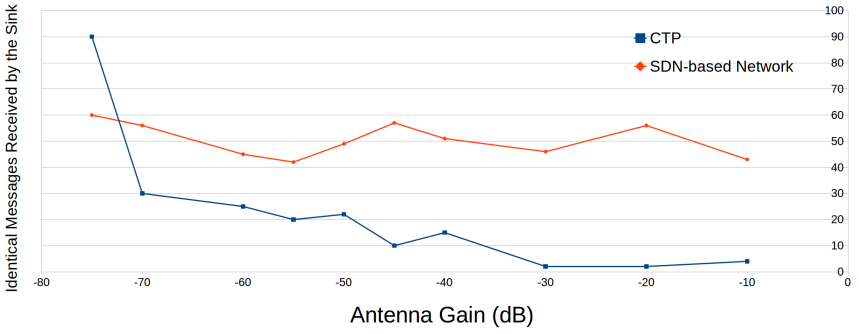


FIGURE 4.6: Number of Duplicate/Repeated messages in CTP(top) SDN-based network(bottom)

4.3 Network Performance

This section deals with measuring network performance of the SDN-based network. The main metric used is the percentage of successfully acknowledged packets sent by the network as a whole. The antenna gain is also varied from +3 dB to beyond -90 dB. The noise trace used has a noise floor of around -90 dB. The topologies used for this test are the 7 node network along with the 49 node network.

4.3.1 Neighbor Discovery Phase Performance

The Neighbor Discovery phase consists of identifying all the neighbors of each node and forwarding that list to the sink. For a 7 node network, the minimum number of messages to be received is 45. This trend is followed in Figure 4.7 . Figure 4.7 plots 20 runs of the same 7 node network, with the same noise trace (noise floor of around -90 dB) and the same antenna gain of -75 dB for all nodes. Although none of the parameters are changed the number of sent messages changes from one run to another.

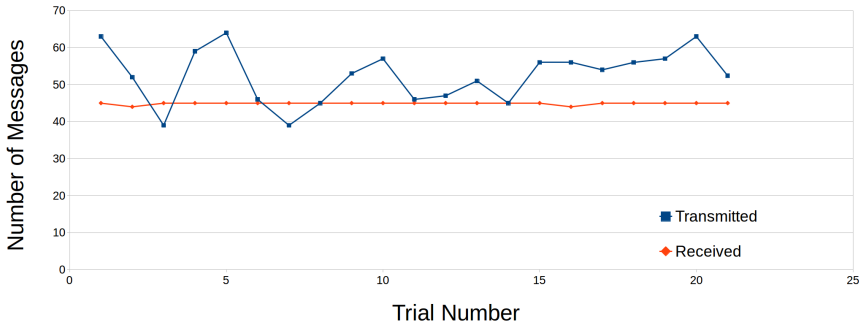


FIGURE 4.7: Variation of the number of messages sent and received over 20 trial runs

This variation can be attributed to the way TOSSIM decides whether a packet is received by a node or not. As the antenna strength approaches the noise floor, Packet Reception Ratio (PRR) tends to zero and when the strength moves away from the noise floor, the PRR tends towards 100

When TOSSIM has to decide whether a particular node received a packet or not, it first uses the user fed noise level and computes the SNR. Using the SNR value it calculates the corresponding PRR say 'p', using a curve that is hardcoded into TOSSIM. According to this curve, PRR tends to 0% when SNR is close to 0 dB and PRR tends to 100% when SNR is above +10 dB. It then picks a random number between 0 to a 100 and compares it against 'p'. If this number is less than 'p' then the message is considered received. Otherwise, the message is discarded. This process is carried out for every packet that is received by any node in the network.

Thus, the random nature of message reception can cause fluctuations in the number of packets in the network. If a message is not received, the packet is retransmitted

and the number of retransmissions thus, can vary from one run to another in a random manner.

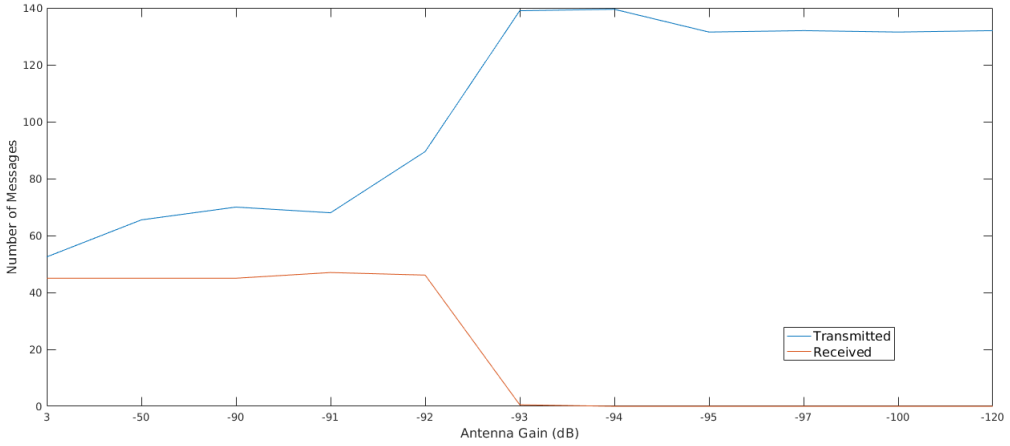


FIGURE 4.8: Variation of the number of messages sent and received for -75 dB antenna gain)

The variation of the number of packets in the network can affect our ability to analyse network performance. So, a bash script that simulated the TOSSIM network 30 times and records the number of sent and received messages was used. The values obtained are averaged and the plot of the number of transmitted and received packets is compared as in Figure 4.8. This plot shows that the network performs as expected. When the antenna gain is comparatively higher than the noise floor the number of messages sent has the lowest value and the number of received packet is 45, which is the number of messages required by the network to create a network map. As the antenna gain approaches the noise floor, the number of messages received drops to zero while the number of messages sent reaches a high and levels out at around 130 packets.

Next, the percentage of successfully acknowledged packets during the neighbor discovery phase is calculated. Figure 4.9, Figure 4.10 and, Figure 4.11 represent the variation of the percentage of acknowledged messages during the Neighbor Discovery Phase.

The percentage of the number of successfully acknowledged messages during the Neighbor Discovery Phase follows a practical trend. When the SNR is highest i.e. antenna gain is +3 dB and the noise is around -90 dB, all messages sent by the network are received. This is described by Figure 4.9 where the percentage of successful transmissions is a 100%.

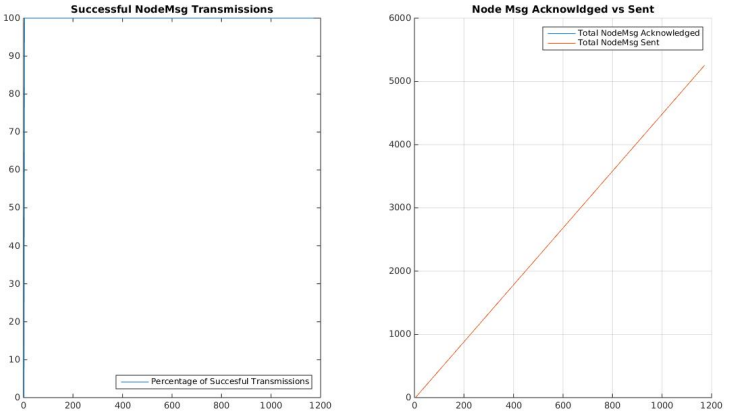


FIGURE 4.9: Neighbor Discovery Performance for +3 dB antenna gain with -90 dB Noise floor (left). All packets are received and the packet reception rate is 100%. Total Number of messages sent vs Total number of messages acknowledged (right)

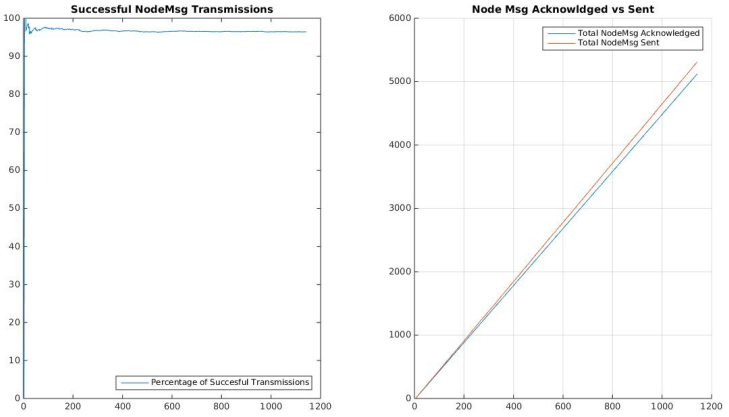


FIGURE 4.10: Neighbor Discovery Performance for -50 dB antenna gain with -90 dB Noise(left). Total Number of messages sent vs Total number of messages acknowledged (right)

As the antenna gain decreases, to -50 dB , the percentage of successful transmissions also decreases to around 95%. This is supported by the small difference in the total number sent and the number of messages acknowledged.

This difference increases when the gain decreases further to -75 dB as shown in Figure 4.11. This difference results in a lower success rate of a little less than 90%.

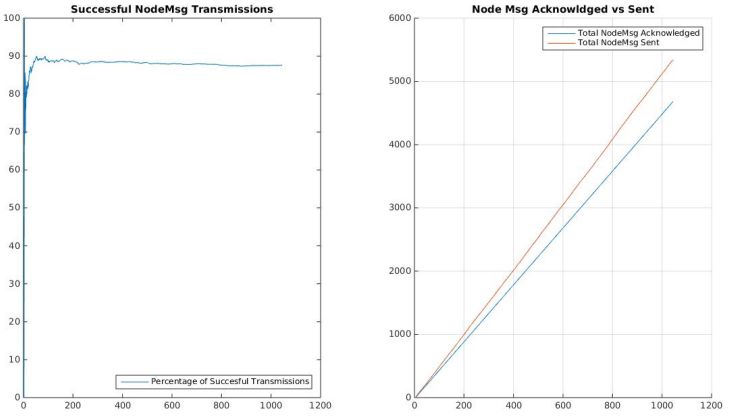


FIGURE 4.11: Neighbor Discovery Performance for -75 dB antenna gain with -90 dB Noise(left). Total Number of messages sent vs Total number of messages acknowledged (right)

4.3.2 Routing Phase Performance

The performance for the Routing Phase substantially decreases because of the sheer number of messages that of transmitted. The periodic timer that determines when the node has to sense its environment is kept low, at 2000 ticks of the node counter which is once every 2 seconds.

Figure 4.13, Figure 4.12, and Figure 4.14 show the variation of successful packet transmissions for networks with antenna gain varying from +3 dB, -50 dB to -75

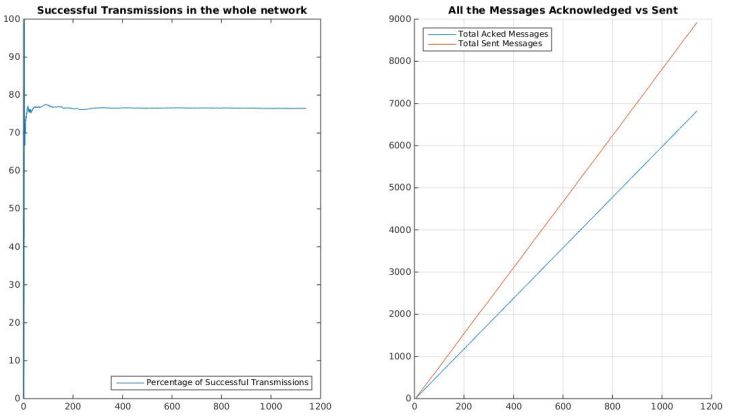


FIGURE 4.12: Routing Phase Performance for the 7 node network with -50 dB antenna gain and -90 dB Noise(left). Total Number of messages sent vs Total number of messages acknowledged (right)

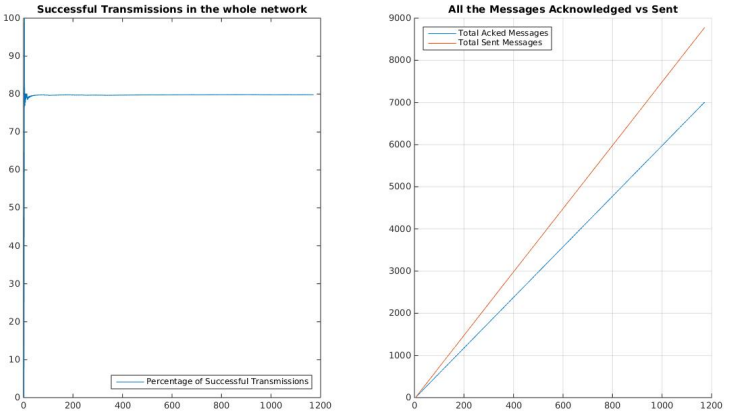


FIGURE 4.13: Routing Phase Performance for the 7 node network with +3 dB antenna gain and -90 dB Noise(left). Total Number of messages sent vs Total number of messages acknowledged (right)

dB respectively.

The success rate for the 7 node topology with -90 dB noise floor falls to 80% for +3 dB antenna gain, 75% for -50 dB antenna gain and less than 70% for -75 dB antenna gain. This is naturally accompanied by an increasing difference between the total number of messages sent and the total number of acknowledged messages.

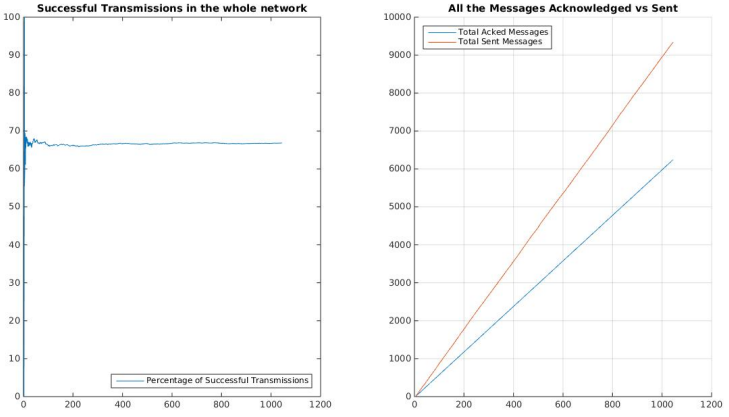


FIGURE 4.14: Routing Phase Performance for the 7 node network with -75 dB antenna gain and -90 dB Noise(left). Total Number of messages sent vs Total number of messages acknowledged (right)

Routing Performance is also evaluated for the 49-node topology. This is done for a range of antenna gain and in turn for a range of SNR values. The noise

remains fixed at around -90 dB. The antenna gain starts at -10dB and all values in decrements of 10dB are used to plot Figure 4.15.

This was done to verify that the success of packet transmission decreased corresponding to the small decrements of antenna gain. As is expected, the percentage of successful transmissions starts at a 100% and drops to less than 50% soon and averages out as time goes by. They all settle between the range 25% to 35%. Figure 4.16 is provided to help follow the plot under various antenna gain easier and is plot from the same values as Figure 4.15. It can be observed that the success of transmission of a packet decreases as the antenna gain decreases from -10 dB to -75 dB.

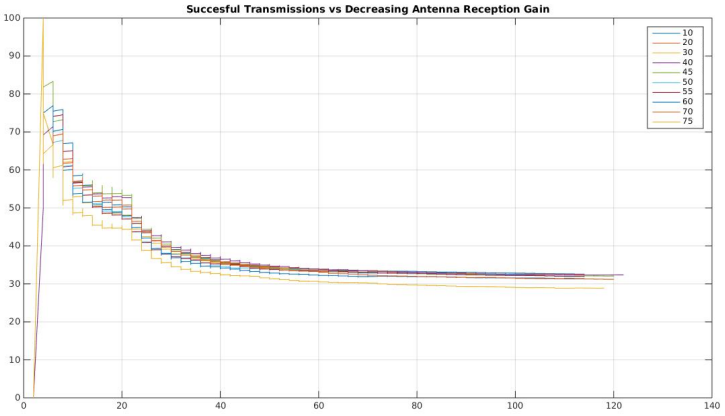


FIGURE 4.15: Percentage of successful transmissions as antenna gain varies from -10 dB to -75 dB

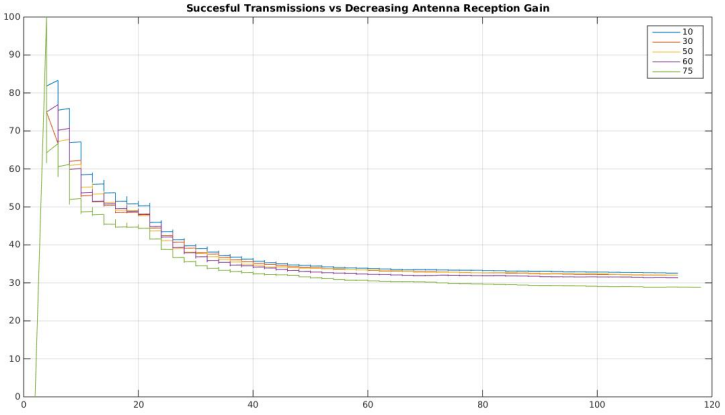


FIGURE 4.16: Percentage of successful transmissions as antenna gain varies from -10 dB, -30 dB, -50 dB, -60 dB, and -75 dB

Chapter 5

Future Work and Conclusion

5.1 Future Work

The SDN-based network does not perform as well as a WSN with traditional decentralized routing protocol. This would be the primary objective of any future work. The performance of the proposed network could be improved by reducing the number of duplicate messages transmitted by the nodes as shown by Figure 4.6.

The proposed network has the capability of changing the routing protocol used in the network by merely changing code at the sink to produce different routing tables and updating the corresponding nodes with the new routing tables. This feature could be explored further by implementing several protocols, interchanging between them and observing changes in network performance for each protocol.

Arguably, the most important proposal of this thesis is the use of dynamic routing tables to change how a node behaves or changing the actions it performs without having to reprogram each node in the network separately. This feature could be incorporated in a decentralized routing protocol. Assuming the sink has the capability of transmitting the routing table to all nodes in the network directly, the overhead of such a feature would be low.

5.2 Conclusion

In conclusion, we present a Wireless Sensor Network routing protocol that has the ability to dictate how each node operates in the network while also providing an easy way to change the routing protocol of the network. The performance of the proposed SDN-based network was not as good as a traditional decentralized routing protocol but, this thesis can be considered as a starting point to improve the design and maximize performance. To the best of our knowledge, the proposed SDN-based WSN routing protocol is a novel idea and further investigations could help bridge the gap in its performance compared to traditional decentralized routing protocols.

Appendix A

Relevant Code

The code for this thesis can be found on https://github.com/SDN-WSN/SDN-WSN.

Bibliography

[1] S.K. Singh, M.P. Singh, and D.K. Singh. A survey of energy-efficient hierarchical cluster-based routing in wireless sensor networks. *International Journal of Advanced Networking and Application (IJANA)*, page 570–580, 2010.

[2] S.K. Singh, M.P. Singh, and D.K. Singh. Energy-efficient homogeneous clustering algorithm for wireless sensor network. *International Journal of Wireless & Mobile Networks (IJWMN)*, pages 49–61, 2010.

[3] Dr. Shu Yinbiao. Internet of things: Wireless sensor networks. *International Electrotechnical Commission (IEC0)*, 2014.

[4] Steve Cosgrove. Software defined networking: Last agianst the wall. *Computing and Information Technology Research and Education*, 2015. URL http://www.citrenz.ac.nz/conferences/2015/pdf/2015CITRENZ_3_Poster_Cosgrove_SoftwareNetworking_v2.pdf.

[5] Jamal N. Al-Karaki and Ahmed E. Kamal. Routing techniques in wireless sensor networks:a survey. *Wireless Communications,IEEE*, 11(6):6–28, December 2004. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1368893&tag=1.

[6] Kemal Akkaya and Mohammed Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3):325–349, May 2005. URL http://www.sciencedirect.com/science/article/pii/S1570870503000738.

[7] Luis Javier Garcia Villalba, Ana Lucila Sandoval Orozco, Alicia Trivino Cabrera, and Claudia Jacy Barenco Abbas. Routing protocols in wireless sensor networks. *Sensors*, 9(11):8399–8421, October 2009. URL http://www.mdpi.com/1424-8220/9/11/8399/htm.

[8] Shio Kumar Singh, M P Singh, and D K Singh. Routing protocols in wireless sensor networks - a survey. *International Journal of Computer Science & Engineering (IJCSES)*, 1(2):1579–1588, November 2010. URL http://airccse.org/journal/ijcses/papers/1110ijcses06.pdf.

[9] S. Horing, J. Z. Menard, R. E. Staehler, and B. J. Yokelson. Stored program controlled network. *The Bell System Technical Journal*, 61(7): 1579–1588, September 1982. URL https://ia601607.us.archive.org/15/items/bstj61-7-1579/bstj61-7-1579.pdf.

[10] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, April 2008. URL http://dl.acm.org/citation.cfm?id=1355746.

[11] Alon Bernstein, Vince Pandolfi, Charles Duffy, and Sangeeta Ramakrishnan. A software defined networking approach to cable wi-fi. *NCTA*, 2013. URL http://www.nctatechnicalpapers.com/Paper/2013/2013-a-software-defined-networking-approach-to-cable-wi-fi.

[12] Rudradeep Nath. A tossim based implementation and analysis of collection tree protocol in wireless sensor networks. *International Conference on Communication and Signal Processing*, pages 484–488, 2013.

[13] Mohammad S. Obaidat, Sanjay K. Dhurandher, Deepak Gupta, Nidhi Gupta, and Anupriya Asthana. Deesr: Dynamic energy efficient and secure routing protocl for wireless sensor networks in urban environments. *Journal of Information Processing Systems*, 6(3):269–294, September 2010. URL http://52.68.174.105:8080/jips/dlibrary/JIPS_v06_no3_paper01.pdf.

[14] Jing Deng, Richard Han, and Shivakant Mishra. Insens- intrusion-tolerant routing for wireless sensor networks. *Computer Communications*, 29(2): 216–230, January 2006. URL http://www.sciencedirect.com/science/article/pii/S0140366405002045.

[15] K. Saleem, N. Fisal, S. Hafizah, S. Kamilah, and R. A. Rashid. A self-optimized mulitpath routing protocol for wireless sensor networks. *International Journal of Recent Trends in Engineering*, 2(1):93–97, November 2009. URL http://www.ijrte.academypublisher.com/vol02/no01/ijrte02019397.pdf.

[16] Lakshminarayanan Subramanian and Randy H.Katz. An architecture for building self-configurable systems. *Mobile and Ad Hoc Networking and Computing (MobiHOC)*, pages 63–73, 2000. URL http://sahara.cs.berkeley.edu/papers/SK00.pdf.

[17] Katayoun Soharabi, Jay Gao, Vishal Ailawadhi, and Greg Pottie. Self organising wireelss sensor network. *IEEE Personal Communications*, 7:16–27, 2000. URL http://www.seas.ucla.edu/~pottie/papers/WSNs00.pdf.

[18] Binwei Deng and Guangming Huang. Algorithm of creating routing table for wsns. *Wireless Communications, Networking and Mobile Computing*, pages 1–5, September 2009. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5301357&tag=1.

[19] Rik Sarkar, Xianjin Zhu, and Jie Gao. Distributed and compact routing using spatial distributions in wireless sensor networks. *ACM Transactions on Sensor Networks*, 9(3), May 2013. URL http://dl.acm.org/citation.cfm?id=2480735.

[20] Jun Xia, Fei Yin, Yun Rui, Kai Yu, Zhenhong Li, Haifeng Wang, and Zhiyong Bu. Beacon routing algorithm in wireless sensor netowrks with mobile gateway. *Journal on Wireless Communications and Networking*, March 2012. URL http://jwcn.eurasipjournals.com/content/2012/1/86.

[21] Xheng Sun, Xiao guang Zhang, Hui Li, and Anqi Li. The applications of tinyos beaconing wsn protocol in mine safety monitoring. *Mechtronic and Embedded Systems and Applications*, pages 415–419, October 2008. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4735665&tag=1.

[22] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. *Mobicom '00 Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 56–67, 2000. URL http://dl.acm.org/citation.cfm?id=345920.

[23] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. *Proceedings of the First Workshop on Sensor Networks and Applications (WSNA)*, October 2002.

[24] F. Ye et al. A scalable solution to minimum cost forwarding in large scale sensor networks. *Proceedings of International Conference on Computer Communications and Networks (ICCCN)*, October 2001.

[25] L. Subramanian and R.H. Katz. An architecture for building self configurable systems. *Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing*, August 2000.

[26] V. Rodoplu and T.H. Ming. Minimum energy mobile wireless networks. *IEEE Journal of Selected Areas in Communications*, page 1333–1344, 1999.

[27] L. Li and J. Y Halpern. Minimum energy mobile wireless networks revisited. *Proceedings of IEEE International Conference on Communications (ICCÕ01)*, June 2001.

[28] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. *Proceedings of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiComÕ01)*, July 2001.

[29] K. Akkaya and M. Younis. An energy-aware qos routing protocol for wireless sensor networks. *Proceedings of the IEEE Workshop on Mobile and Wireless Networks (MWN 2003)*, May 2003.

[30] Y. Yu, D. Estrin, and R. Govindan. Geographical and energy-aware routing: a recursive data dissemination protocol for wireless sensor networks. *UCLA Computer Science Department Technical Report*, May 2001.

[31] Ossama Younis and Sonia Fahmy. Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach. *International Journal of Computer Science & Engineering Survey (IJCSES)*, November 2010.

[32] B. Karp and H. T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. *Proceedings ACM MobiCom'00*, pages 243–254, Aug. 2000.

[33] Shio Kumar Singh, M P Singh, and D K Singh. Routing protocols in wireless sensor networks - a survey. *International Journal of Computer*

Science & Engineering (IJCSES), 1(2):1579–1588, November 2010. URL http://airccse.org/journal/ijcses/papers/1110ijcses06.pdf.

[34] J. Luo and J. P. Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. *Proceedings IEEE INFOCOM'05*, pages 1735–1746, Mar. 2005.

[35] R.C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling a three-tier architecture for sparse sensor networks. *Proceedings SN P A '03*, pages 30–41, May 2003.

[36] Wendi Rabiner Heizelman, Joanna Kulik, and Hari Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. *MobiCom '99 Proceedings of the 5th annual ACM/IEEE International conference on Mobile computing and netowrking*, pages 174–185, 1999. URL http://dl.acm.org/citation.cfm?id=313529.

[37] R. Shah and J. Rabaey. Energy aware routing for low energy ad hoc sensor networks. *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, March 2002.

[38] C. Schurgers and M.B. Srivastava. Energy efficient routing in wireless sensor networks. *The MILCOM Proceedings on Communications for Network-Centric Operations: Creating the Information Force*, 2001.

[39] 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*, page 293–313, 2002.

[40] Y. Yao and J. Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Record*, September 2002.

[41] N. Sadagopan et al. The acquire mechanism for efficient querying in sensor networks. *Proceedings of the First International Workshop on Sensor Network Protocol and Applications*, May 2003.

[42] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless sensor networks. *Proceeding of the Hawaii International Conference System Sciences*, January 2000.

[43] A. Manjeshwar and D.P. Agrawal. Teen: a protocol for enhanced efficiency in wireless sensor networks. *Proceedings of the 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, April 2001.

[44] A. Manjeshwar and D.P. Agrawal. Apteen: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. *Proceedings of the 2nd International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile computing*, April 2002.

[45] S. Lindsey and C.S. Raghavendra. Pegasis: power efficient gathering in sensor information systems. *Proceedings of the IEEE Aerospace Conference*, March 2002.

[46] M. Younis, M. Youssef, and K. Arisha. Energy-aware routing in cluster-based sensor networks. *Proceedings of the 10th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS2002)*, October 2002.

[47] J.-H. Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *Proceedings of the Advanced Telecommunications and Information Distribution Research Program (ATIRPÕ2000)*, March 2000.

[48] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Maximum lifetime data gathering and aggregation in wireless sensor networks. *Proceedings of IEEE International Conference on Networking (NETWORKS Õ02)*, August 2002.

[49] I.F. Akyildiz et al. Wireless sensor networks: a survey. *Computer Networks*, page 393–422, 2002.

[50] T. He et al. Speed: a stateless protocol for real-time communication in sensor networks. *Proceedings of International Conference on Distributed Computing Systems*, May 2003.

[51] Chatterjea S, De Luigi S, and Havinga P. Dirq: A directed query dissemination scheme for wireless sensor networks. *In Proceedings of the IASTED International Conference on Wireless Sensor Networks (WSN)*, July 2006.

[52] Barenco Abbas C.J., González R., Cárdenas N., and García Villalba L.J. A proposal of a wireless sensor network routing protocol. *Telecommun. Syst.*, page 61–68, 2008.

[53] Chang J.H. and Tassiulas L. Energy conserving routing in wireless ad hoc networks. *Proceedings of the 19th Conference of the IEEE Communications Society (INFOCOM)*, page 22–31, March 2000.

[54] Shah R.C. and Rabaey J.M. Energy aware routing for low energy ad hoc sensor networks. *Proceedings of the Wireless Communications and Networking Conference (WCNC)*, page 350–355, March 2002.

[55] De S., Qiao C., and Wu H. Meshed multipath routing with selective forwarding: an efficient strategy in wireless sensor networks. *Comput. Netw.*, page 481–497, 2003.

[56] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *Proceedings ACM MobiCom'01*, pages 85–96, July 2001.

[57] B. Nath and D. Niculescu. Routing on a curve. *ACM SIGCOMM Computer Communication Review*, pages 155–160, Jan. 2003.

[58] G. Xing, C. Lu, R. Pless, and Q. Huang. On greedy geographic routing algorithms in sensing-covered networks. *Proceedings ACM MobiHoc'04*, pages 31–42, May 2004.

[59] M. Zorzi and R. R. Rao. Geographic random forwarding (geraf) for ad hoc and sensor networks: Mutlihop performance. *IEEE Transactions on mobile Computing*, pages 337–348, Oct.-Dec. 2003.

[60] A. Boukerche, X. Cheng, and J. Linus. Energy-aware data-centric routing in microsensor networks. *Proceedings ACM MSWiM, in conjunction with ACM MobiCom*, pages 42–49, Sept. 2003.

[61] Shio Kumar Singh, M P Singh, and D K Singh. Routing protocols in wireless sensor networks - a survey. *International Journal of Computer Science & Engineering (IJCSES)*, 1(2):1579–1588, November 2010. URL http://airccse.org/journal/ijcses/papers/1110ijcses06.pdf.

[62] Shio Kumar Singh, M P Singh, and D K Singh. Routing protocols in wireless sensor networks - a survey. *International Journal of Computer Science & Engineering (IJCSES)*, 1(2):1579–1588, November 2010. URL http://airccse.org/journal/ijcses/papers/1110ijcses06.pdf.

[63] K. Akkaya and M. Younis. An energy-aware qos routing protocol for wireless sensor networks. *Proceedings of the IEEE Workshop on Mobile and Wireless Networks (MWN 2003)*, May 2003.

[64] Shio Kumar Singh, M P Singh, and D K Singh. Routing protocols in wireless sensor networks - a survey. *International Journal of Computer Science & Engineering (IJCSES)*, 1(2):1579–1588, November 2010. URL http://airccse.org/journal/ijcses/papers/1110ijcses06.pdf.

[65] Shio Kumar Singh, M P Singh, and D K Singh. Routing protocols in wireless sensor networks - a survey. *International Journal of Computer Science & Engineering (IJCSES)*, 1(2):1579–1588, November 2010. URL http://airccse.org/journal/ijcses/papers/1110ijcses06.pdf.

[66] W. Chang, G. Cao, and T. La Porta. Dynamic proxy tree-based data dissemination schemes for wireless sensor networks. *Proceedings IEEE MASS'04*, pages 21–30, Oct. 2004.

[67] Shio Kumar Singh, M P Singh, and D K Singh. Routing protocols in wireless sensor networks - a survey. *International Journal of Computer Science & Engineering (IJCSES)*, 1(2):1579–1588, November 2010. URL http://airccse.org/journal/ijcses/papers/1110ijcses06.pdf.

[68] S. Lindsey, C. S. Raghavendra, and K. M. Sivalingam. Data gathering in sensor networks using the energy delay metric. *Proceedings IPDPS'01*, pages 2001–2008, Apr. 2001.

[69] M. Chu, H. Haussecker, and F. Zhao. Scalable information-driven sensor querying and routing for adhoc heterogeneous sensor networks. *International Journal of High Performance Computing Applications*, pages 293–313, Feb. 2002.

[70] S. Lindsey, C. S. Raghavendra, and K. M. Sivalingam. Data gathering algorithms in sensor networks using energy metrics. *IEEE Transactions on Parallel and Distributed Systems*, pages 924–935, Sept. 2002.

[71] X. Du and F. Lin. Improving routing in sensor networks with heterogeneous sensor nodes. *Proceedings IEEE VTC'05*, pages 2528–2532, Sept. 2005.

[72] J. Kulik, W. R. Heinzelman, and H. Balakrishnan. Negotiation-based protocols for disseminating information in wireless sensor networks. *Wireless Networks*, pages 169–185, 2002.

[73] Q. Li, J. Aslam, and D. Rus. Hierarchical power-aware routing in sensor networks. *In Proceedings of the DIMACS Workshop on Pervasive Networking*, May 2001.

[74] Jamal N. Al-Karaki, Raza Ul-Mustafa, and Ahmed E. Kamal. Data aggregation in wireless sensor networks - exact and approximate algorithms'. *Proceedings of IEEE Workshop on High Performance Switching and Routing (HPSR) 2004*, pages 18–21, April 2004.

[75] S. Dulman, T. Nieberg, J. Wu, and P. Havinga. Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks. *WCNC Workshop*, March 2003.

[76] Q. Fang, F. Zhao, and L. Guibas. Lightweight sensing and communication protocols for target enumeration and aggregation. *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing (MOBIHOC)*, pages 165–176, 2003.

[77] Q. Fang, F. Zhao, and L. Guibas. Lightweight sensing and communication protocols for target enumeration and aggregation. *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing (MOBIHOC)*, pages 165–176, 2003.

[78] Q. Fang, F. Zhao, and L. Guibas. Lightweight sensing and communication protocols for target enumeration and aggregation. *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing (MOBIHOC)*, pages 165–176, 2003.

[79] I. Stojmenovic and X. Lin. Gedir: Loop-free location based routing in wireless networks. *International Conference on Parallel and Distributed Computing and Systems*, pages 3–6, Nov. 1999.

[80] I. Stojmenovic and X. Lin. Gedir: Loop-free location based routing in wireless networks. *International Conference on Parallel and Distributed Computing and Systems*, pages 3–6, Nov. 1999.

[81] I. Stojmenovic and X. Lin. Gedir: Loop-free location based routing in wireless networks. *International Conference on Parallel and Distributed Computing and Systems*, pages 3–6, Nov. 1999.

[82] F. Kuhn, R. Wattenhofer, and A. Zollinger. Worst-case optimal and average-case efficient geometric ad-hoc routing. *Proceedings of the 4th ACM International Conference on Mobile Computing and Networking*, pages 267–278, 2003.

[83] Jamal N. Al-Karaki and A.E. Kamal. On the correlated data gathering problem in wireless sensor networks. *Proceedings of The Ninth IEEE Symposium on Computers and Communications*, July 2004.

[84] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost outdoor localization for very small devices. *Technical report 00-729, Computer science department, University of Southern California*, Apr. 2000.

[85] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. Tinyos: An operating system for sensor networks. *Ambient intelligence. Springer Berlin Heidelberg*, pages 115–148, 2005. URL https://people.eecs.berkeley.edu/~culler/papers/ai-tinyos.pdf.

[86] Tossim. *TinyOS Wiki*, . URL http://tinyos.stanford.edu/tinyos-wiki/index.php/TOSSIM.

[87] Network protocols. *TinyOS Wiki*, . URL http://tinyos.stanford.edu/tinyos-wiki/index.php/Network_Protocols.

[88] Micaz. *TinyOS Wiki*, . URL http://tinyos.stanford.edu/tinyos-wiki/index.php/MICAz.

[89] Peter PECHO, Petr HANA CEK, and Jan NAGY. Simulation and evaluation of ctp and secure-ctp protocols. *RADIOENGINEERING*, APRIL 2010. URL http://www.radioeng.cz/fulltexts/2010/10_01_089_098.pdf.

[90] Platform hardware. *TinyOS Wiki*, . URL http://tinyos.stanford.edu/tinyos-wiki/index.php/Platform_Hardware.