Power Optimization through Global Sleep Schedule for MAC Protocols in Wireless Sensor Networks

Stephanie Imelda Pella

Bachelor of Engineering

A thesis submitted in total fulfilment of the degree of

Master of Science (Computer Science)

School of Engineering and Mathematical Sciences
College of Science, Health and Engineering
La Trobe University
Melbourne, Australia

September 21, 2016

Declaration

I certify that, except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and, any editorial work, paid or unpaid, carried out by a third party is acknowledged.

Stephanie Imelda Pella

21 September 2016

Credits

Parts of the work presented in this thesis have been submitted in the following forums:

- An Offset Based Global Schedule Algorithm for Self-Organizing Wireless Sensor Networks (Accepted to be presented in The 15th IEEE International Symposium on Network Computing and Applications (NCA 2016))
- Performance Analysis of Synchronized Duty Cycle Protocols in Wireless Sensor Networks with and without Hidden Terminals (Submitted to TELKOMNIKA (Telecommunication, Computing, Electronics and Control))

Acknowledgement

All praise, honour and glory to my lord and saviour, Jesus Christ, for His richest grace and mercy on the accomplishment of this thesis.

This thesis becomes a reality with kind support of many individuals that I like to extend my sincere thanks to all of them

I would like to express my sincere gratitude to my advisors Dr Prakash Veeraraghavan and Dr Somnath Ghosh for the continuous support of my study for their patience, motivation, and immense knowledge. Their guidance helped me through the research and writing parts of this thesis. Besides my advisors, I would like to thank the rest of my thesis committee: A/Prof. Seng Loke and Dr. Kinh Nguyen, for their insightful comments and encouragements, but also for the hard questions, which incited me to widen my research from various perspectives.

My deepest gratitude is for my family who supported me from afar through their love, encouragements and prayers. I would like to express my love and thanks to my Mom and Dad, my first teachers who taught and shaped me to be who I am now. And, to my sister, Anita, my first and most loyal friend who always loves me unreservedly. The last but definitely not the least, to my husband, Hendro, my closest friend and partner in life, who continuously loves and supports me and makes the completion of this study possible.

This research was financially supported by the Australian Awards Scholarship from the department of Foreign Affairs and Trade (DFAT) Australia. I am grateful to the Australian Government for providing this generous funding.

Abstract

In wireless sensor networks (WSNs), the low duty-cycled scheme is widely used to conserve the networks' energy. In this scheme, each node alternately switches its radio module between the *sleeping mode* (off) and the *active mode* (on) according to a certain schedule, called the *sleeping schedule*. The nodes in a network that operate on the same schedule form a virtual cluster. Having a multi-cluster network, though it is common in a multi-hop WSN, is disadvantageous since it causes a higher energy depletion rate in the nodes located near the border of two or more clusters. Our study focuses on achieving a global schedule in a self-organizing WSN, where there is no pre-configuration and global time reference. There are two main outcomes of this study: (1) the development of an analytical model to evaluate the performance of the duty-cycle-based WSN under both a single and multi-cluster scenario; and (2) the development of an algorithm to achieve a *global schedule* in self-organizing WSNs.

We begin by investigating the effect of having a multi-cluster network on the energy wastage caused by data packet collision in the border nodes. We divide the work into two major parts, namely, the development of a Markov model to analyse the stationary performance of a single cluster network and extending the model to analyse additional energy wastage due to packet retransmission in the border nodes of a multi-cluster network. The proposed Markov model consists of an idle state, representing a condition where there is no packet in a node's transmission buffer and contending states, representing the back-off counters during the time a node contends to access the channel. From the model, we derives the equations to calculate packet delivery ratio (PDR) and the network throughput for various densities and offered load, which we validate using simulations. The result provides a way to calculate the maximum allowed network load for a certain network density to achieve a specific PDR requirement. For example, for network density n = 5, to achieve 90% of PDR, the maximum packet arrival rate allowed (λ) is 0.27, and for n = 7 and the same PDR requirement, the maximum λ is 0.14. The result also shows the relationship of the network throughput and the offered load for various network densities. We progress to the second part of the work by calculating the energy waste due to packet retransmissions in a multi-cluster network and compare it to the ones in a single cluster network. Both the analytical model and the

simulation show that having a single cluster network can save up to 90% of energy wastage due to the retransmission of collided packet.

We then proceed with the investigation of the schedule drift problem in a network, in which the nodes have different perceptions regarding the start of a schedule. We begin by calculating the maximum propagation path length in a network for various sizes and densities of a network and transmission ranges of the nodes, using the space curve filling algorithm, a set of simulations and a mathematical estimation. Since the nodes in the network communicate their schedule by propagating control packets, the maximum schedule drift in a network is proportional to the maximum propagation path length of the network.

The development of our protocol comprises three main tasks, namely, defining the set of rules to choose a winning schedule, developing a merging procedure and evaluating the performance of the protocols. We propose the use of the offset of two schedules as the winning schedule criteria. We use the maximum schedule drift calculated in the previous part of the work in defining a set of rules that a node uses to decide which schedule to follow when encountering more than one schedule. Unlike the other global schedule protocols where a node makes a decision for itself when discovering another schedule, in the proposed merging procedure, a border node makes the decision for its cluster and propagates the decision to the network. We evaluate the performance of the proposed protocol using a mathematical estimation and simulation. The results show that the proposed scheme has up to 50 times shorter convergence time and saves up to 90% more energy during the merging process compared to the other global schedule protocols.

Contents

List of Figures

List of Tables

List of Acronyms

ACK	Acknowledgement
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CTS	Clear to Send
CWND	Contention Window
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronics Engineers
LPL	Low Power Listening
MAC	Medium Access Control
PDR	Packet Delivery Rate
RTS	Request to Send
S-MAC	Sensor MAC
TDMA	Time Division Multiplexing
WSN	Wireless Sensor Network

Chapter 1 Introduction

1.1 Self-Organizing Wireless Sensor Networks

Wireless Sensor Networks (WSNs) provide a way to remote monitor physical environments via communication networks. A WSN comprises a large number of sensor nodes and one or more sinks. The sensor nodes periodically sample the physical environment and send the data to the sinks, which report it to the outside network to be used by the intended users of the network. The nodes in WSNs communicate with each other and with the sinks in a multi-hop manner and usually through a low data rate channel [1, 2]. WSNs have been implemented in various areas of applications, such as health systems [3-5], military applications[6], smart home networks [7, 8] and disaster monitoring systems [9-13].

One of the most widely implemented and highly researched WSN applications is the natural disaster monitoring system. Statistics show that natural disasters often result in heavy casualties, immense financial loss, the destruction of the natural environment and even the loss of human lives. The 2004 Indian Ocean earthquake and tsunami, for instance, killed +220,000 people in 14 countries and caused more than USD 10 billion of financial loss [14]. Official data released by the Ministry of Environment and Forestry in Indonesia reveal that 71,000 Ha of the rain forest area in Indonesia were wiped out between 2010 and 2015 by forest fires [15]. On top of this, the fire caused around 200,000 cases of respiratory tract infection related diseases, the majority of cases affecting children[16]. The early detection of a disaster provides a way to minimize the cost of the disaster. In the case of preventable disaster, early detection could help to avoid or locale the disaster to prevent further damage. In an unpreventable disaster, early detection gives extra time for the authorities to relocate people and valuable assets to save human lives and minimize financial loss.

Disaster monitoring systems, such as forest fire monitoring applications [10, 11] have a self-organizing nature, in which the nodes do not have initial information on the network, such aso their positions and the network topology. During their operational time, the nodes must be able to form a network dynamically and autonomously. A self-organizing network is commonly randomly deployed. Figure 1-1 shows an example of a randomly deployed forest monitoring system where the nodes are deployed by throwing them out of an airplane. In the early phase of the network, the nodes need to construct the network by communicating with their neighbouring nodes. Generally, nodes have a limited radio range, thus, the nodes in WSN communicate with the sinks in a multi-hop manner.

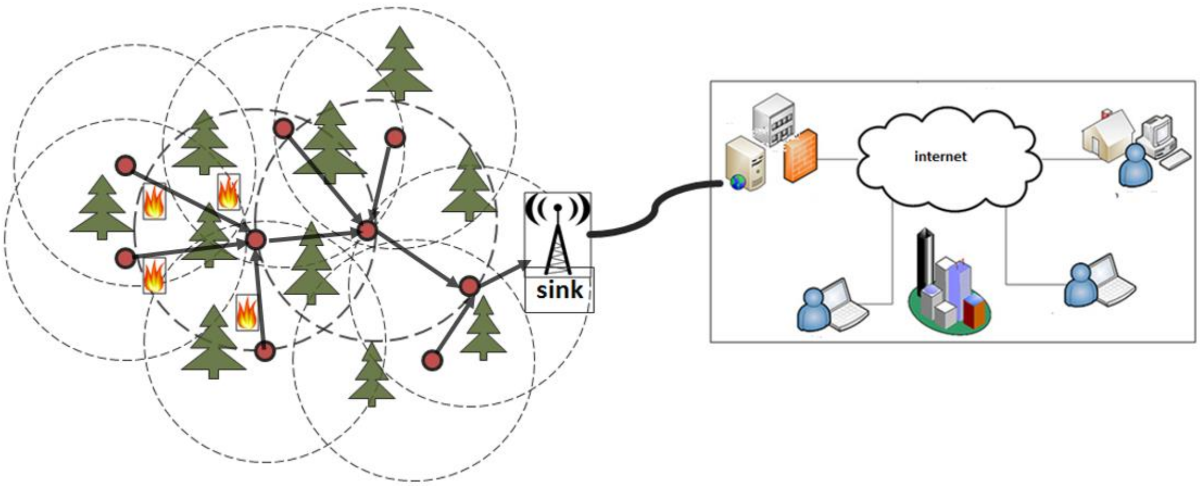

Figure 1-1 A Randomly Deployed Self-Organizing Sensor Network

Self-organizing WSNs have several advantages over the other types of WSNs. Due to the ability of the nodes to autonomously construct a network, the network can be deployed randomly. This is not only cost effective, but also reduces the difficulty of placing the nodes in hard-to-reach locations, such as a rain forest, a volcano or a battle field. Another advantage is that it reduces the cost of pre-configuring a large number of nodes prior to deployment. Finally, due to the nodes' ability to dynamically learn and construct the network, this type of WSN has a high fault tolerance.

The system should satisfy the following characteristics in order to make the application cost effective:

(1) **Low production cost:** Due to the number of nodes in the network, most sensor networks use low-cost devices with limited resources (energy, computation, and networking) to suppress the cost of production. A project called “Smart Dust”[17] aims to develop compact size nodes at a cost so low that the nodes can easily be discharged when the network lifetime expires.

(2) **Long operational time:** Related to point (1), the nodes in the sensor networks need to employ an energy efficient algorithm against their limited power resources. Installing new nodes during the operational network time is not only costly, but also causes network reconfiguration

(3) **Minimal installation and maintenance cost**. The use of a distributed algorithm, which enables nodes to learn about their surroundings and start building the network upon booting up, can suppress the pre-configuration cost. Also, the distributed algorithm helps the nodes to adapt to network changes.

(4) **Fault Tolerant:** In WSNs, some nodes may fail during their operational time. They could run out energy or be physically damaged because of the harsh environment. The remaining nodes should be able to rebuild the network in case one or more nodes fail.

Of all the characteristics mentioned above, energy constraint in the network has been the most discussed subject of many studies. In most applications, the nodes in the networks are expected to work for a long time and are only equipped with small batteries as the energy source. Due to the cost, replacing the batteries in the node is generally not an option. The most common approach to conserve network energy is to reduce energy waste in the networks

1.2 A Multi-Cluster Wireless Sensor Network

WSNs are generally low traffic networks because, between the sensing periods, the nodes do not generate data packets. In a forest fire monitoring system, for example, the sampling periods are in the order of 2-7 minutes, so for the largest portion of the operational time, the nodes are in idle mode. A study in [18] shows that the energy spent by a wireless node in idle mode is up to 50% of the energy spent by the node when it transmits or receives a packet. Hence, most of the energy saving protocols in WSNs are developed to reduce the nodes' idle time by allowing them to turn off their radio modules when they are not participating in a packet transmission. This scheme is called the *duty cycle* scheme.

In a duty cycle-based network, the nodes periodically turn off their radio modules according to certain schedules. The operational period of a node is divided into the active period, when its radio module is on and the sleeping period, when its radio module is off. In order to work well in a self-organizing environment, the duty cycle protocols either (1) let the nodes choose their sleeping schedule randomly in the initialization process, pioneered by B-MAC [19] or (2) require the nodes to listen for and adopt an existing schedule in their neighbourhood during the initialization process, pioneered by S-MAC [20, 21]. However, if the nodes fail to discover a schedule, they are permitted to create their own schedule randomly. Both approaches could generate a multi-schedule network.

In a network where the nodes operate on more than one schedule, the nodes that follow a common schedule form a *virtual cluster*. Initially, the nodes belonging to different clusters cannot send and receive packets from each other because they have different active periods. Hence, the border nodes need to have a special arrangement to act as the gateway between clusters. Preliminary studies on multi-cluster WSNs in [22-24] show that the border nodes have higher energy depletion rates than the other nodes. Moreover, further in this study, we show that they also suffer from more packet collisions. Since the border nodes play such an important role in the network, this could severely affect the whole network performance.

1.3 Problem Statement

As discussed in the previous sections, energy conservation is one of the main concerns in developing WSN protocols. The main approach to conserve energy in the network is the use of the duty cycle scheme, where nodes periodically switch to sleep mode to conserve energy. The existing duty cycle protocols result in the formation of multi-clusters in the networks, which in the end, adversely affect the networks' performance.

It is important to evaluate the performance of a duty cycle protocol in a single cluster network and in a multi-cluster network. Even though there are numerous studies that propose mathematical models to evaluate the performance of wireless networks [57-64], none of them investigates the effects of multi-cluster networks on the performance of the border nodes. One of the aims of this study is to develop an analytical model to analyse the performance of synchronized duty cycle protocols in a single and multi-cluster WSN.

In order to improve the energy saving mechanisms in WSNs, it is desirable that the nodes in a network operate on a single schedule. Having multiple schedules in the early phase of the network operational time is unavoidable, mainly because the network is not yet fully connected. However, after choosing its schedule, if a node discovers another node implementing a different schedule, the node needs to be able to make a decision as to whether or not it adopts the new schedule and abandon its current one. If all the nodes in the network reach a consensus decision, eventually all of them will operate under a single schedule. In a self-organizing network without centralized control, the decision regarding the schedule selection needs to be made in a distributed manner.

There are several challenges in developing a global schedule algorithm for self-organizing WSNs. In a network without a global time reference, it is important to calculate the upper bound of the synchronization drift in the network. This value can be used in determining the important parameters in designing the network, such as the length of the guard bits, the maximum data rate in the network and the maximum schedule drift in duty-cycle networks. The next challenge is designing a schedule selection algorithm. Firstly, the algorithm needs to work without the node having global knowledge of the network. Secondly, it is desirable to have a short convergence time, i.e. the time needs to achieve a single schedule network. Finally, to maintain the energy efficiency of the network, the algorithm also needs to have low computation and small packet control overhead.

1.4 Outline and Main Contributions

We started by conducting an extensive literature study of the existing Medium Access Control (MAC) layer energy conservation protocol in WSNs. In general, energy saving protocols in WSN can be classified into channel-division based protocols and carrier sensing multi-access (CSMA) based protocols. The CSMA-based protocols can be further grouped into synchronized and asynchronous duty cycle protocols. We then continued discussing the formation of multi-cluster networks and how they affect the performance of the border nodes. This serves as the motivation of the study. Understanding the problems of the current state of the existing energy saving mechanism for WSNs, we defined a set of research questions that will be answered in the study

The next step in the study is to develop an analytical model to measure the performance of WSN protocols. We developed a Markov model of a node's MAC state using the parameter of S-MAC, a well-implemented WSN protocol. Using the model, we calculated the throughput of a network with various densities, $n= 3, 5, 7$, and offered loads (*15-210 %* of allowable network capacity). The results show that the throughput has a linear relation to the offered load up to the maximum throughput (≈ 90% of network capacity). After this, the throughput declines as the network enters a saturated condition. The results show a dense network is more prone to throughput reduction than a sparse network.

We then extended the model to a multi-cluster network. We investigated the hidden terminal problem and calculated the energy waste in the border node due to data packet collision in a multi-cluster network for various numbers of clusters (m = 2, 3, 4), packet sizes and packet arrival rates. We then compared the results to similar conditions in the single cluster network. The results show that in a multi-cluster network, the energy waste due to data packet collision in the border nodes is significantly higher than in the single cluster network. In a four-cluster network, for instance, the energy waste due to data packet collision is up to 100 times larger than in a single cluster network.

As part of developing a new global schedule protocol, we need to estimate the maximum schedule drift that is possible in the network. Since the leading cause of schedule drift is propagation delay in a long chain network, a part of our work is estimating the longest path in the network. We started by calculating the theoretical longest path in a network given the dimensions of the network. Then we ran a simulation where we randomly deployed nodes of various sizes and densities of networks and measured the longest path in the network. Based on the simulation results, we derived the estimation of a tighter bound of the maximum longest path in a network. Using the estimated longest path in the network, we can calculate the maximum schedule drift, δ_{max}.

In the final part of the work, we developed a global sleeping schedule protocol to improve the energy saving mechanism in the network. We proposed the offset between two schedules as the winning schedule criteria in our schedule selection algorithm. We define the offset of of two schedules S_1 and S_2, d_{S1s2}, as the minimum duration of the start of a frame in S_2 precedes the start of a frame in S_1. In our proposed algorithm, upon discovering a new schedule and deciding to adopt the schedule, a border node informs the other nodes in the cluster of its decision. It broadcasts a special control packet, called *SYNC-M*, containing the information of the newly discovered schedule. The other nodes in the cluster, using the information in the SYNC-M, will make the same decision, hence the whole cluster will adopt the new schedule at approximately the same time.

In calculating the offset between two schedules, we considered the schedule drift in the network. When the offset between two schedules is in the range of δ_{max}, we assumed that they are actually the same schedule, and the difference in the start of a frame in these two schedules is a result of schedule drift in the network. Finally, we evaluated the performance of our proposed protocols in terms of convergence time and the energy spent in sending the control packets during convergence time. Both the mathematical estimation and simulation show that our proposed protocol has a network convergence time which is approximately 50 times smaller and saves up to 90% more energy in comparison to the other two global schedule protocols [22-24] .

1.5 Organization

The rest of this thesis is organized as follows. Chapter 2 presents the literature survey of existing energy conservation protocols, discusses the main motivation of the study and defines the research questions. In Chapter 3, we develop a model for analysing the performance of synchronous duty cycle protocols. We then extend the model to evaluate the hidden terminal problem in the border nodes of a multi-cluster network. In Chapter 4, we present our proposed protocols to improve the energy saving mechanism in WSNs. Chapter 5 presents the conclusion and future work.

Chapter 2 Background and Literature Review

2.1 Introduction

Most wireless sensor network (WSN) applications, such as disaster monitoring systems, require the network to work for a long period of time. Due to the energy limitation of the nodes, energy efficiency is always one of the main motivations in designing protocols for WSNs. A large number of protocols have been proposed with the main aim of conserving energy of WSNs. In this chapter, we present an extensive review of the important work in the field, discuss the potential problem of the existing energy saving mechanism, and define a set of research questions to be addressed in this study.

The rest of the chapter is organized as follows. Section 2.2 provides the background of the energy constraint problem in WSNs and possible solutions to prolong the network life span. Section 2.3 reviews the state-of-the-art in energy saving protocols in the medium access control (MAC) layer and section 2.4 specifically discussed the carrier sense multi-access (CSMA) based duty cycle protocols. In section 2.5, we present the main motivation of our study. We discuss the formation of multi-clusters in synchronized duty cycle-based networks and the problems it causes. Based on this, we formulate our research questions in section 2.6. Section 2.7 provides the summary of the chapter.

2.2 Energy Constraint in Self-Organizing Wireless Sensor Networks

WSNs are distributed systems designed for sensing and processing information on a specific phenomenon, such as temperature, movement, and humidity, and collaboratively reporting the sensed data via wireless channels [25]. Nodes in WSNs are either sensor nodes or sinks. Sensor nodes gather specific information from the environment and send reporting packets to the sinks, usually in a multi-hop manner. The sinks are special nodes that have communication links with the outside network. They act as the gateway between the authorized users of the applications and the network. WSNs are application-specific networks where the nature of the networks highly depends on the application they run.

WSN applications that are located in hard-to-reach locations or require a large number of nodes usually have a self-organizing nature. In this type of network, the nodes are deployed with minimum or no prearranged configuration. Consequently, initially, they do not have information on their locations, the other nodes in their neighbourhood or the routes to the sink [26]. In the early phase of their operational time, the nodes construct a network by discovering the neighbouring nodes, negotiating channel access and constructing routing tables. The nodes also usually do not have a global time reference, thus synchronization needs to be done locally. The protocols that are developed for a self-organizing WSN need an algorithm to achieve all of the mentioned tasks in a distributed manner.

The nodes in every WSN have an intended lifespan, which usually is related to the application of the network. Nodes in forest fire monitoring applications, for instance, need to stay alive for an entire summer or dry season, which roughly covers 4 to 8 months of operational time, depending on the geographical locations of the networks. A network can only be fully functional if most of the nodes are still operating. When some nodes stop working because they run out of energy, the performance of the network could be severely impacted. In the worst case, it could result in a network partitioning, where some areas of the network are permanently isolated from the main network. Nodes in these areas are unable to send their report to the outside network since there is no available path to any of the sinks.

In most WSNs, energy conservation is among the crucial issues for the following reasons. Firstly, the nodes in WSNs are commonly battery-powered devices with limited energy resources. Due to their location, battery charging and replacing in WSN nodes is much more difficult compared to the any other wireless handhelds. In self-organizing WSNs, energy constraint poses an even bigger problem since replacing the nodes’ batteries in the field is almost impossible due to its randomly deployed nature. Hence, energy conservation is high priority when developing protocols for WSN. Secondly, WSN nodes are active for a longer period compared to any other wireless network application, whereas handhelds can be turned off when they are not participating in packet transmission.

Generally, there are two approaches to tackle the energy constraint problem in WSNs, namely, recharging the nodes' batteries and minimizing energy waste in the network. A common way to recharge the nodes' batteries is by harvesting the natural energy from the nodes' physical environments. The work in [27] summarizes the recent studies on energy harvesting methods in WSNs from various sources of ambient energy such as solar energy, mechanical vibrations, temperature gradient, wind energy, water flow and magnetic fields. On the other hand, the protocols aiming to minimizing energy waste in the network develop schemes to reduce the occurrences of energy wasting events.

Of these two approaches, minimizing energy waste is the most popular approach for several reasons. Adding battery-recharging capabilities increases the cost of each individual node and in the end, the cost of the whole network. Moreover, each time a node runs out of energy, it has to disconnect from the network for a certain amount of time. This affects the whole network performance by causing packet rerouting or even network partition. When the disconnect node is charged and ready to join the network, booting up processes have to be repeated and all other nodes have to readjust to the re-joining node. A network with a high rate of disconnecting and connecting always stays in a converging state.

A plethora of studies has been conducted to minimize energy expenditure in WSNs. The approaches to minimize energy waste in WSNs can be classified into following categories:

(1) **Radio optimization** is mainly done in the physical layer, which includes the use of an optimal modulation [28, 29], transmission power control [30, 31] and the right type of antennas for certain types of WSN applications.

(2) **Data aggregation [32-34]** aims to reduce the number of packets sent in the networks by performing data fusion in the nodes along the path to the sink. For instance, instead of sending each individual report on the sensed data, the intermediate nodes send the average, maximum or minimum received data depending on the requirements of the applications.

(3) **Energy efficient routing** faces two main challenges, namely, (a) finding the shortest route between the nodes and sinks without knowing the network topology and (b) reducing the imbalanced energy expenditure in packet forwarding, i.e. the closer a node to a sink, the more packets it needs to forward. Studies investigating this topic can be grouped into cluster architectures [35], energy (instead of shortest path) as the routing metric [36], multipath routing [37, 38] and mobile sink [39, 40].

(4) **Duty cycle scheme** allows the nodes in the network to turn their radio components off (sleep mode) when they are not supposed to send or receive data to minimize energy expenditure in the nodes. This approach works in the medium access control (MAC) layer.

Of all the approaches, our study focuses on minimizing energy waste in the MAC layer by implementing the duty cycle scheme.

2.3 The State-of-the-Art of Medium Access Control Protocols in Wireless Sensor Networks

Medium access control (MAC) regulates channel access and data transmission in local networks. Since this layer logically manages the radio modules, the most power consuming components in WSN nodes, MAC protocols in WSNs are generally designed to be energy efficient. In this section, we discuss the existing energy efficient MAC protocols, which are developed for the WSN environment. We classify the protocols based on their channel access mechanism and their schedule synchronization approach. We then analyse the performance of each group against the nature of self-organizing networks.

Most energy conservation protocols in the MAC layer implement the duty cycle scheme to conserve energy. This scheme allows the nodes in a network to turn off their radio modules when they are not participating in packet transmission. The condition in which nodes operate with their radio modules off is called the sleep mode. Periodically, nodes in a network need to wake up, i.e. turn on their radio modules to check the channel traffic. During this period, the nodes operate under the active mode. Each node implements a sleep schedule which it chooses during its initialisation process. The schedule determines when the node starts its active and sleep period.

The percentage of time the nodes in the network operate under the active mode defines the duty cycle of the network and, consequently, the energy depletion rate of the network. A small duty cycle results in a small network energy budget. However, in a high traffic network, a small duty cycle yields a small network throughput and long packet delays. Some studies [41, 42], as described later in the chapter, propose ways to implement a dynamic duty cycle in the network, where nodes in the network increase or decrease their duty cycle as the network traffic changes.

The energy waste in the MAC layer is defined as the energy spent by a node that does not result in successful data packet transmission. The work presented in [43] lists the following major sources of energy waste in WSNs.

(1) **Idle listening** is a condition where a node has its radio module on to listen for possible incoming packets during the idle channel. Generally, a WSN is a low traffic network where the nodes do not have data packets to send between sensing, thus, most of the time they operate on idle listening mode.

(1) **Packet collisions** occur when more than one node in a common collision range sends their packets at the same time. Since this does not result in successful transmission, the energy spent to send and receive the packets is wasted.

(2) **Overhearing** refers to the condition where nodes pick up packets that are not intended for them.

(3) **Control overhead** refers to the energy spent to send and receive the control packets.

(4) **Over emitting** is a condition in which a transmitter sends a packet to a receiver which is not ready, thus, the packet needs to be resent.

Based on their channel accessing schemes, energy saving mechanisms in the MAC layer are classified into channel division-based protocols and random access-based protocols. In channel division-based protocols, each node transmits its data packets in its own specific designated slots, based on time, frequencies, or codes, to avoid collision. On the other hand, in random access protocols, channel accessing is based on carrier sense multi-access (CSMA) [18]. Each node that has a packet in its transmission buffer contends for the channel by drawing a back off counter between zero and the contention window (CWND-1). A contending node decrements its counter at the end of every time slot if it senses that the channel is idle. A node has the right to send its packet only if its counter is zero. The channel division-based protocols guarantee collision free transmission. In contrast, in the CSMA-based networks, the probability of collision increases as the network traffic increases.

Despite the advantage of having collision-free transmission, the channel division-based protocols need certain conditions that do not exist in a self-organizing environment. TDMA-based protocols, for instance, are sensitive to clock drift ,which could be a problem in a network without global time reference [43]. Most TDMA protocols assume that nodes' clocks are somehow synchronized and clock drift can be corrected by using a time stamp mechanism or GPS [44, 45], which in the end increases the network cost.

The distributed nature of CSMA protocols makes them suitable for implementation in self-organizing networks with minimal or no pre-configuration. A CSMA-based duty cycle protocol can either be an asynchronous or a synchronized duty cycle protocol. In the synchronized duty cycle-based network, nodes try to synchronize their schedule with their neighbour, i.e. wake up at the same time, to enable the nodes to communicate with each other. In the asynchronous duty cycle protocols, on the other hand, each node operates on a schedule that is independent from their neighbours' schedule. This study focuses on synchronized duty cycle protocols. In Section 2.4, we discuss this class of protocols in detail.

The asynchronous duty cycle protocols use the low power listening (LPL) scheme to conserve the network energy. The nodes in an LPL-based network are mainly in sleeping mode when they are not participating in data transmission. Periodically, the nodes poll the channel by waking up briefly. A transmitter, before transmitting its packet, needs to send a preamble to notify its intended receiver. Following a successful preamble, the transmitter and receiver engage in data packet transmission while other nodes in the neighbourhood switch to sleep mode for the period of data transmission.

Several protocols have been developed, based on the LPL scheme. Berkeley Medium Access Control (B-MAC) [19],Wise-MAC [46] and synchronized channel polling (SCP)[47] are some examples of the long preamble LPL protocols. Each time a node wants to transmit its packet, it sends a preamble that is longer than the network's polling period to ensure the preamble intersects with the receiver polling time. The energy to send a long preamble contributes to the control overhead, one of the major energy waste sources in WSNs. The recent LPL protocols try to reduce the control overhead by replacing the long preamble with a burst of a short preamble (e.g. X-MAC [48]), the receiver initiated transmission (e.g. RI-MAC[49]) and the pseudo random schedule (e.g. PW-MAC[50]).

Asynchronous duty cycle protocols save more energy than synchronized duty cycles in a low traffic network. However, the advantages are accompanied by several major drawbacks. The low energy used in the receivers causes high-energy consumption on the transmitter sides. When the traffic of a network increases, such as during event detection, the collision rate of the preambles increases significantly. This wastes the energy spent on transmitting the preamble and decreases the throughput of the network.

2.4 Synchronized Duty Cycle Protocols

In this section, we discuss the well-known existing synchronized duty cycle protocols. The protocols belonging to this class employ a mechanism to enable the nodes to inform their schedules to their neighbourhood. Most of the protocols in this class prefer that all the nodes in a network operate on a single common schedule. However, to maintain the distributed nature of the network, they allow the nodes to generate new schedules in case they cannot discover an existing one during their initialization process.

We start this section by presenting an overview of S-MAC, the pioneer protocol in this class. We continue with a discussion of the other protocols in this class, which are developed based on S-MAC and aim to improve its performance. This is followed by a discussion on a special subclass of synchronized duty cycle protocols that utilize an underline tree topology to decrease the latency caused by the sleep period. In each subtype of protocols, we show either the protocol result in the existence of multi-schedules in the network or assume that all the nodes operate on a common single schedule without elaborating on a way to achieve this.

2.4.1 Overview of S-MAC Protocol

S-MAC [20, 21] is one of the earliest synchronized duty cycle protocols in WSNs. Most of the other synchronized duty cycle protocols are developed based on S-MAC and aim to improve the performance of the protocol. S-MAC defines the basic mechanisms of synchronized duty cycle protocols that enable them to work and conserve network energy in the self-organizing WSN. The basic mechanisms in the S-MAC protocol include periodic sleep and wake up, schedule selection, maintaining synchronization and message passing.

Periodic Sleep and Wake up

In S-MAC, the nodes' operational time is divided into frames. A frame constitutes of an active/wake-up time and a sleeping time, as shown in Figure 2-1. The active time consists of the SYNC period where nodes broadcast their SYNC packets and the DATA period where nodes exchange their RTS/CTS packet. In the sleeping time, nodes that do not participate in data transmission turn their radio modules off during sleeping time. In an *idle frame* where none of the nodes in a neighbourhood contends for a channel, the nodes start the sleeping period after the completion of the active period. In a non-idle frame, the nodes that listen to either an RTS or CTS that is not intended for them invoke an early sleeping period.

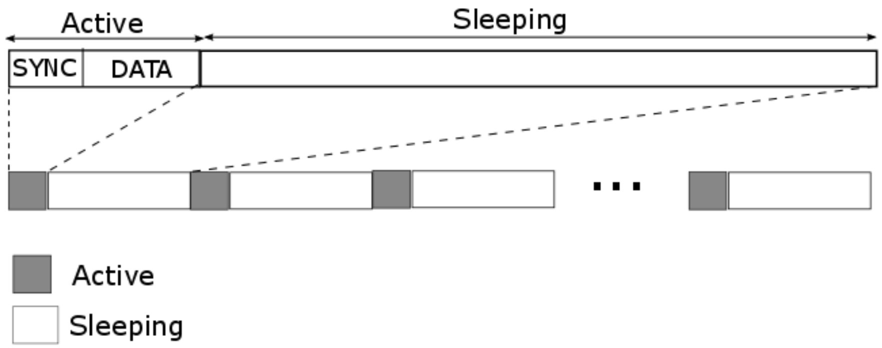

Figure 2-1 Frames in Synchronized Duty Cycle-based Protocols

Schedule Selection

Nodes choose their sleeping schedule during the initialisation process, according to the following rules.

(1) Upon coming to life, a node enters a discovery period where it listens for a broadcasted schedule announcement for a fixed amount of time.

(2) During the discovery period, a node chooses its schedule based on the following two conditions:

  (i) If the node receives a schedule announcement, it adopts the schedule and broadcasts a packet to announce its newly adopted schedule.

  (ii) If the node does not receive any schedule announcement until the end of the discovery period, it randomly generates a new schedule and announces the newly created schedule to the network.

(3) In the condition where a node that has adopted a schedule receives another schedule from its neighbour(s), it makes a decision according to the following two conditions:

  (i) If the node has at least a neighbour that implements its current schedule, it adopts the new schedule as its secondary schedule. The node wakes up during the active periods of its primary and secondary schedule(s), but only announces the primary schedule.

(ii) Otherwise, it adopts the newly received schedule and ignores its current one.

Synchronization Mechanism

S-MAC proposes the SYNC frame and synchronization cycle to maintain the clock and schedule synchronization in a local network. In S-MAC-based networks, a group of ten frames forms a cycle. In each cycle, each node picks a frame randomly as its SYNC frame, where it broadcasts a SYNC packet to announce its schedule. Every ten cycles, a node randomly chooses its synchronization cycle. During the synchronization cycle, a node stays active for the duration of the entire cycle to ensure it receives any sent SYNC in its neighbourhood.

A SYNC packet contains the ID of the sender, the ID of the schedule and the duration until the next wake up in the schedule. Broadcasting a SYNC serves three purposes. First, it announces the existence of a node to its neighbours. In the absence of a SYNC packet from any of its neighbours for a defined time, a node assumes that the neighbour is no longer alive and removes the neighbour from its neighbour table. Second, it corrects the clock drift between nodes implementing a common schedule. In receiving a SYNC from its neighbour operating on the same schedule, a node can determine the time difference between the starts of the next frames of two nodes and adjusts its schedule accordingly. Finally, it notifies the other nodes implementing a schedule which is different from this one. With the purpose of discovering the SYNCs sent by the nodes implementing a different schedule, every ten cycles, the nodes randomly pick a cycle as their synchronization cycle.

Message Passing

S-MAC adopts the mechanism in IEEE 802.11 for transmitting a data packet. During the DATA period in an S-MAC frame, a node that has a data packet to send draws a back-off counter to compete for access to the channel. In a unicast transmission, the sender and the receiver of the transmission participate in an RTS/CTS exchange prior to sending the data packet to reserve the channel. RTS/CTS packets have a field that contains the network allocation vector (NAV) of the transmission, which serves as the virtual carrier-sense for other nodes in the local network. NAV informs the other nodes in the transmission range of the RTS/CTS senders of the duration of the incoming transmission. At the end of a successful data frame in a unicast transmission, the receiver sends an ACK to acknowledge the transmitter.

The simplicity and the fully distributed nature of S-MAC makes it suitable for the conditions of self-organizing WSNs. However, the schedule selection mechanism in this protocol allows the existence of multiple schedules in a network. Although S-MAC claims that this occurrence is expected to be a rare, some later studies [23] prove otherwise. Later sections in this chapter discuss this issue and its effects on the performance of a network in more detail.

2.4.2 Improvements of S-MAC Protocols

Various studies have proposed several improvements to the S-MAC protocol. This section presents significant work in the synchronized duty cycle class that improved the performance of S-MAC.

T-MAC [51] proposes three additional schemes to improve S-MAC, namely, timeout activation (TA), future RTS (FRTS) and full buffer priority. During the active period, if a node does not hear any activity for a duration of TA, it assumes that there will be no packet transmission in the current frame and switches to sleep mode. The duration of TA should be bigger than the total duration of the maximum contention window, RTS transmission and radio switching time. A node sends an FRTS before switching to sleep if it has a packet in its transmission buffer but loses the channel contention to notify its receiver of a delayed transmission. The FRTS contains the duration of the incoming transmission. After the transmission concludes, both the transmitter and the receiver of the FRTS switch to active mode. The transmitter resumes decrementing its back-off counter and sends its RTS. A node with an almost full buffer has a higher priority to send a packet than to receive one. When it receives an RTS, instead of sending the corresponding CTS, the node sends its own RTS, which is guaranteed to win the channel. T-MAC shares S-MAC's tendency of having multiple schedules in the network. The simulation shows that T-MAC consumes less energy than S-MAC, however, as the network load increases, T-MAC suffers more latency and degrading throughput.

DSMAC [41] proposes a way to have dynamic duty cycle based on network load in order to decrease the latency introduced by the sleeping delay in S-MAC. When its latency is larger than a certain threshold, a node doubles its duty cycle by shortening its sleeping time and advertises the change in its SYNC packets. The latency is computed locally in each node by calculating the difference between the time a packet arrives in a node's queue and the time the packet is sent. The neighboring nodes, upon receiving the SYNC packet, also double their duty cycle if they have packets to send. Accordingly, when the latency drops below a certain threshold, the nodes can halve their duty cycle and announce this in their SYNC packets. An upper bound T_E is used to determine the highest duty cycle a node can implement. The simulation shows that DSMAC outperforms S-MAC in delay sensitive applications. However, in order to maintain the dynamic duty-cycle schedule, the node needs to have a common schedule at the start. DSMAC assumes this condition without elaborating further on how to achieve a common schedule in the network.

The DSN-Scheme [52] proposes a dynamic duty cycle scheme in a pseudo-centralized way. In a local network, nodes take turns to act as a designed sensor node (DSN). The DSN maintains a table of the other nodes' schedules and is always active during its DSN duty period. The other nodes in the network query the DSN for the schedule information of their intended receivers if they want to initiate a packet transmission. At the end of its DSN duty, a DSN transfers the schedule table to the new DSN and switches back to a normal node mode.

2.4.3 Staggered Wake-up Schedule Protocols

The previously discussed duty cycle protocols, while significantly saving network energy, in a multi-hop network suffer from additional latency, namely sleeping delay. In a multi-hop packet transmission, an intermediate node cannot forward the packet immediately after receiving it. The node needs to buffer the packet until the next wake up period before contending for the channel. Staggered wake-up schedule protocols propose the use of tree topology to reduce the latency introduced by the sleeping delay. The protocols in this class assume there is an underlying tree along the propagation path with the sink as the root. The protocols require the parent nodes to wake up just before their child nodes are scheduled to send their packet. This approach can significantly reduce the sleeping delay.

DMAC [53] implements a staggered wake-up schedule along the routing path in the data gathering tree with the sink as the root and child nodes report to their parent nodes. A frame in DMAC is divided into sending, receiving and sleeping slots. The slots are designed in such way that a parent's receiving slot coincides with its children's transmitting slot. The goal is to deliver a packet from the leaf node to the root in one transmission frame. Sending and receiving slots have the same length as μ, which covers the total time to transmit packet data and receive an ACK.

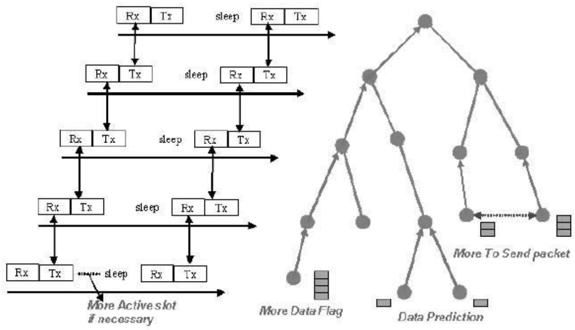

Figure 2-2 Sleeping Schedule in DMAC Gathering Tree [53]

LEEMAC, proposed in [54], aims to improve DMAC by introducing the slot renewal mechanism. The sender embeds the length of its filled buffer in its data packet and the receiver adds additional receiving time accordingly. After sending the first packet, the transmitter switches to sleep mode for the period of 3μ, where μ is the duration of an active period. The period of 3μ covers the transmitting period of its receiver and its receiver's parent (see Figure 2-2). After sleeping for 3µ, the transmitter switches to active mode and sends the remaining packets in its buffer. Using this scheme, the sender can send its all buffered packets after a 3µ sleep instead of having to sleep for 3µ for every additional packet sent.

SPEED-MAC [55] deals with packet contention and the collision problem in a tree-based protocol by having each transmitter send a signal packet before sending its actual packet. SPEED-MAC divides its operational frames into a signalling period (wake up period) and data delivery period (sleep period). In the signalling period, a child node send a signal packet containing the sender's address and a collision bit to its parent if it has a data packet in its buffer. It then stays awake to receive the ACK of the sent signalling packet. If there is more than one child that wishes to send their packets at the same time, the signalling packets collide in their parent node. Upon detecting a collision, the parent node needs to set the collision bit to one and sends the collision signal to notify the network that, currently, there is more than one node competing for the channel. Each node that receives the collision signal has to add an additional active slot for the duration of at least two full transaction times to accommodate the additional packet transmission.

The staggered wake-up protocols have an advantage in terms of smaller latency compared to the other types of protocols. The nodes in this subclass of protocols operate on a single schedule with a different offset, based on their position on the underlying tree topology. However, in a low traffic network, the cost of constructing and maintaining the tree structure causes a large control overhead that results in higher energy depletion in the node.

Table 2-1 summarizes the significant work on CSMA-based energy conservation protocols and their strengths and limitations when implementing them in a self-organizing sensor network.

Table 2-1 CSMA-based Scheduling Protocols

Name /Category	**Special Features**	**Strengths**	**Limitations**
B-MAC[19]/ ADC	Pioneer in the category	Very low energy consumption in a low traffic network.	Long preamble for transmitter, low throughput in a busy network.
Wise MAC[46] / ADC	Control information is piggybacked on data packets	Very low energy consumption. Shorter preamble compared to B-MAC.	Long preamble for transmitter, low throughput in a busy network.
SCP[47] / ADC	Nodes maintain the schedules of their neighbours to reduce the preamble	Very low energy consumption, shorter transmitter preamble.	Low throughput in a busy network.
X-MAC [48] / ADC	Burst of short preamble	Reduces the long preamble. Very low energy consumption.	Low throughput in a busy network.
RI-MAC[49] / ADC	Receiver initiates a transmission	Reduces the long preamble. Very low energy consumption.	Low throughput in a busy network.
PW-MAC[50] / ADC	Pseudo random schedule	Reduces the long preamble. Very low energy consumption.	Low throughput in a busy network.
S-MAC[20,	Periodically sleep and wake	Simple, fully decentralized,	Sleeping delay.

21]/ SDC	up to conserve network energy	low-energy consumption.	Introduces a multi-schedule network.
T-MAC[51] / SDC	Introducing early sleeping (Ta) and Future RTS to reduce latency	Simple, fully decentralized, less energy consumption.	High latency. Introduces a multi-schedule network.
DSMAC[41] / SDC	Dynamic duty cycle. Nodes increase their duty cycle as traffic increases and propagates it to the network	Low latency, adaptable for variable traffic.	Strict synchronization.
PS_MAC[42] (Probability Sensor MAC) / SDC	Dynamic duty cycle. Randomly determines schedule based on pre-wakeup probability (P_i) and seed number ($Seed_i$)	Low latency and higher PDR in a busy network.	Lost synchronization in missing SYNC packet.
DSN-Scheme [52] / SDC	Dynamic duty cycle. DSN stays awake and keeps tab of the schedule information. Nodes take turns to be DSN.	Low overhead of broadcast (flooding) packet.	Big energy consumption for DSN.
DMAC[19]	Staggered schedule.	Low latency.	Controls overhead to maintain tree structure, high contention and collision in a multi-source network.
Q-MAC [56]/ SDC	Downlink traffic (sink query).	Low latency.	Controls overhead to maintain tree structure, extended idle listening.
LEEMAC[54]/ SDC	Slot renewal mechanism.	Lower latency than DMAC.	Controls overhead to maintain tree structure, high contention and collision in a multi-source network.
SPEED-MAC[55]/ SDC	Signalling packet to deal with multi-source network.	Lower packet collision.	Controls overhead to maintain tree structure and signalling packet.

*ADC/SDC: Asynchronous /synchronized duty cycle protocols

2.5 Challenges in Multi-Cluster Wireless Sensor Networks

In this section, we present the main motivation of our study. We start by describing the formation of multi-clusters in a WSNs. We then proceed by discussing the drawbacks of a multi-cluster network in relation to network performance. Finally, we discuss the existing work that investigates this issue.

2.5.1 The Formation of a Multi-Cluster Wireless Sensor Network

In the early phase of a randomly deployed wireless sensor network, the nodes come to life at slightly different times. In this stage, the network consists of several disconnected partitions of nodes. As more nodes become active and join the network, the partitions are gradually connected to form a network. A node, upon coming to life, starts a discovery period and listens for an advertised schedule. If it does not hear any announcement, at the end of the discovery period, it assumes that it is the first operating node in its neighbourhood and creates a new schedule, as shown in Figure 2-3.

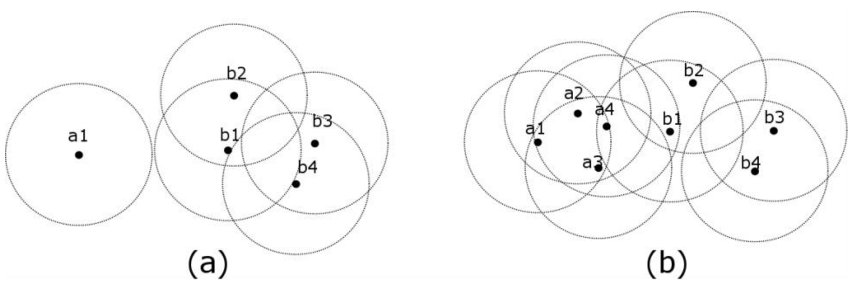

Figure 2-3 The Formation of the Multi-Cluster Network

In Figure 2-3, node a_1, during its initialisation process, listens for a schedule announcement. Even though there are already some operating nodes in the network (b_1-b_4), which implement a schedule (S_B), node a_1 cannot hear their schedule advertisements since they are out of a_1's receiving range. At the end of its initialisation process, node a_1 creates a new schedule S_A and periodically advertises the schedule. At this time, there are two schedules in the network. Over time, more nodes, located in a_1's neighbourhood, join the network and adopt schedule S_A. Nodes that follow S_A form a virtual cluster *A* and nodes that follow S_B form a virtual cluster *B*. As node a_4 comes to life, clusters A and B 'meet' and nodes a_4 , and b_1 become the *border nodes* of the two clusters. The border nodes are the nodes that are located near the border of two or more clusters and act as the gateway between the clusters. These nodes need to ensure that the other nodes in both clusters can communicate with each other even though they operate on different schedules.

Studies show that the occurrence of multiple clusters in a network is quite common and increases as the dimension of the network increases [23, 24]. The simulation in [23] shows that in a 10R*10R network where R is the transmission range of a node, up to 50% of the nodes follow more than one schedule.

2.5.2 Problems with Multi-Cluster Wireless Sensor Networks

As discussed in the previous section, the border nodes act as the gateway between their bordered clusters to enable packet transmission between the clusters. S-MAC proposes the following two ways to accomplish this purpose:

(1) Each border node needs to wake up in the active time of all of its bordered clusters to ensure that it can receive the packet transmission from the other cluster. This means that the node adopts the other clusters' schedules on top of its cluster schedules, which leads to a problem called the bottleneck problem.

(2) Each border node stores the information of all its bordered clusters’ schedules, but only needs to operate on its own cluster schedule. If the node needs to send a packet to another cluster, it looks up the cluster's schedule on its storage and wakes up during the active time of the cluster only during the frame it intends to send the packet. This leads to another problem called the hidden terminal problem.

The Bottleneck Problem

This problem arises when the border nodes adopt the schedule(s) of its bordered clusters in order to serve as the gateway between the clusters. Nodes that adopt multiple schedules have higher duty cycles, hence becoming the bottleneck of the network and consequently, they run out of energy much sooner than the other nodes as shown in the simulation result in [22, 23]. We propose two possible scenarios of how the border nodes prematurely leaving the network affect the performance of the network as a whole.

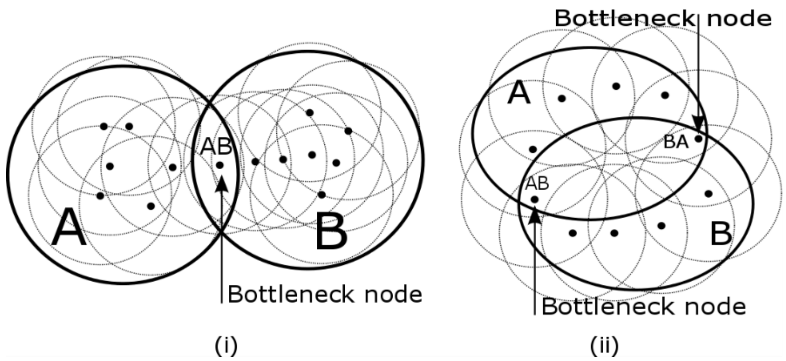

Figure 2-4 The Bottleneck Problem in a Multi-Cluster WSN

1^st^ Scenario: Network Splitting - In Figure 2-4(i), the nodes in cluster A implement a schedule called S_A and the nodes in cluster B implement a different schedule called S_B. Previously disconnected, the two clusters find each other when node AB comes to life and adopts both schedules; hence, AB has an active time up to twice the other nodes in the network. Consequently, node AB exhausts its battery much sooner than the other nodes. Being the gateway between the two clusters, when node AB is down, the network is fragmented into two disconnected partitions.

2^nd^ Scenario: Increase of Hop Count - The second scenario is shown in Figure 2-4(ii). Node AB and BA are the border nodes of the two clusters. If node AB uses up its energy and leave the network, the two clusters are still able to communicate with each other. However, it doubles the maximum hops between nodes. This leads to both bigger packet latencies and more energy consumption to send the packet through a longer route.

In both scenarios, the border nodes hold critical roles in the network, which means that the performance of the system significantly drops when they run out of their energy. A study in [23] ran a simulation to compare the lifetime of a multi-cluster network and single cluster network. It shows that a single cluster network has on average a 60% longer life span than a multi-cluster network.

The Hidden Terminal Problem

This problem arises when the border nodes only wake up during the active time of its schedule if they do not send a packet to another node in a different cluster. Consequently, the border nodes have at least one neighbour with a different active period.

In wireless networks, the IEEE 802.11 family protocols use the CSMA/CA scheme with the RTS/CTS technique to avoid data packet collisions in the network. This system works well in most wireless network applications since all nodes are always active during their operational time. In WSN, the condition is slightly different. Having their radio modules off for some portions of the time, the nodes might miss their neighbours' RTS packets.

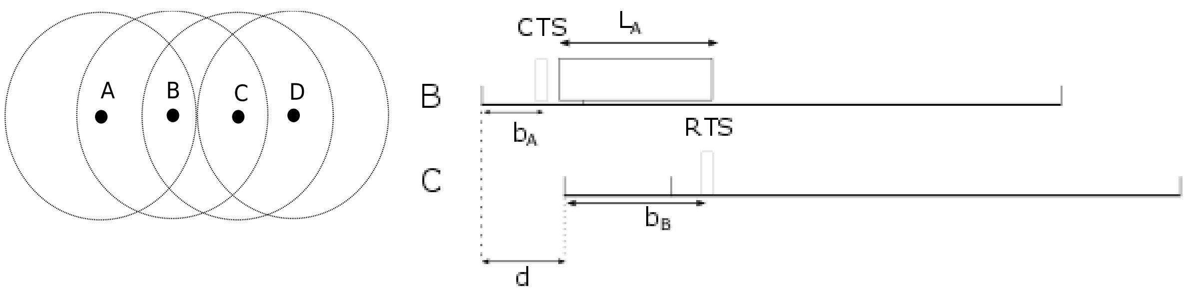

Figure 2-5 The Hidden Terminal Problem in the Multi Sleeping Schedule WSN

In Figure 2-5, nodes *A*, *B*, *C* and *D* form a chain topology network. Nodes *A* and *B* belong to a common cluster and nodes *C* and *D* are members of another cluster, thus, nodes *B* and *C* are neighbouring nodes that implement different sleeping schedules with a certain *offset* (*d*). We consider a scenario where node *A* has a packet to send to *B*. Nodes *A* and *B* wake up at the same time and exchange RTS/CTS packets. Because they both participate in data transmission, they do not turn their radio module off during the sleeping period, they instead start their data packet transmission. During transmission, node *C* wakes up. Since it does not hear the transmission from *A* to *B*, if it has a packet in its buffer, it contends for the channel and send its RTS. A collision of node *A*'s data packet and node *C*'s RTS packet then occurs in *B* and causes energy waste without both *A* and *C* being aware of the condition.

2.5.3 Existing Global Sleeping Schedule Protocols

Some studies have recognized and investigated the problems of multi-cluster networks. A study in [23] proves that the occurrence of multiple schedules in S-MAC is not as infrequent as assumed in [20, 21]. It simulates a 1 km square network in which nodes are deployed randomly. Each node has in average *k* neighbours and transmission range of 0.2 km. The simulation results show that the larger the *k*, the greater the percentage of nodes following multiple schedules. Moreover, the smaller the transmission range, the larger the percentage of nodes following multiple schedules. In some cases in the simulation, approximately 50% of the nodes following multiple schedules and a few nodes follow up to four schedules.

A study in [24] proposes a modification to S-MAC by introducing two new algorithms called the Global Sleep Algorithm (GSA) and Fast Path Algorithm (FPA). GSA aims to achieve a single global schedule in the network by having age as the criterion for selecting the winning schedule when a node hears the announcement of different schedules. Nodes advertise the age of their schedule in their SYNC packets and upon listening to a new schedule, the nodes select the oldest schedule. The same study also introduces the fast path algorithm (FPA) to reduce the latency caused by the sleeping delay in a multi-hop network. FPA adds an additional wake-up period into the regular sleeping schedule along a multi-hop routing, so that the next node wakes up when the current node is ready to send. Nodes setup the routing path by piggybacking the path setup message in the first packet in the path.

Similar to [24], S-MACL[22, 23] modifies S-MAC by introducing a scheme for the global sleeping schedule to avoid fast energy depletion in nodes that adopt more than one schedule. In receiving a different schedule from its neighbour, a node compares the schedules' IDs and follows the one with the smaller identifier. The node then announces its newly adopted schedule during the listening time of both its new and old schedule.

Both studies in [22-24] acknowledge the advantage of having a global schedule in the network and propose a way to achieve this. However, both the proposed algorithms have a limitation that can severely affect the performance of a network. Using a schedule ID as a sole winning criterion does not promote efficient merging in a network. Moreover, it could lead to a fatal condition where two different schedules with the same schedule ID propagate through the network in a rechargeable network. Using schedule age as the winning criterion introduces a huge overhead in the control packet.

2.6 Research Questions

As discussed in the previous sections, the schedule selection mechanisms in the existing duty cycle protocols of WSNs result in the possibility of the formation of multi-cluster networks. In a multi-cluster network, the border nodes suffer from either the bottleneck problem or the hidden terminal problem, which in the end, adversely affects the whole network performance. To address this issue, this study aims to develop a protocol to improve the energy saving mechanism in self-organizing WSNs. In order to achieve this aim, we develop a mechanism that enables the nodes to reach a consensus decision for a global sleeping schedule in the network in a distributed manner. We break down the tasks in the study into the following research questions (RQs):

RQ 1: How to evaluate the performance of the existing synchronized duty cycle protocols and how having multiple schedules in the network contributes to the energy wastage in the network?

We address this question in Chapter 3 by answering the following sub-questions

RQ 1.1. *How to analytically evaluate the performance of the existing synchronized duty cycle protocols?*

We develop a Markov model that represents the properties of synchronized duty cycle protocols. The states in our Markov model represent (1) the contention states of a node, ranging from zero to the contention window (*CWND*-1) and (2) the idle state, representing the condition where the node has an empty transmission buffer. Based on the Markov model, we derive the equation to measure the collision probability and throughput in the network.

RQ 1.2. *How to extend the model in RQ1.1 to evaluate the effects of multiple schedules in the network on the additional energy waste in the network?*

We answer the question by deriving an equation to measure the data packet collision rate in the border nodes. We then derive the equations to measure the energy wastage due to collisions in multi-cluster and single-cluster networks.

RQ 1.3. *How to validate the mathematical models in RQ 1.1. and RQ 1.2?*

We run extensive simulations in the MATLAB 2014 environment to validate the models in RQ 1.1. and RQ 1.2.

RQ 2: How to develop a distributed protocol to achieve a global sleeping schedule in the self-organizing environment and how to measure the performance of the protocol?

We consider the ideal properties of an energy saving protocol in the MAC layer to work in a self-organizing WSN with no pre-configuration and global time reference. We then define the main challenges in developing a global schedule protocol in a self-organizing environment, namely, (1) the schedule synchronization drift and (2) the merging efficiency.

In our global schedule protocol, we propose the offset between two schedules, i.e. the difference of the times of the next sleep in two schedules as the winning criterion. In developing the protocol, we need to respond to the following sub-questions.

RQ 2.1. *How to determine the upper bound of synchronization drift in a network?*

This question is related to the first challenge in developing a global sleeping schedule protocol. In a synchronized duty cycle-based network, the nodes broadcast SYNC packets to announce their schedules. The propagation delay of the SYNC packets and the lack of a global time reference cause the receiving nodes to perceive the start of the next frame of a schedule slightly behind the actual one. This condition is called schedule synchronization drift. Since a propagation delay is a function of the distance between the transmitter and the receiver, in calculating the upper bound of the synchronization drift in a network, we derive the maximum propagation path length in the network. We use the space filling curve mechanism to derive the theoretical upper bound of the propagation path length. We then develop a simulation to obtain the maximum hops between the two furthest nodes in a randomly deployed network, based on the size and node density of the network. Finally, we derive an equation to estimate the relation between the network size and the maximum propagation path length in a dense network.

RQ 2.2. *How to develop a schedule selection and merging algorithm for a global schedule network in a distributed environment?*

This question is related to the second challenge in developing a global sleeping schedule protocol, which relates to merging efficiency. To achieve a smaller convergence time, we develop a cluster-merging algorithm in which the nodes in a cluster merge at the same time, as opposed to the individual node-merging scheme proposed in the existing global schedule protocols.

RQ 2.3. *How to measure the performance of the proposed algorithm compared to the existing global sleeping schedule algorithms?*

We use the convergence time and the energy spent in sending the control packets during the convergence time as the performance parameters of our proposed scheme. We derive mathematical equations to estimate the performance parameters and validate these using simulations. We then compare the results of our proposed scheme to the schemes proposed in the existing global sleeping schedule protocols.

2.7 Summary

In this chapter, we presented a broad overview of important topics and work in relation to the energy conservation problem in WSN research. As discussed, energy conservation poses a crucial problem in WSNs, especially those with a self-organizing nature. We reviewed the existing studies and protocols developed to tackle this problem and discuss their performance in self-organizing WSNs. Due to the focus of this research, we presented a more detailed review of the MAC layer energy-efficient protocols, specifically, the sleeping schedule algorithm in synchronized duty cycle protocols, as discussed in section 2.4.

As shown in section 2.5, the duty cycle-based protocols lead to the formation of a multi-cluster network, which potentially degrades the performance of the network. Although there are already some preliminary studies in this field, some of the crucial challenges that prevent the protocols from working in self-organizing WSNs have not been thoroughly discussed. Further, in this section, we discussed the potential problems of a multi-cluster network, namely the hidden terminal problem and the bottleneck problem, that can adversely affect the performance of the network. This serves as the main motivation of our study.

Having discussed the state-of-the-art of the existing energy saving protocols in WSNs, their limitations and the potential problems, in section 2.6, we presented two main research questions, which were divided into several sub-questions. The questions outline the specific tasks in the study to find a solution to the problem stated in section 1.3. As a response to the first research question, in Chapter 3, we investigate the impact of the hidden terminal problem in a multi-cluster network on the performance of the network. We conduct this task by developing an analytical model and validating the model using simulation. Chapter 4 responds to the second research question. In Chapter 4, we present the development of a global schedule algorithm with cluster merging.

Chapter 3 Modelling and Performance Analysis of Synchronized Duty Cycle-Based Wireless Sensor Networks

3.1 Introduction

There are three common ways to evaluate the performance of a computer network protocol, namely test bed simulation, computer program simulation and a mathematical model. The mathematical model outperforms the other methods in that it provides the measurement of the steady state performance of a network. Generally, either a test bed or a computer program simulation is used along with a mathematical model to validate the model. In this chapter, we present the development of an analytical model to analyse the performance of the synchronized duty cycle-based protocols as a response to research question 1 (RQ 1). We validate the model by running an extensive simulation in the MATLAB 2014 environment for various sets of conditions.

Wireless sensor networks (WSNs) have specific characteristics that distinguish them from other wireless network applications. Firstly, between the sensing processes, there is no data packet generated in a sensor, thus WSNs mostly operate under low traffic network conditions and the nodes are, most of the time, in an idle condition. Secondly, in most WSNs, energy efficiency takes priority over the quality of service (QoS); hence, measuring the energy waste in a network is one of the key features in performance evaluation. Thirdly, related to the second reason, to conserve energy, WSN protocols require the nodes to turn off their radio modules periodically. Both the model and the simulation used to measure the performance of WSNs need to consider all these characteristics.

As discussed in Chapter 2, the synchronized duty cycle-based protocols allow the nodes to generate their own sleeping schedules if they fail to discover any existing ones in their initialisation processes. This could lead to the existence of the multi-schedule network, where the nodes that operate on a common schedule in a neighbourhood form a cluster. The nodes located near the borders of two or more clusters have one or more neighbours that operate on different sleeping schedules. The condition where neighboring nodes sleep and wake up at different times introduces the hidden terminal-like problem in WSNs that the RTS/CTS mechanism in the IEEE 802.11 protocol does not anticipate.

The work in this chapter has two main contributions. Firstly, we develop a Markov model to analyse the performance of the synchronized duty cycle-based protocols. Secondly, we extend the mathematical model to analyse the performance degradation due to the hidden terminals in multi-cluster networks. To validate the model, we develop a simulation using strict schedule timing and compare our simulation results with the analytical model.

The rest of this chapter is organized as follows. In section 3.2, we present the existing work on analytical models to evaluate the performance of wireless networks. Some of the work was developed to evaluate the performance of IEEE 802.11 networks and the others were developed for specific types of WSNs. We highlight the contributions of each study and point out the reasons why neither of them fully represent the characteristics of a synchronized duty cycle-based WSN. In section 3.3, we present our proposed analytical model. We start by listing the properties of the proposed model. We then present the Markov model of the system and the equations derived from the system. In section 3.4, we extend the model in section 3.3 to calculate the additional energy wastage in the border nodes of a multi-cluster network caused by data packet collisions. In section 3.5, we run an extensive simulation for validating the model and analyse the result of the simulation. Section 3.6 summarizes the chapter.

3.2 Existing Analytical Models of Wireless Networks

The work in modelling the performance of a wireless network was initiated by the pioneering efforts of Bianchi. In [57], he proposes a two-dimensional Markov model to evaluate the throughput of the distributed coordinated function (DCF) in IEEE 802.11 networks [18] under a saturated condition. Each state in Bianchi's model represents the retransmission stage and the back-off counter of a node at time t. From the model, he derives the collision probability, the successful transmission probability and the network throughput. The accuracy of the analytical model is validated against the simulation results. Since Bianchi's model was proposed, a number of authors have modified the model to suit different network conditions.

The work of Ma and Chen in [58] and Wang et al. in [59] propose some modifications to Bianchi's model to evaluate the performance of the broadcast traffic. Similar to the work in [57], Ma and Chen's model also assumes an underlying saturated traffic condition. The work introduces the term of consecutive freeze process (CFP), which is a condition where a station consecutively gets access to the channel by choosing a zero back-off counter. CFP more severely affects the broadcast transmissions than the unicast transmissions. In unicast traffic, CFP only occurs after a successful transmission. Due to the lack of a packet acknowledgment mechanism in a broadcast transmission, either a successful transmission or a failed/collided transmission could lead to CFP. To avoid this condition, the model prohibits a zero initial back-off counter. Wang et al. in [59] propose two modifications to Ma and Chen's model. Firstly, unlike the previous two models which were designed for a network under a saturated condition, Wang et al. include the probability of having a new packet to send (q) in their model. When the value of q is equal to 1, the network enters a saturated condition. Secondly, in IEEE 802.11 networks, in detecting a busy channel, all the contending nodes freeze their back-off counters. The work in [57, 58] fail to incorporate this condition in their models. Wang et al.'s model, however, considers the probability of freezing the back-off counter due to a busy channel.

The previously discussed models assume a network with an ideal channel, where there is no hidden terminal. The studies in [60] and [61] analyse the performance of a wireless network in the presence of hidden terminals. The work in [60] models two stations that are hidden from each other but connected to a common access point (AP). The work presents the condition in a two-dimensional Markov model, in which each state represents the retransmission stage in both stations. The packet collisions in AP occur if the difference of the back-off counters of the two hidden stations is less than the vulnerable time T_V. In the basic access scheme, the vulnerable time equals the duration of sending a frame and in the RTS/CTS scheme, it equals the duration of sending an RTS. The study in [61] presents a three-dimensional Markov model to investigate a similar problem. Both studies assume the network operates under saturated traffic.

Several works have intended to develop accurate analytical models for WSN [62-64]. The studies in [63] and [64] proposed a three-dimensional Markov model to calculate the performance of ZigBee-based WSN. The characteristics of the models, however, do not represent the characteristics of the duty cycle protocols. The model in [62] analyses the performance of synchronized and asynchronous duty cycle WSN. In the model, each state represents the queue size in a node. The model tackles the contention process as a black box process, in which the probability of winning the channel and the probability of the successful transmission of a node solely depends on the random back-off counter the node chooses before contending for the channel. The model assumes that if a node is loose in the channel contention in a frame, the node needs to randomly re-draw the back-off counter in the next frame. This does not comply with the DCF mechanism in the IEEE 802.11 protocol, where instead of redrawing the back-off counter, after losing a channel contention, a node simply freezes its counter. The node then resumes decrementing its counter the next time the channel is sensed to be idle.

To our knowledge, none of the papers discussed above capture the properties that fully reflect the nature of synchronized duty cycle protocols. Table 3-1 summarizes the existing performance analysis models in the wireless network and their limitation in evaluating the performance of the synchronized duty cycle-based WSN. In the next section, we address this gap and propose an analytical model for synchronized duty cycle protocols for single and multi-cluster WSNs. Section 3.3.1, in particular, shows how we incorporate the properties of the previously proposed models into our proposed model.

Table 3-1 Existing Performance Analysis Models

Study	Focus	State	Feature	Limitation*
Bianchi Model [57]	Throughput of saturated network	Back-off counter, retransmission stage	Basic access, RTS/CTS mechanism	WSN, in general, is a low traffic (unsaturated) network
Ma & Chen [58]	Throughput of saturated network	Back-off counter	Broadcast service	Idem [57] Some transmissions in WSN are unicast services
Generic Model [59]	Throughput of unsaturated network	Back-off and counter Idle State	Broadcast service	Idem [44]
[60]	Saturation throughput, Hidden terminal	Retransmission stages in 2 hidden terminals	Basic access, RTS/CTS	Idem [57] Scalability**
[61]	Throughput, collision, hidden terminal	Retransmission stage, back-off counter, remaining time	Basic access, RTS/CTS	Idem [57]
[62]	Throughput of	Buffer size	S-MAC and X-	Simplified access

	duty-cycled MAC		MAC	mechanism***
[63] and [64]	Performance prediction, configuration	Back-off stage, back-off counter, retransmission stage	ZigBee IEEE 802.15.4	Complexity

*Against the properties of synchronized duty cycle in WSN
** The study develops a two-dimensional Markov model in which each state represents the retransmission stage in each station. Adding more stations to the model increases the dimension and the complexity of the model.
*** The probability of winning the channel and the probability of successful transmission solely depends on the random back-off counter a node choose before contending for the channel. What happens during the contention or in the last round of transmission, i.e. the other nodes freeze their back-off counter decrementing process in detecting a busy channel and resuming it when the channel is idle, is completely neglected

3.3 The Analytical Model

3.3.1 The Model Properties

Synchronized duty cycle-based WSNs have certain distinct characteristics that differentiate them from the other wireless network applications. Nodes in WSN mainly operate with low traffic load. Moreover, nodes in WSN operate with their radio module off for parts of their operational time. The mathematical model proposed in this study has the following properties:

(1) The Markov states in the proposed model represent (a) the back-off counters in a node similar to the models in [57-59] and (b) the idle state where the node does not have any packet in its buffer similar to the model in [59].

(2) The model assumes the buffer size in each node is one similar to the models in [59, 63, 64] because of the low traffic characteristic of WSN.

(3) The model considers the probability of packet collision and retransmission in the unicast traffic.

(4) In the duty cycle-based networks, such as S-MAC[20, 21], the nodes use DCF mechanism to compete for the channel access. The losing node switches to sleep after freezing its counter and resumes decrementing its counter in the next frame. The model strictly implements the DCF mechanism in the IEEE 802.11 protocol, in which a node freezes its back-off counter when another node wins the channel contention.

(5) The model incorporates the duration of the sleeping period in calculating the time spent in a contention state due to a busy channel and the time spent in an idle state due to an empty transmission buffer.

(6) The model incorporates the probability of having a data packet collision in addition to the usual RTS/CTS packet collision caused by hidden terminals in the multi-cluster network.

3.3.2 Analytical Model for Single Cluster Network

Let us assume a network with n nodes contending for a channel. The stochastic model includes the contending states $\{b_k\}$, which is the stationary distribution of back-off states k and b_I , which is the stationary distribution of the idle state. We assume that the packets are generated according to the Poisson distribution, i.e., a node only sends a reporting packet in the case of a detection of certain events.

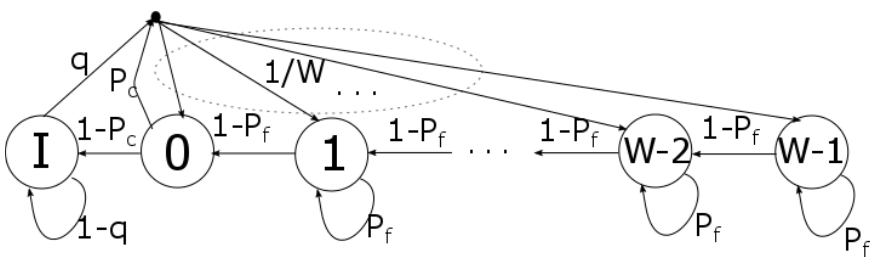

Figure 3-1 Markov Model for WSN Node

The Markov model in Figure 3-1 is developed based on the following assumptions:

(1) Each station has a buffer size of one and packets arrive in each node according to the Poisson distribution with arrival rate λ.

(2) Each node has CWND size of W and before sending its packet, a node randomly picks a back-off counter between [0 ... W-1].

(3) In contending states, a node with the probability of P_f needs to freeze its back-off counter decrementing process due to the busy channel for the duration of T_f.

(4) A node can only transmit its packet when the back-off counter is zero. After a successful transmission, a previously transmitting node switches to idle state and following a collided transmission, a node withdraws a new back-off counter for the same packet.

Table 3-2 summarizes the notations used in the analytical model

Table 3-2 Notation Used in Analytical Model

Notation	Description
0 … W-1	Back-off counter state
I	Idle state
P_f	Probability that the channel is busy at the given slot (node needs to freeze its back-off time)
P_c	Probability of collision
q	Probability of having a new packet
W	Number of slots in the contention wind
σ	The duration of an empty slot
T_b	The duration of a busy slot
T_I	The duration of an idle slot
T_v	The duration of a virtual slot
N	Number of nodes in 1 hop network (Network density)

Equation (3-1) shows the only non-null one-step transition probabilities in the Markov model in Figure 3-1.

$$\begin{cases} \boldsymbol{P\{I|0\} = 1 - P_c} \\ \boldsymbol{P\{k|I\} = \frac{q}{W}, \quad 0 \le k \le W-1} \\ \boldsymbol{P\{k|k\} = P_f, \quad 1 \le k \le W-1} \\ \boldsymbol{P\{k|0\} = \frac{P_c}{W}, \quad 0 \le k \le W-1} \\ \boldsymbol{P\{k|k+1\} = 1 - P_f, \quad 0 \le k \le W-2} \\ \boldsymbol{P\{I|I\} = 1 - q} \end{cases} \tag{3-1}$$

Deriving the global balance equation for the idle state in the Markov model, we can express the stationary probability of being in idle state, b_I, as

$$b_I = b_0 P(I|0) + b_I P(I|I)$$

Thus

$$b_I = \frac{b_0 P(I|0)}{1 - P(I|I)}$$

Substituting *P(I|0)* and *P(I|I)* in equation (3-1), we have

$$b_I = \frac{1 - P_c}{q} b_0 \tag{3-2}$$

Let the stationary distribution of the Markov model be $b_k = lim_{t\to\infty} P(b(t)) = k$, for $1 \le k \le (W\text{-}2)$, P_f be the probability of a busy channel and P_c be the probability of a packet collision. Deriving the global balance equation for any b_k, we have

$$b_k = b_{k+1}(P(k|k+1) + b_k(k|k) + b_0(P(k|0)) + b_I\big(P(k|I)\big)$$

$$b_k = b_{k+1}\big(1 - P_f\big) + b_k P_f + b_0 \frac{1}{W}$$

i.e.

$$b_k\,(1 - Pf) = b_{k+1}\big(1 - P_f\big) + b_0 \frac{1}{W}$$

i.e.

$$b_{k+1} = b_k - \frac{b_0}{W(1 - P_f)}$$

This implies

$$b_k = b_0 - \frac{k b_0}{W(1 - P_f)}$$

Deriving the global balance equation of state b_{W-1}, we have

$$b_{W-1} = b_{W-1} P(k|k) + b_0 P(k|0) + b_I P(k|I)$$

Thus

$$b_{W-1} = \frac{b_0 P(k|0) + b_I P(k|I)}{1 - P(k|k)}$$

Substituting equation (3-1), we have

$$b_{W-1} = \frac{1}{W(1-P_f)} b_0 \qquad (3\text{-}3)$$

The probability of transmitting is the probability of being in state 0, b_0. According to the probability theory

$$b_I + b_0 + \sum_{k=1}^{W-2} b_k + b_{W-1} = 1$$

Substituting equations (3-1), (3-2) and (3-3), we have

$$\left(\frac{1-P_c}{q} + 1 + \frac{W-1}{2(1-P_f)}\right) b_0 = 1$$

Let τ be the probability of a node transmitting in a virtual slot time, then we have τ as

$$\tau = \frac{1}{\frac{1-P_c}{q} + 1 + \frac{W-1}{2(1-P_f)}} \qquad (3\text{-}4)$$

When a node is contending, it needs to freeze its back-off counter whenever it senses the channel is busy. We derive the freezing probability P_f in a virtual slot as the probability of at least one of any remaining nodes is transmitting in the given virtual slot. Therefore, for a neighbourhood of n nodes, we have equation (3-5) as

$$P_f = 1 - (1-\tau)^{n-1} \qquad (3\text{-}5)$$

The probability of a successful transmission, P_s, in the neighbourhood of n nodes equals the probability that exactly one node transmits in a given slot, given the condition that the channel is busy. We have Equation (3-6) as

$$P_s = \frac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n} \qquad (3\text{-}6)$$

A collision occurs when more than one node transmits their packets at a same slot time. Hence, we have the probability of collision in equation (3-7) as

$$P_c = 1 - (1 - \tau)^{n-1} \quad (3\text{-}7)$$

Figure 3-2 shows the collision probability in a network for various packet arrival rates and network densities

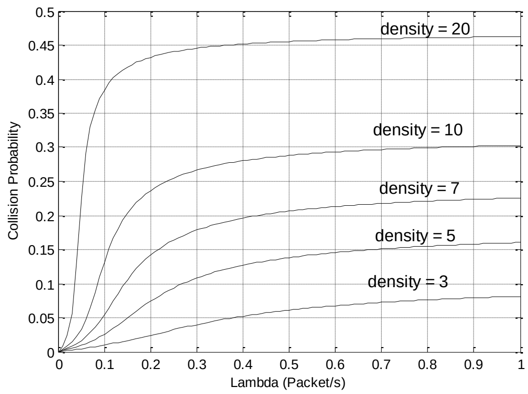

Figure 3-2 The collision probability in a network based on its density (n) and packet arrival (λ)

In a non-saturated channel, for parts of the operational time, a node is in an idle state when it does not have a packet to process. Assuming that packets arrive in a node according to the Poisson distribution with inter-arrival rate ($1/\lambda$), the probability of packet arrival in a slot with a duration T_v is as shown in Equation (3-8)

$$q = 1 - e^{-\lambda T_V} \quad (3\text{-}8)$$

A virtual slot in the model is the average of the time spent in one of the states in the Markov chain. In the contending state $\{b_k\}$, the virtual slot includes an empty slot for an idle channel and a freezing slot (transmission or collision slot). The empty slot σ is a fixed duration between two consecutive back-offs during an idle channel. The freezing slot T_f is the duration that a node needs to freeze its back-off counter during a busy channel. The duration of the freezing slot equals the sum of the duration between the time the node senses the channel is busy until the start of the RTS/CTS period in the next frame. The duration of the virtual slot during an idle state (T_I) is the duration of an entire frame. Therefore, for a node with W contending state and one idle state, we have the average duration of a virtual slot T_v in equation (3-9) as

$$T_v = \frac{W}{(W+1)}((1-\tau)^n\sigma + (1-(1-\tau)^n T_f)) + \frac{1}{(W+1)}(1-q)T_I \tag{3-9}$$

The throughput of a network is defined as the rate of successful transmission per second and given in equation (3-10) as

$$s = \frac{n\tau(1-\tau)^{n-1}}{T_v} \tag{3-10}$$

The solution of the system is formed by five unknown variables, τ, P_c, P_f, P_I, and q.

3.3.3 Analytical Model of a Multi-Cluster Network with Hidden Terminals

In an IEEE 802.11 network, a packet collision can be grouped into the RTS collisions and data packet collisions. RTS collision happens when two or more stations send their RTS at the same time slot. Packet collision occurs if a node misses the RTS/CTS exchange of its neighbours and sends its own RTS during a packet transmission.

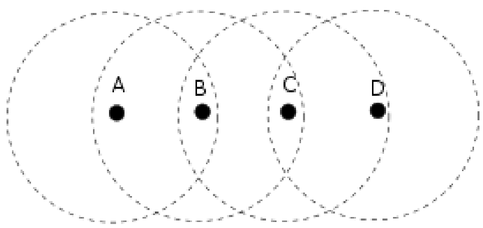

Figure 3-3 A Simple Two-Cluster Network

To investigate the data packet collision in a multi-cluster network, we consider a simple two-cluster network with four nodes that are connected in a chain topology, as shown in Figure 3-3. Cluster 1 consists of nodes *A* and *B*, and cluster 2 consists of nodes *C* and *D*, thus, nodes *B* and *C* are the border nodes of the two clusters. The border nodes follow their cluster schedule unless they are sending a packet to the other cluster. Since they sleep and wake up at different times, nodes *B* and *C* could miss each other's RTS/CTS packets, which could result in packet collision.

Figure 3-4 shows the timing diagram of a collided transmission in the network. We consider a case where node *A* intends to send a packet to node *B* and node *C* intends to send a packet to node *D* in two intersected frames. Node *C* wakes up after node *A* and *B* and finishes their RTS/CTS exchange in the middle of the data packet transmission. After completing decrementing its back-off counter, node *C* sends its RTS. Node *C*'s RTS packet then collides with node *A*'s transmitted data packet in node *B*.

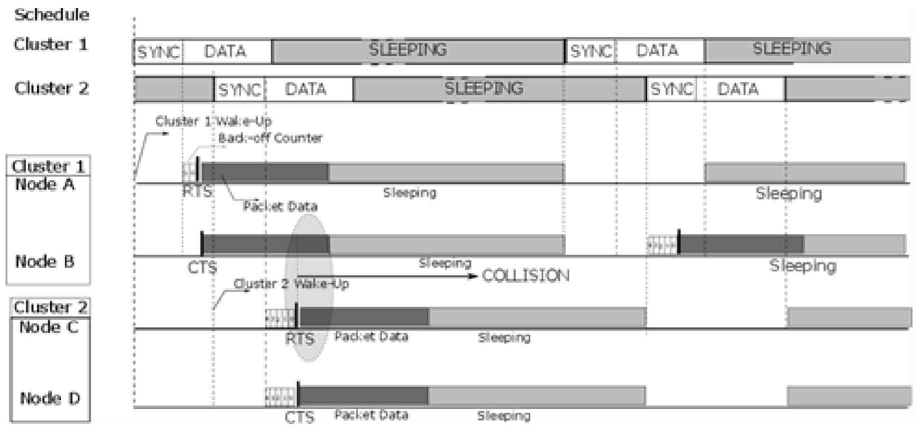

Figure 3-4 Time Diagram Collided Transmission in a Two-Cluster Network

Proposition 1 *Let the network be a single cluster network with n nodes contending for the channel. Then the probability of having an RTS collision is*

$$P_{C_{RTS}} = 1-(1-\tau)^{n-1} \quad (3\text{-}11)$$

where τ is the probability of a node winning the channel. The probability of having a data packet collision (P_{c_Data}) is 0.

Proof.

1) The probability of RTS collision equals P_c in equation (3-7) as proven in section 3.3.2.
2) In the single cluster network, the nodes start the active period at the same time with a typical drift in order of 10^{-6} s [22]. At the beginning of DATA duration, all nodes in the network are already in the active state. Since all the nodes can hear any ongoing RTSs, the probability of data packet collision is 0.

Proposition 2: *Let a network be a two-cluster network, where every node has (n-1) neighbours. Let each of the two clusters have exactly one border node that connects the clusters. Then we have the probability of RTS collision*

$$P_{C_{RTS}} = 1-(1-\tau)^{n-2} \quad (3\text{-}12)$$

the probability of data packet collision with packet transmission duration L_{packet} is

$$P_{C_{DATA}} = \frac{\frac{1}{n-1}\tau L_{packet} P_s}{T} \quad (3\text{-}13)$$

Proof.

(1) The probability of RTS collision is the same as the one in a single cluster with the density of *(n-1)* nodes.

(2) Let *x* be a border node of cluster *X* that has exactly a neighbour *y* in cluster *Y*.

(3) Let every node in the network have in total *n* neighbours, and both nodes *x* and *y* have exactly *(n-1)* neighbours in their clusters respectively, and all of their (*n*-1) neighbours are inner nodes. The probability of a node sending a successful RTS in a single cluster in the cluster is P_s as derived in Section 3.3.2. Equation P_1 gives the probability of *x* being a receiver of a data packet transmission

$$P_1 = \frac{1}{(n-1)} P_s$$

(4) Let cluster *Y* wake up just after the RTS/CTS exchange in node *x* concludes and L_{packet} be the duration of data packet transmission in node *x*. Then the probability of node *y* sending its RTS during the data transmission in node *x* is proportional to L_{packet} as seen in Figure 3-5.

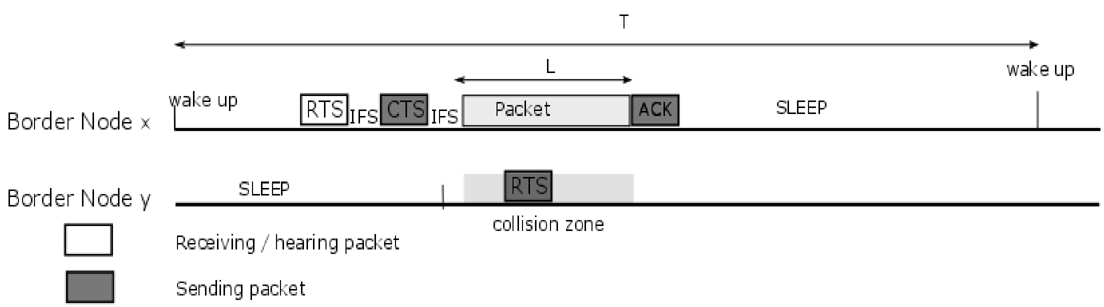

Figure 3-5 Packet Data Collision in Border Nodes

Let P_2 be the probability of the RTS packet of node y colliding with the data packet received by node x, τ is the probability of node y sending an RTS and $\frac{L}{T}$ is the probability of the RTS being sent during the collision zone. Then we have

$$P_2 = \frac{\tau L}{T}$$

(5) Data packet collision happens when a border node receives a data packet and its neighbour, which is a member of a different cluster, sends its RTS during the data packet transmission.

$$P_{C_{DATA}} = P_1 P_2 = \frac{\frac{1}{n-1}\tau L P_s}{T}$$

Proposition 3: *Let us generalize the previous network to an m-cluster network, where every node has (n-1) neighbours. Let a cluster X have a node x that has (m-1) neighbours that are members of different clusters as shown in* Figure 3-6.

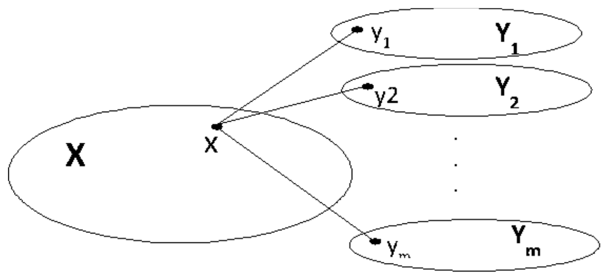

Figure 3-6 A multi-cluster network

Then we have the probability of RTS packet collision in x

$$P_{C_{RTS}} = 1-(1-\tau)^{n-m} \qquad (3\text{-}14)$$

The rate of data packet collision with packet size L_{packet} is

$$P_{C_{DATA}} = \frac{1}{n-1} P_s \left(1-\left(1-\frac{L}{T}\tau\right)^{m-1}\right) \qquad (3\text{-}15)$$

Proof.

(1) The probability of RTS collision is the same as the one in a single cluster with the density of (n- m) nodes.

(2) The probability of a node sending a successful RTS in a single cluster in the cluster is $P_{S,}$ as derived in Section 3.3.2. Equation P_1 gives the probability of x being a receiver of a data packet transmission from any of its neighbours.

$$\mathrm{P}_1 = \frac{1}{(\mathrm{n}-1)} P_s$$

(3) Let cluster $Y_{k,}$, where $1 \leq k \leq m$, wake up just after the RTS/CTS exchange in node x concludes and L_{packet} be the duration of data packet transmission in node x. Then the probability of a border node y_k sending its RTS during the data transmission in node x is proportional to $L_{packet,}$ as derived in Proposition 2.

Let P_2 be the probability that the RTS packet of node y_k collides with the data packet received by node x, τ be the probability of node y_k sending an RTS and $\frac{L}{T}$ be the probability of the RTS being sent during the collision zone in Figure 3-5, then we have

$$P_2 = \frac{\tau L}{T}$$

(4) A data packet collision occurs in node x when x receives a data packet and at least one of the border nodes from the bordering cluster sending an RTS during the collision zone. Hence, we have

$$P_{C_{DATA}} = P_1(1-(1-P_2)^{m-1}) = \frac{1}{n-1} P_s \left(1 - \left(1 - \frac{L}{T}\tau\right)^{m-1}\right)$$

3.3.4 Energy Wastage Analysis for Packet Retransmission

Let the power for transmitting a packet be P_{TR}. Let e be the energy needed to send a single bit, and R is the data rate of the channel. Then, we have

$$e = \frac{P_{TR}}{R} \quad (3\text{-}16)$$

The energy wasted in collision without a hidden terminal is equal to the energy to transmit the RTS/CTS packet. Let E_1 be the energy waste in an RTS collision in a single cluster network and $E[RTS]$ be the size of an RTS/CTS packet. Then, we have

$$E_1 = P_{C_{RTS}} * 2 * E[RTS] * e \quad (3\text{-}17)$$

Energy wasted in a packet collision in a multi-cluster network with hidden terminals comprises (1) the energy spent in the case of RTS/CTS collision and (2) the energy spent in the case of a RTS/CTS and a packet data collision. Let E_2 be the energy waste due to a packet collision in the border nodes of a multi-cluster network and $E[P]$ be the size of a data packet. Then we have

$$E_2 = E_1 + P_{c_{DATA}} * (E[P] + E[RTS]) * e \quad (3\text{-}18)$$

The model proposed in section 3.3 yields the following performance evaluation parameters, which will be validated in section 3.4.

(1) The network troughput for various network densities and traffic load as shown in equation (3-10).
(2) The probability of packet collision for single cluster networks, as shown in equation (3-11) and multi-cluster networks, as shown in equation (3-14) and (3-15).
(3) Additional energy spent for packet retransmission, in single and multi cluster networks, as shown in equation (3-17) and (3-18).

3.4 Performance Evaluation

3.4.1 Simulation Environment

To validate our proposed model, we run multiple simulations in the MATLAB 2014 environment and compare the results to those obtained from the mathematical model. Table 3-3 lists the parameters used in the simulation. The parameters used in the simulation are based on S-MAC implementation in Mica motes described in [21].

Table 3-3 MAC Parameters for the Simulation

Parameter	Value
Contention Slot (σ)	$2.5*10^{-3}$ s
Listen Time	15 slots SYNC and 31 slots DATA
Duty cycle	10%
TX Power	36mW
Data Rate	100 kbps
Packet Length	0.25, 0.5, 1. 1,5, 2 KB
RTS/CTS length	10 bytes

Each node generates unicast packets according to the Poisson distribution with mean arrival rate λ. In S-MAC-based networks, nodes in a common collision channel can only transmit one packet per frame, thus the maximum throughput of the neighbourhood is 1/T. For the duty cycle of 10%, the frame length is 1.15 s, and then the maximum throughput of the network is 0.87. The offered load of the network is

$$offeredLoad = \frac{1}{T} n\lambda$$

When the offered load approaches one, then the network enters the saturated condition. In the simulation, we use the network density, n = 3, 5, 7. Table 3 shows the maximum value of the packet arrival rate before the network enters the saturated condition.

Table 3-4 Maximum Arrival Rate for Unsaturated Channel

	n =3	**n= 5**	**n=7**
Max λ	0.29	0.174	.124

To measure the network performance, we use three parameters, throughput, packet delivery ratio (PDR) and energy wasted caused by packet collision. To maintain the accuracy of the simulation, for each scenario, we run the simulation for 10000 S-MAC frames (in this case equivalent to 11500 s) and repeat this at least 20 times or until the result is stabilized.

3.4.2 Result and Analysis

In the first part of the simulation, we show the performance evaluation in a single cluster network. Figure 3-7 shows the network throughput for various packet arrival rates and contending nodes. As shown, the simulation result (markers) closely follows the mathematical model (lines). In general, the throughput shows a linear relation with the packet arrival rate up to reaching the maximum throughput (≈0.87). After that, the throughput declines as the network enters saturation. As expected, a network with more contending nodes is more prone to throughput reduction.

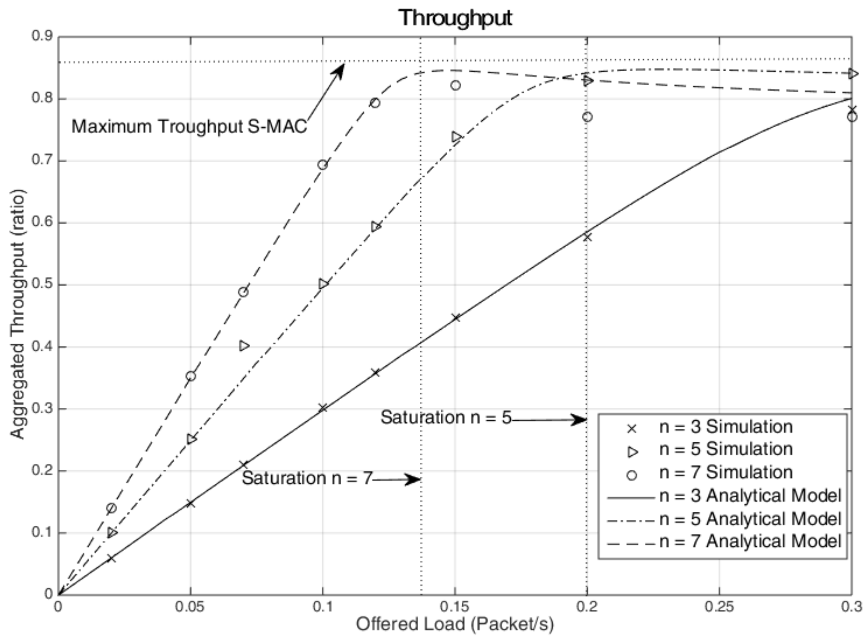

Figure 3-7 Aggregated Throughput for Various Packet Arrival Rate

Figure 3-8 shows the packet delivery rate (PDR), i.e. 1- P_c, as a function of packet arrival rate λ. Based on the figure, the simulation produces an almost identical result to the model. The graph also shows the maximum arrival rate allowed for a particular requirement of PDR for each neighbourhood of size *n*. For example, for network density n = 5, to achieve 90% of PDR, the maximum λ allowed is 0.27, and for n= 7 and the same PDR requirement, the maximum λ is 0.14.

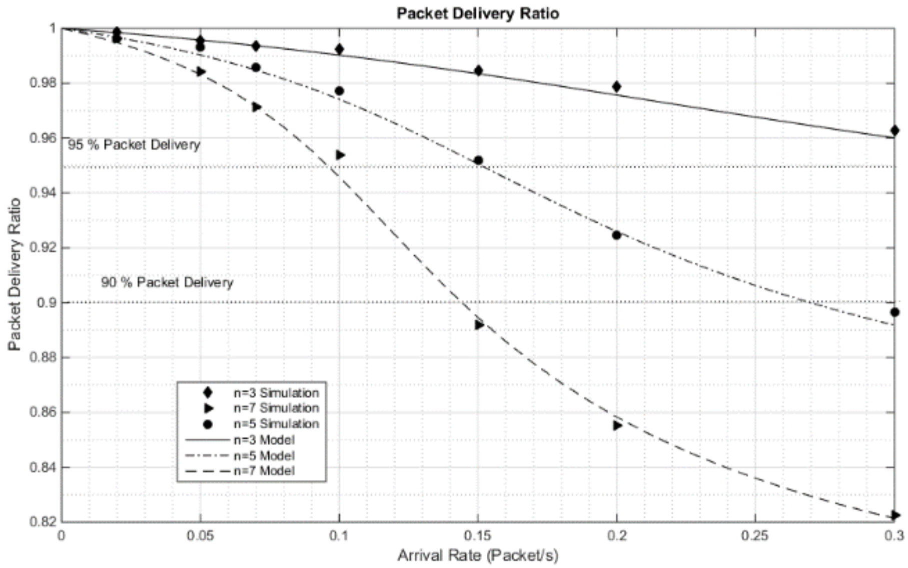

Figure 3-8 Packet Delivery Ratio versus packet arrival rate

In the next part of the simulation, we analyze the impact of having a multi-cluster network on the additional energy waste in the border nodes due to data packet collision. The results derived from the analytical model are represented in lines and the results of the simulation are represented in markers. For all the figures, we use network density *n* equals five, i.e., each node has four neighbours.

Figure 3-9 shows the data packet collision rate in a two-cluster network as a function of the data packet size for various packet arrival rates with *n*= 5. Based on the figures, the packet collision rate for each lambda exhibit a linear relation to the length of the packet. The simulation results follow the analytical model closely.

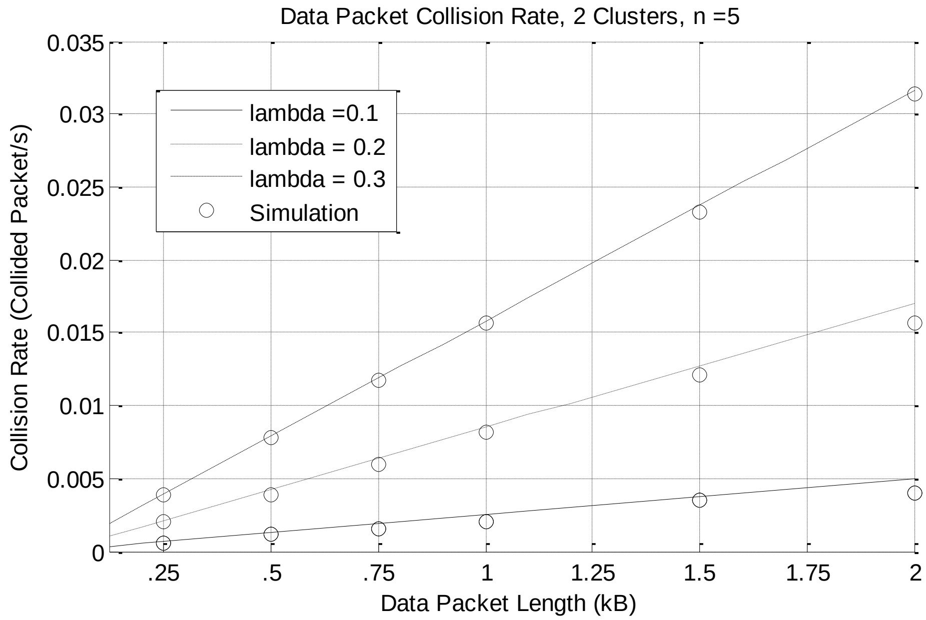

Figure 3-9 Data Packet Collision Rate in the Border Nodes of a Two-Cluster Network for Various Packet Sizes

Figure 3-10 shows the data packet collision rate in a border node of a two-cluster network as a function of the packet arrival rate. For a packet size of 1.5 kB and packet arrival rate of 0.3, for instance, the probability of the data packet collision is approximately 8%.

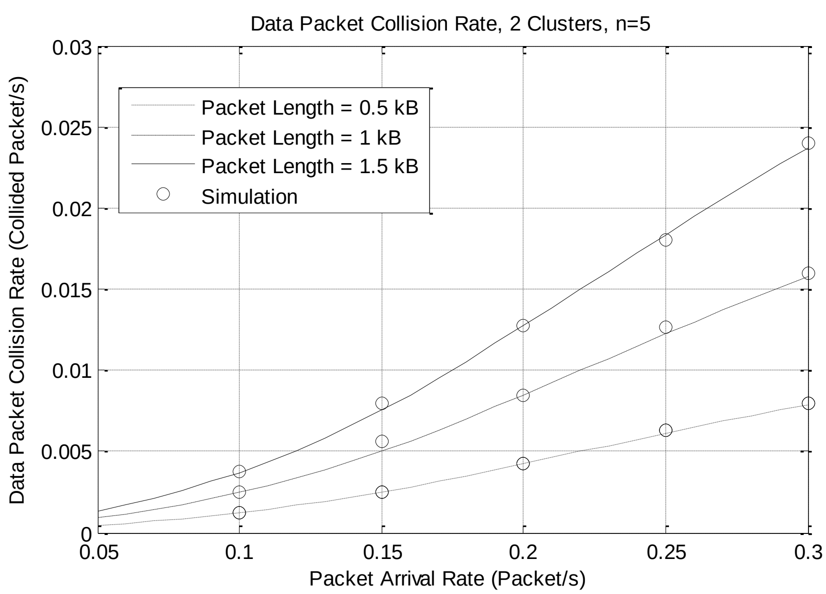

Figure 3-10 Data Packet Collision Rate in the Border Node Two-Cluster Network for Various Packet Arrival Rates

Figure 3-11 shows the data packet collision rate in an m-cluster network as a function of the packet arrival rate.

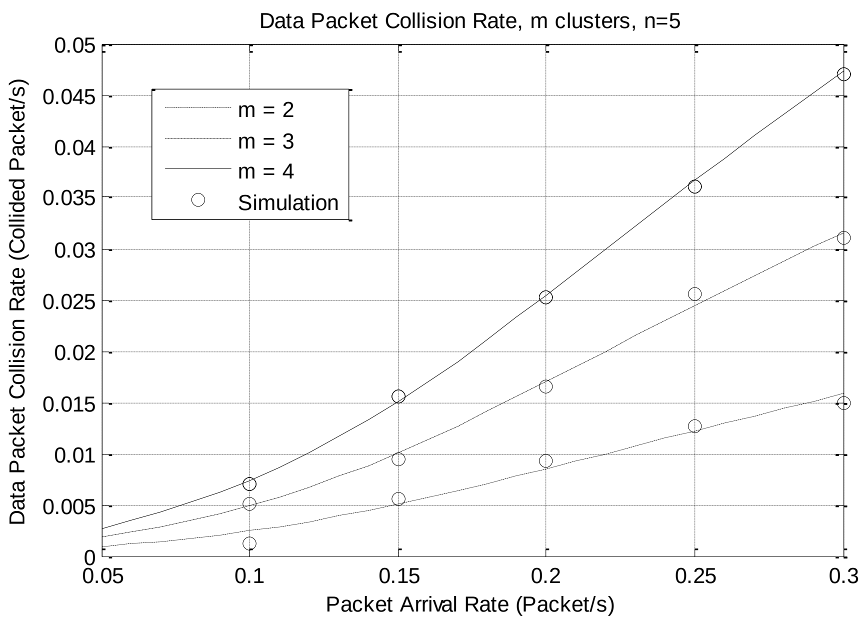

Figure 3-11 Data Packet Collision Rate in the Border Nodes of an M-Cluster Network

Finally, we examine the impact of the hidden terminal problem in a multi-cluster network on the additional energy waste in the border nodes, due to data packet collisions. The energy spent in retransmitting a data packet is much larger than the energy consumed in retransmitting an RTS packet and is proportional to the size of the data packet. As shown in Figure 3-12, energy waste due to packet retransmissions in the border nodes of a multi-cluster network with m=2 is approximately 10 times larger than the nodes of a single cluster network. The additional energy wastage due to packet collisions in a border node also depends on the number of clusters it can hear.

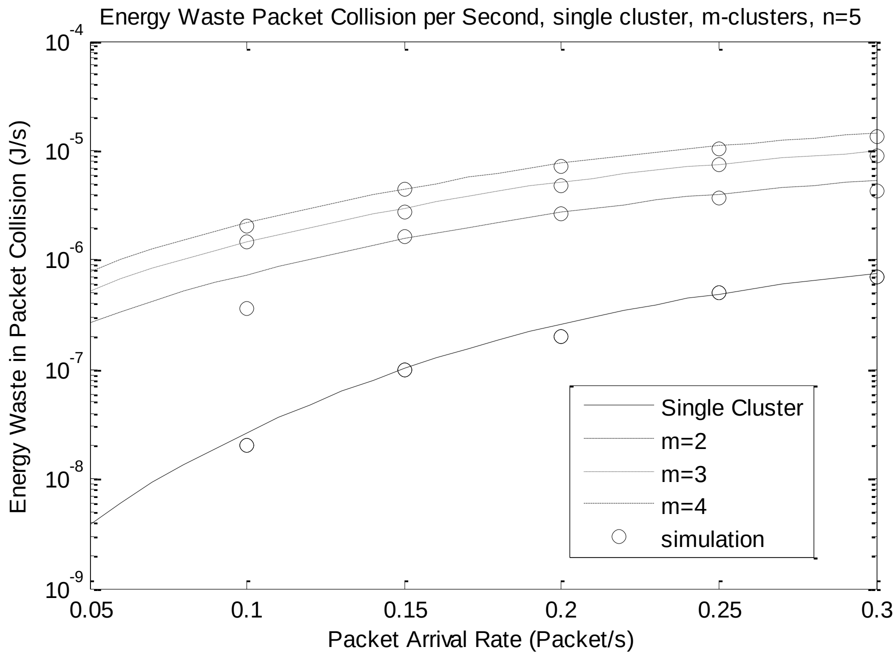

Figure 3-12 The Energy Spent in Packet Retransmission

3.5 Summary

In this chapter, we presented an analytical model to evaluate the performance of the synchronized duty cycle protocols in WSNs. In developing the mathematical model, we consider the conditions of the single cluster networks and the multi-cluster networks in the presence of hidden terminals. We use the parameter of the well-known WSN MAC layer protocol, S-MAC, in evaluating the throughput, packet delivery ratio, and energy waste because of packet collisions.

The work in this chapter can be grouped into two main parts. In the first part of the work, we presented a Markov model to evaluate the performance of a single cluster WSN such as the network throughput and the packet delivery rate (PDR) for various network densities and packet arrival rates. We validated the mathematical model using a computer simulation. The simulation results closely follow the analytical model. In the second part of the work, we extended the model presented in the first part of the work to investigate the effects of having multi-schedule networks on the additional energy wastage caused by data packet collisions in the cluster border nodes. We plotted the data packet collision rate and the additional energy wastage rate in the border node of the model (estimated/average value) and simulation (for various schedule-offset values). The results show that in the worst case, the energy wastage in the border nodes of a multi-cluster network could be up to 100 times greater than in the nodes of a single cluster network.

Having established the disadvantage of having a multi-cluster network in terms of energy efficiency, in the next chapter, we describe the development of a global schedule algorithm for self-organizing WSNs.

Chapter 4 A Global Schedule Algorithm for Self-Organizing Wireless Sensor Networks

4.1 Introduction

Chapter 2 presented the existing energy conservation protocols in the medium access control (MAC) layer for wireless sensor networks (WSNs). Furthermore, we discussed the formation of a multi-cluster network in self-organizing WSNs, and the way it affects network performance. Basically, there are two crucial problems that may occur in a multi-cluster network, the bottleneck problem and the hidden terminal problem. The bottleneck problem is a condition where the border nodes have more active time than the other node in the network and exhaust their energy prematurely. Several studies that have been conducted to investigate this problem [22-24] show that the border nodes consumes up to 50% more energy that the other nodes in the network. The hidden terminal problem causes higher data packet collision rate in the border nodes because they have a different active period with their neighbours. There is no existing study that analytically investigates this problem.

In Chapter 3, a model to analytically evaluate the performance of single and multi-cluster WSNs has been developed. Based on the model, several equations were derived to calculate the probability of a packet collision in a node of a single cluster WSN and a border node of a multi cluster WSN as well as the energy wasted in packet re-transmission in both scenarios of networks The mathematical model and simulation show that the energy wasted due to packet collisions in the border nodes of a multi-cluster network is up to 100 times more than in a single cluster network.

In this chapter, we present an energy-efficient scheme to achieve a single sleeping schedule in self-organizing WSNs. Our proposed scheme uses the schedule offset, the difference between the starting of the next frames of two clusters, as the winning schedule criteria. In the proposed algorithm, the nodes merge as a cluster rather than as an individual node.

The rest of the chapter is organized as follows. Section 4.2 discusses the local synchronization mechanism in duty cycle-based protocols. In section 4.3, we discuss the synchronization issues in duty cycle-based WSNs because of schedule drift. We present three ways to calculate the schedule drift in the network, using a space filling curve algorithm, Monte Carlo simulation, and estimation. In section 4.4, we introduce the basic mechanism of our proposed global schedule protocol. In section 4.5, we present the cluster-merging algorithm in our protocol and in section 4.6, we analyse the performance of the proposed protocol through simulations. Finally, section 4.7 summarizes the paper.

4.2 Schedule Synchronization in Duty Cycle- Based Networks

Nodes in a duty cycle-based WSNs are periodically in active and sleep mode by switching on and off their radio modules to conserve energy. The schedule that determines the time a node switches on and off its radio module is called the sleeping schedule. The duty cycle protocols allow the nodes to generate their sleeping schedules independently if they fail to discover an existing one during their initialisation processes to cope with the distributed nature of self-organizing WSNs. However, in this condition, nodes that implement different schedules cannot communicate with each other since they have different active times. The pioneer of the duty cycle based protocol, S-MAC [20, 21], proposed the use a special packet, called SYNC, to deal with this issue.

Nodes in a WSN broadcast a control packet called SYNC to announce their schedules to their neighbourhoods. A SYNC contains the ID of its transmitter, the schedule ID (i.e. the ID of the cluster in which the sender is a member) and the duration until the next wake up of the schedule. SYNC serves two purposes. Firstly, it tells the neighbourhood that a particular node is still active. Secondly, it notifies the nodes of neighbouring clusters of its cluster's schedule.

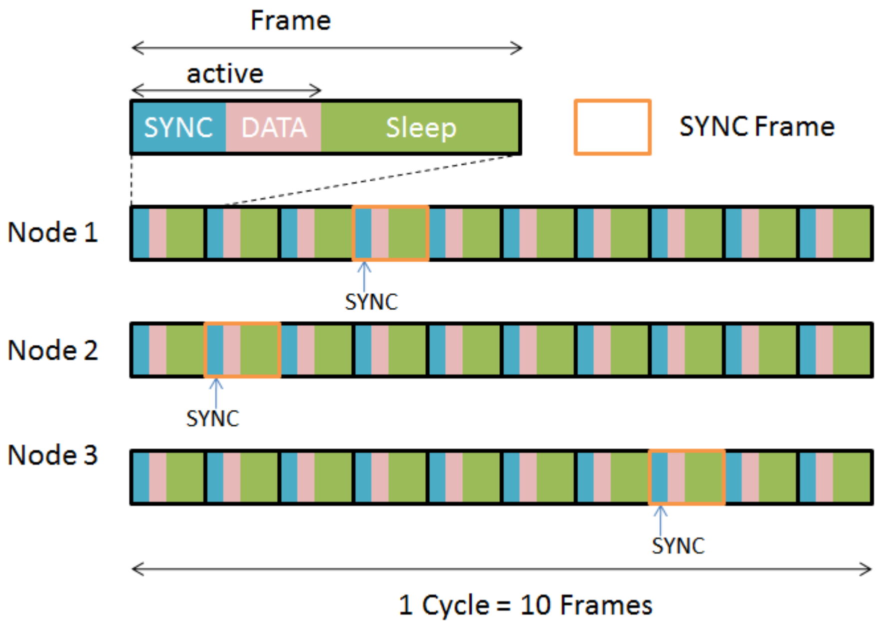

Figure 4-1 SYNC Frame

Nodes broadcast their SYNCs during the SYNC periods of a randomly chosen frame, called the SYNC frame, as shown in Figure 4-1. A node randomly chooses its SYNC frame within the frames in a cycle to decrease the probability of SYNC collision. During its SYNC period in a SYNC frame, a node contends for the channel access before sending its SYNC. During the other frames, on the other hand, the nodes stay active to listen for SYNCs from other members of their cluster.

Nodes could miss other nodes' SYNCs if they are members of different clusters because they sleep and wake up at different times. In Figure 4-2, node 1 and node 2 are neighbouring nodes and members of two different clusters, hence when one of the nodes sends its SYNC, the other node is in the sleep mode. With the purpose of discovering SYNCs sent by nodes in neighbouring clusters, S-MAC proposed a mechanism called synchronization cycle. In every ten cycles, nodes randomly pick a cycle as their synchronization cycle. During its synchronization cycle, a node stays active for the duration of the entire cycle to ensure it receives any sent SYNC in its neighbourhood.

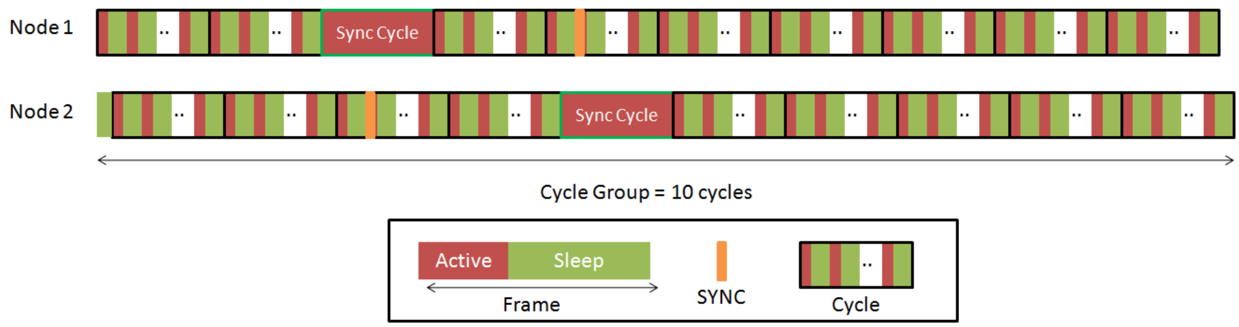

Figure 4-2 Synchronization Cycle

4.3 The Challenges in Developing a Global Sleeping Schedule Algorithm in Self Organizing WSNs

Developing a global sleeping schedule protocol for self-organizing sensor networks encounters several challenges that have not been discussed thoroughly by existing protocols. This section discusses the main challenges of developing a global sleeping schedule in such a network, namely the merging efficiency and the schedule drift problem.

4.3.1 The Merging Efficiency

During the early phase of a network, when it has not been fully connected, having a multi-cluster network is unavoidable. However, as more nodes join the network, the clusters grow and the border nodes start discovering the other clusters. To eventually achieve a single cluster network, when a border node discovers another cluster implementing a different schedule, it needs to make a decision as to whether it will join the newly discovered cluster or not. The process in which nodes in a cluster decide to join another cluster is called the merging process and the time needed to complete the merging process in a network is called the convergence time. Ideally, the convergence time should be as small as possible to reduce the additional energy waste during the merging process. As described in section 2.5.3, the use of either schedule ID or schedule age as a winning schedule criteria in a merging process, as proposed in previous global sleeping schedule algorithms [22-24], is not effective. They either do not promote an efficient convergence time or could result in a large packet overhead.

Another issue in developing an efficient merging is to avoid the schedule oscillation in a distributed network. We consider a network with three virtual clusters, *A*, *B*, and *C*, each implementing a different schedule as shown in Figure 4-3.

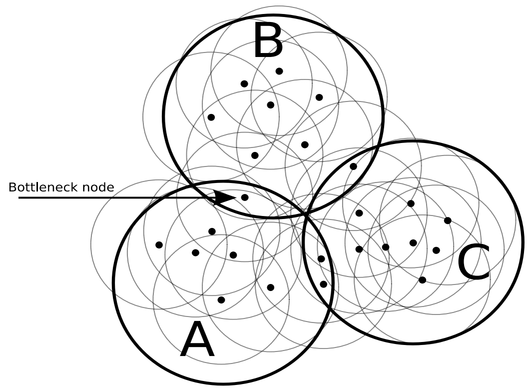

Figure 4-3. Oscillation Problem

Nodes in the clusters' intersection listen to the neighbouring clusters' schedules, make their decisions and propagate their decisions to their neighbouring nodes. The decision needs to be consistent for all node members of a common cluster. Otherwise, the network will be in an infinite convergence time. This could happen when non-linear criteria of a winning schedule are used. Using the size of the network with distributive counting is one example of nonlinear rules. At any given time, the nodes in the network could have different ideas of the number of nodes in the network. A case where some nodes in cluster *A* decide to convert to schedule *B*, some nodes in cluster *B* choose to convert to schedule *C* and some nodes in *C* choose to switch to schedule A results in an oscillation problem.

4.3.2 Schedule Drift in a Long Chain Network

In a self-organizing WSN, nodes need to deal with synchronization issues that are slightly different from the ones in the other types of networks due to the following conditions.

(1) Each node comes to life and starts its clock at slightly different times.
(2) The nodes do not possess knowledge of their neighbourhood, including the distance to their neighbours which could pose challenges in local synchronization
(3) Synchronizing the clock with an outside sever (e.g. the use of GPS satellite synchronization) in most cases is hardly an option due to (a) the cost, (b) the difficult location of the nodes, (c) the limited energy the nodes have.

There are two types of possible synchronization problems in this type of network: clock drift and schedule drift due to long chain propagation.

(1) Clock drift is caused by the inaccuracy of internal clock oscillator with an error rate in the order of 10^{-9} - 10^{-6} s. Numerous studies have proposed algorithms for clock synchronization in WSNs such as RBS [65], TSPN [66], FTSP [67, 68] and TDP [69]. Also, CSMA-based protocols such as S-MAC [20, 21] are much less sensitive to clock synchronization and can tolerate up to 10 µs, which means in the worst case, having network synchronization once in 10 seconds is enough.

(2) Schedule drift due to the propagation delay of the SYNCs. Propagation delay in the network causes nodes to see the start (and consequently the end) of each schedule slightly differently according to their position in the network.

This section deals with the synchronization problem due to nodes seeing the start of a schedule slightly different from their neighbours.

As discussed in the previous section, the nodes in a duty cycle-based WSN maintain local synchronization by periodically broadcasting SYNC packets containing their schedule IDs and the length of time until their next frames. The reception nodes receive the packets after some amount of delay, including processing delay, transmission delay, and propagation delay. These delays make different receivers 'see' the start of the advertised schedule differently. While the receivers can easily correct the drifts due to the transmission and propagation delays, there is no way to calculate the drift due to propagation delays.

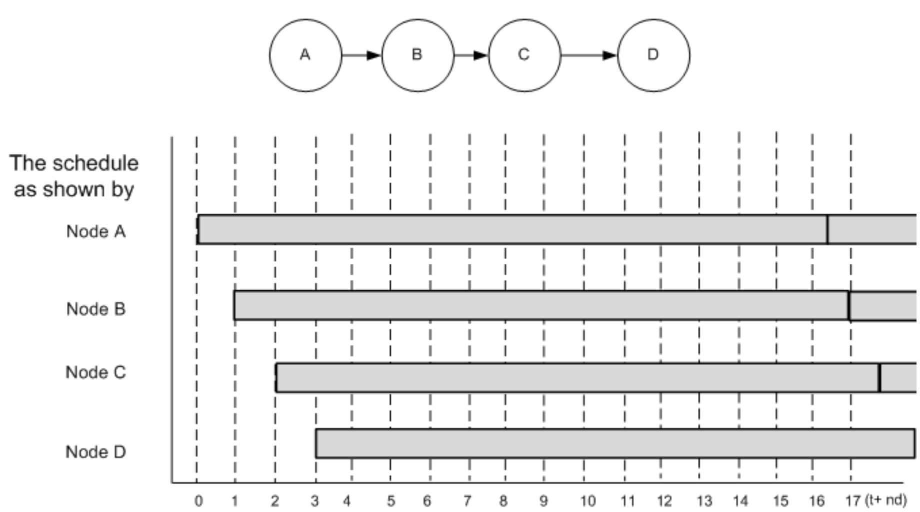

Figure 4-4 Schedule Drift in a Chain Network

We consider the case of a network with four nodes that are connected in a chain topology as shown in Figure 4-4. Because of the propagation delays of the SYNC packets, nodes *B*, *C* and *D* 'see' the start of the schedule slightly behind the time that node *A* 'sees' it . Assuming that the SYNC packets have an average propagation delay of *d*, if node *A* advertises schedule S starts at time *t* then nodes *B*, *C*, and *D* see the schedule start at time $t+d$, $t+2d$, and $t+3d$ respectively.

The drift caused by a propagation delay is generally small and easy to correct using the guard bits. However, in a long chain network, the accumulated drift could cause a problem. We consider a long chain network with $n+1$ nodes where node $N_{(n+1)}$ is in the transmission range of node N_1 and node N_n as shown in Figure 4-5.

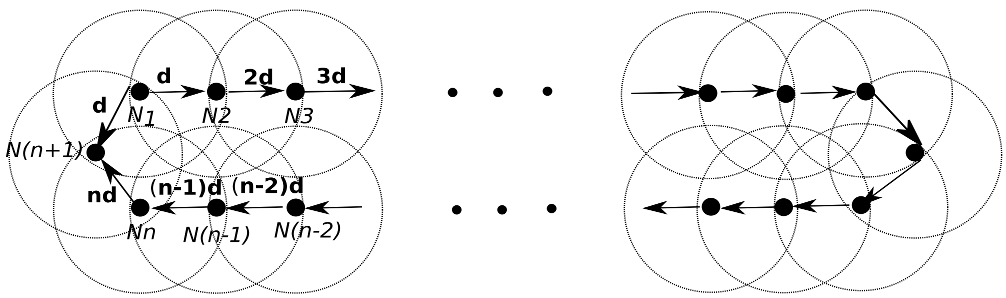

Figure 4-5 Schedule Drift in a Long Chain Network

In Figure 4-5, the nodes N_1, N_2, .. N_n are connected in a chain topology and implement a common schedule S. Because of the propagation delays of the SYNCs, *Nn* starts its frame $(n-1)d$ time unit after N_1 does. Consider a case where new node $N_{(n+1)}$ recently joins the network and start its discovery period. During the discovery period, it receives two SYNCs from N_1 and N_n announcing schedule *S*. If, for example, the SYNC sent by N_1 announces that the next frame starts at time *t*, then, consequently, the SYNC sent by N_n announces that the next frame starts at time $t+(n-1)d$. There are two issues that $N(n+1)$ needs to address. Firstly, N_{n+1} needs to realize that the SYNC packets of N_1 and N_n actually announce the same schedule. Secondly, for example, if it decides to synchronize itself with node N_1, it then receives the packets from node *Nn* with $(n-1)d$ unit time drift. In a low data rate network where the bit duration is much longer than the schedule drift, this would not pose a problem. However, as the data rate of the network increases, if the bit duration is smaller than the schedule drift, this could cause an error bit interpretation. In the next section, we derive the maximum schedule drift in a network.

4.4 Estimating the Schedule Drift in a Network

This section responds to *RQ 2.1* by presenting several methods to estimate the schedule drift in a network. As discussed in the previous section, the schedule drift in a network is caused by the propagation delay of the SYNC packets advertising the schedule. Since the propagation delay of a packet is the function of the distance between the transmitter and the receiver, we derive the maximum schedule drift in a network by calculating the maximum propagation path length in a network.

4.4.1 The Theoretical Upper Bound of Maximum Schedule Drift in a Network

The first step in deriving the maximum bound of schedule drift in a network is to calculate the upper bound of the chain length in the network. We consider a square network with a dimension of M x M. Nodes with a uniform transmission range of *r* are placed in particular positions in the network to create the maximum possible chain length given the area of the network, as shown in Figure 4-6.

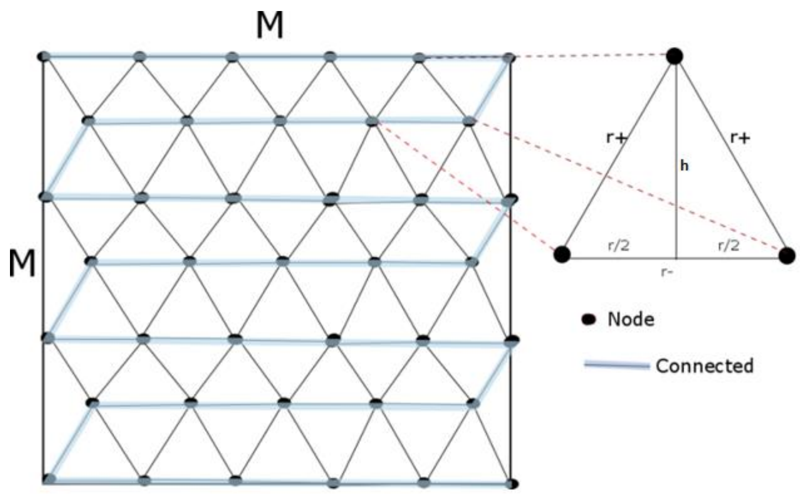

Figure 4-6 Upper Bound Longest Chain in a Square Network

In Figure 4-6, the maximum distance between two nodes that are connected is *r*- (slightly less than *r*) and the minimum distance between two nodes that are not connected is *r*+ (slightly larger than *r*). Because of the geographical span, the chain needs to be 'folded back' to fit the dimension of the network. The distance between folds is given by $h = \frac{1}{2}r\sqrt{3}$. The maximum number of 'folds' in the network, $F = \frac{M}{h} = \frac{2M}{r\sqrt{3}}$. The number of hops in each 'fold' is $\frac{M}{r}$. Finally, we can calculate the maximum number of hops in a network N as shown in equation (4-1).

$$N = \frac{2M^2}{r^2\sqrt{3}} \tag{4-1}$$

Assuming that a packet in the network propagates at the speed of light, we have the maximum schedule drift as the maximum physical distance the SYNC packet propagated through divided by the speed of light, as shown in equation (4-2)

$$\delta_{max} = \frac{Nr}{c} = \frac{2}{\sqrt{3}}\frac{M^2}{cr} \tag{4-2}$$

Noting that M^2 is the physical area of the network, we can estimate the upper bound of schedule drift in a network with an area of A as shown in equation (4-3).

$$\delta_{max} = \frac{2}{\sqrt{3}}\frac{A}{cr} \tag{4-3}$$

In Figure 4-7, we plot the upper bound of schedule drift in a square network for various network sizes and transmission ranges.

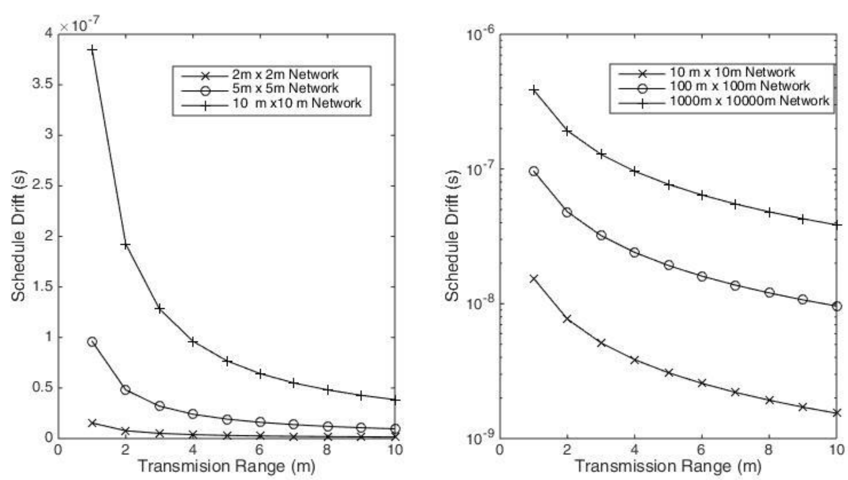

Figure 4-7 The Upper Bound of Schedule Drift in Randomly Deployed Networks

Calculating the upper bound of schedule drift in the network serves two main purposes in developing protocols for self-organizing WSNs:

(1) It gives the minimum size of guard bits needed in the network to avoid the error bit interpretation. Due to schedule drift, nodes may wrongly interpret the start of a frame and wake-up after one of their neighbours has transmitted its packet. Waiting for the duration of several guard bits before actually starting a transmission process ensures that the other nodes in its cluster have been active, before a transmitting node sends its packet.

(2) It defines the maximum time difference between the starts of the next frames in two different nodes to be considered that the two nodes operate on the same schedule. When the difference of the starts of frames of two schedules is within the range of maximum schedule drift, we assume that they operate on a same schedule. They wake-up and sleep at approximately the same time. The small time difference of the starts of the frame can be tolerated by the guard bits. We use this in designing our proposed global schedule protocol in section 4.5.

This section provides an asymptotic analysis of the drift in a network. The equations in (4-1), (4-2), and (4-3) serve as the upper bound of the number of hops and drift in the network where the nodes are placed specifically to create the maximum chain length. In a reality of a randomly deployed network, the numbers are considerably smaller as shown in the simulations we run in the next section.

4.4.2 Simulation of Maximum Chain Length in a Randomly Deployed Network.

To observe a realistic maximum chain length in a network where the nodes are randomly deployed, we conduct a Monte Carlo simulation in the MATLAB 2014 environment, with various network sizes and densities for a unit transmission range (r=1). To achieve good accuracy, we run 100 simulations for each condition.

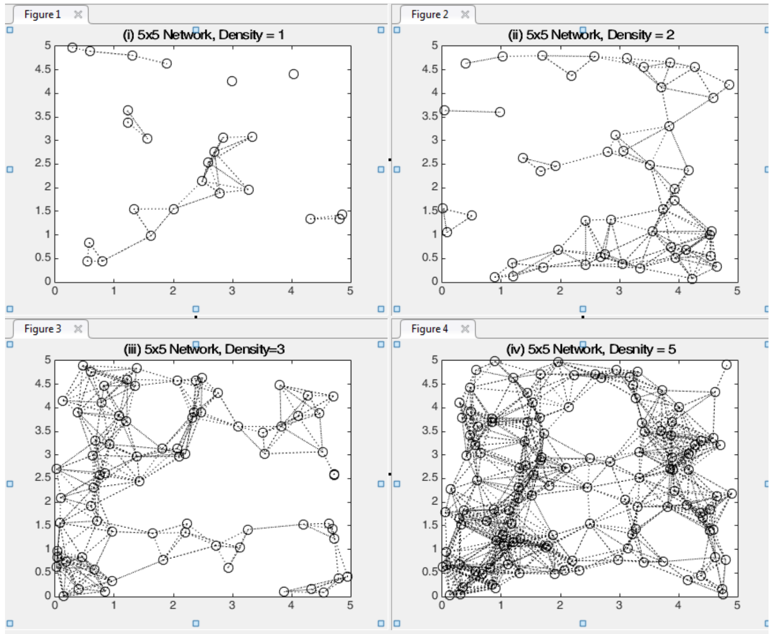

Figure 4-8 Example of Randomly Deployed Network

Figure 4-8 is an example of the connectivity in randomly deployed networks with a fixed network size and various densities. In a very sparse network, as shown in Figure 4-8 (i), the connectivity is so low that the network is partitioned into smaller isolated networks. As the density of the network increases, so does the connectivity of the network (ii). In (iii), the network is already fully connected. From this point, the connectivity of the network remains while the density of the network continues to increase and causes more nodes to be available in a position that can create a shorter path of the two furthest nodes in a network (iv).

The relation between the connectivity of a network and the network density for various sizes of networks is shown in Figure 4-9. The solid lines on the graph show the mean of the network connectivity in 100 simulations and the dashed lines on the graph show the percentage of times the networks reach 100% connectivity (fully connected) in 100 simulations.

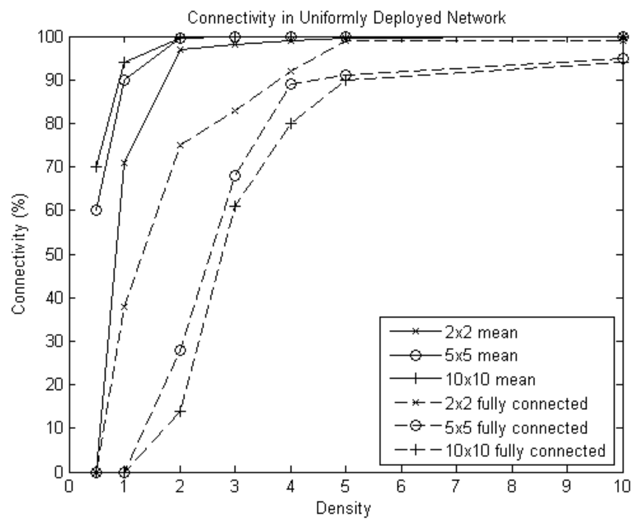

Figure 4-9 Connectivity in a Uniformly Distributed Network of Various Sizes and Densities

Figure 4-10 shows the number of hops between the two furthest nodes in a uniformly deployed network of various network sizes and densities. The solid lines show the average values of 100 simulations and the dashed lines show the maximum values of 100 simulations. A low density network forms a low chain length due to isolated groups of nodes. As the density increases, so does the chain length since more nodes are connected (as shown in Figure 4-9). After the network reaching the maximum connectivity, increasing the node density causes they are nodes in certain positions that create a smaller chain length. This causes the number of hops between nodes to decrease until they are stabilized at a certain value.

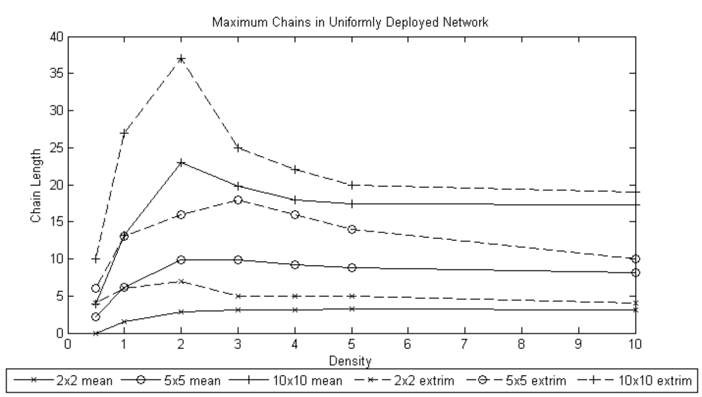

Figure 4-10 The Maximum and Average Value of Maximum Number of Hops in a Uniformly Deployed Network

In the next section, we derive an estimation of the expected maximum number of hops between the two furthest nodes in a network.

4.4.3 Estimated Value of Maximum Chain Length in the Network

As shown in the simulation in the previous section, the maximum chain length in a network highly depends on the area of the network and the density of the network. Given a fixed area of a network, when the density increases, there will be a shortest path connecting the farthest nodes in the network.

We consider a square *MxM* network with the nodes' transmission range *r* equals one. The diameter of the network, $D = M\sqrt{2}$, is the physical distance of two furthest points on a network. In a very dense network, there will be nodes positioned along the shortest path between the two farthest points in such a way that the minimum hops between the two farthest point is $N_{min} = M\sqrt{2}$ .We define n as the ratio of the average number of hops between two furthest points in the simulation and N_{min} as shown in equation (4-4)

$$n = \frac{N}{M\sqrt{2}} \qquad (4\text{-}4)$$

We plot the ratio *n* for various sizes and densities of networks, as shown in Figure 4-11. The results show that as the density of the network increases, n approaches 1.2.

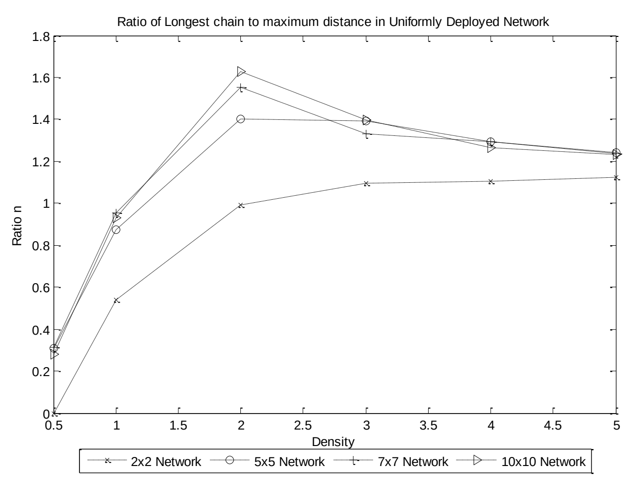

Figure 4-11 The Ratio of the Maximum Hops of a Network to the Diameter of the Network.

Table 4-1 shows the upper bound, the estimated and the average of the number of hops between the two furthest nodes in a network for several network sizes.

Table 4-1 Number of Hops between Two Furthest Nodes in a Network

Network Size	Upper Bound	Estimated	Simulation*
2x2	4.619	2.828	2.923
5x5	28.868	7.071	8.485
10x10	115.470	14.152	16.967

*Simulation with density = 10

In this section, we presented the answer for research question 2.1 regarding the schedule drifts in a network. We present the upper bound, estimated and simulation values of the maximum chain length in a network, which is proportional to the drifts in the network. We use the presented value in calculating the critical range of offset between two schedules in our proposed algorithm in section 4.5.

4.5 Offset based Schedule Selection Scheme

4.5.1 Winning Schedule Criteria

The nodes in the synchronized duty cycle-based WSN periodically broadcast SYNC packets containing their IDs and the duration of time until their next wake up. In a multi-cluster network, when a node receives a SYNC packet from another node belonging to a different cluster, it calculates the offset between its cluster schedule and the newly received schedule. The offset between two schedules S_1 and S_2, d_{S1S2}, is the difference between the time for the start of a new frame for S1,measured from the time for the start of a preceding frame for S2 in Figure 4-12.

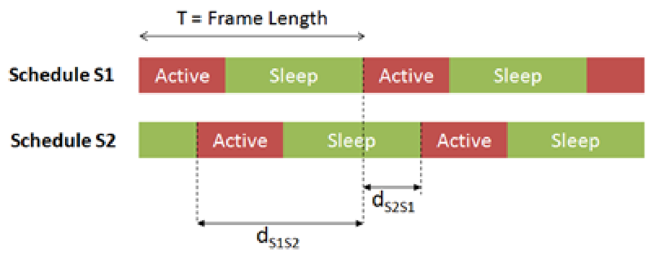

Figure 4-12 The Offset of Two Schedules.

We use equation (4-5) to determine the offset between any two schedules

$$d_{S_1 S_2} = (t_{S_1} - t_{S_2}) \, mod \, T \quad (4\text{-}5)$$

where d_{S1S2} is the schedule offset between S_1 and S_2, and t_{S1} and t_{S2} are the duration of time until the start of the next frame in schedule S_1 and S_2 respectively, and T is the length of a frame. In our proposed scheme, we assume a fixed duty cycle network, so that the duration of a frame, T, is predefined. A node in a cluster S_1 needs to merge with a newly discovered cluster S_2 if $d_{S1S2} >$ ½ T, i.e. the start of the next frame in S_2 precedes that in S_1.

The use of schedule offset as a winning criterion overcomes two drawbacks in the previous proposed global schedule protocols discussed in section 2.5.3.

(1) It avoids a large control overhead resulting from the use of schedule age as the winning criterion, as proposed in [24].
(2) It eliminates the problem of having different schedules with the same schedule ID in a WSN where the nodes have the ability to recharge their batteries, unlike the use of ID as the winning criterion [22, 23]

4.5.2 The Critical Range of Schedule Offset

As discussed in section 4.4, the propagation delay of SYNCs causes schedule synchronization drift in a network. In a long chain network, the propagation delay could be so large that nodes in a cluster may make different decisions in receiving SYNC packets advertising a schedule of another cluster. Let us assume that node, A, a member of cluster S_1, advertises that the next frame in schedule S_1 starts at t_0. Let B and C be members of another cluster, S_2, and implement schedule S_2. Let the next frame in S_2 start at time t_{s2}. Node B calculates the schedule offset as $d_B = ((t_0+\delta_B - t_{s2}) \, mod \, T$ and node C calculates $d_C = ((t_0 + \delta_C) - t_{S2}) mod \, T$, where δ_B and δ_C are the SYNC propagation delay in B and C respectively. If due to the propagation delay, the nodes find that $d_B < ½ \, T$ and $d_C > ½ \, T$, then, according to the merging rule, node B, and C will make different decisions in finding cluster S_1. When nodes which are members of a common cluster make inconsistent merging decisions, this causes an infinite convergence time. A situation where some nodes in cluster S_1 decide to merge with cluster S_2 and some nodes in S_2 decide to merge with S_1 is called network oscillation.

Having established the problem, we define the range of $d = ½ \, T \pm\delta_{max}$ as the critical range of the schedule offset, where δ_{max} is the maximum propagation delay in the network and a function of the network size and the nodes' transmission range. To deal with the inconsistencies in merging when the offset falls into the critical range, we propose the use of a virtual frame in calculating the schedule offset.

4.5.3 Schedule offset of Virtual Frames

The previous section shows that, due to the propagation delays of the SYNC packets, a merging problem may occur when d_{S1S2} falls into the critical range [½ T - δ_{max} ,½ T + δ_{max}]. In the offset-based global schedule, we introduce a concept called virtual frame to deal with the issue, as shown in Figure 4-13.

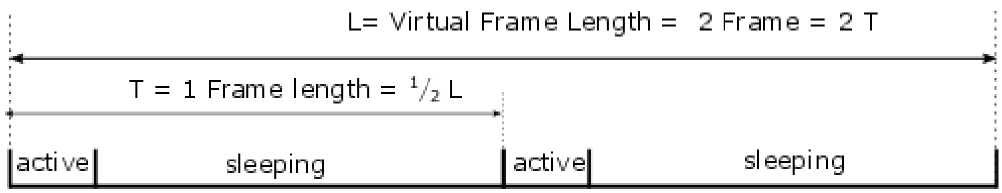

Figure 4-13 Offset Based Global Schedule Virtual Frame

A virtual frame consists of two original frames and has a duration of $L = 2T$. The first active time starts at the beginning of the virtual frame and the second active time starts at $t = ½ L$. In their SYNC packets, nodes advertise the duration of time until the next start of their virtual frame. In discovering a new schedule, a node calculates the offset of its schedule and the newly received ones using equation (4-6)

$$d_{S_1 S_2} = (t_{S_1} - t_{S_2}) mod\, L \qquad (4\text{-}6)$$

Where d_{S1S2} is the offset between the virtual frame in S_1 and S_2, and t_{s1} and t_{s2} are the duration until the starts of next virtual frames in S_1 and S_2 respectively. In this scheme, if the offset of two schedules S_1 and S_2 is $½L$, the nodes in the two clusters essentially wake up and sleep at the same time, hence the clusters do not need to merge to maintain the connectivity of the network.

When a node receives a SYNC that announces another cluster schedule, it follows the rules listed in Table 4-2.

Table 4-2 Merging Rules in the Offset-Based Global Schedule

Offset Value	**Consequence**
[½ L + δ_{max} , L]	Merge with the newly discovered cluster

[0, ½ L - δ_{max}]	Do not merge with the newly discovered cluster, instead broadcast SYNC in the next SYNC period of the other cluster schedule
[½ L - δ_{max} ,½ L + δ_{max}]	Do not need to merge,since the other cluster basically operate on the same schedule.

4.6 Synchronization and Cluster Merging Algorithm

4.6.1 A Local Synchronization Mechanism

In our proposed protocol, we adopt the synchronization mechanism in S-MAC and modify it into the virtual frame scheme. Similar to the scheme in S-MAC, the nodes broadcast their SYNC packets in the SYNC frames to maintain the local synchronization. A SYNC advertises the cluster ID of the sender and the duration until the next virtual frame of the sender's schedule. The SYNC frame is chosen randomly among the frames within five virtual frames. A synchronization cycle consists of five virtual frames and is randomly chosen within ten cycles. A node, during its synchronization cycle, stays active to discover SYNCs from neighbouring nodes, which are members of other clusters. The comparison of the basic frame as proposed in S-MAC and the virtual frame proposed in our study is detailed in Table 4-3.

Table 4-3 Synchronization in S-MAC and the Proposed Protocol

Detail	**Basic Frame (S-MAC)**	**Virtual Frame (the Proposed Protocol)**
Duration	T = Active time + Sleeping Time	L = 2*T = 2*(Active time + Sleeping Time)
Duty Cycle	$\frac{ActiveTime}{T}$	$\frac{2 * ActiveTime}{L}$
1 Cycle	10 frames	5 virtual frames
SYNC Frame	Randomly picked in every ten frames (1 cycle)	Randomly picked in 5 virtual frames then randomly picked between the 1^{st} and the 2^{nd} active time

Synchronization Cycle	Randomly picked in every 10 cycles.	Randomly picked in 10 cycles.

4.6.2 Control Packet Formats

In the proposed protocol, we use two control packets for schedule synchronization, namely, SYNC and SYNC-M packets. Similar to the mechanism in S-MAC, a node periodically sends SYNC packets to announce its schedule. When the nodes in cluster S_1 decide to merge into cluster S_2, cluster S_1 is called the merging cluster and cluster S_2 is called the destination cluster. The nodes in the merging cluster and in the destination cluster broadcast SYNC-M packets to notify their neighbourhood of the merge.

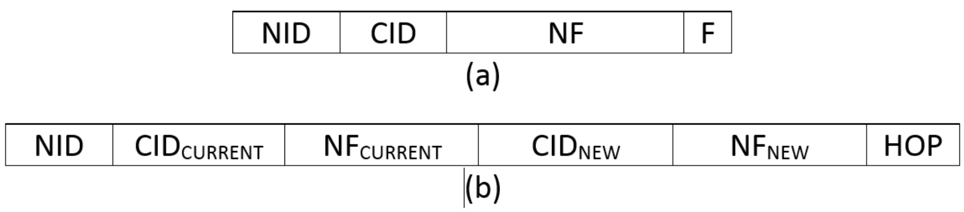

Figure 4-14 Synchronization Packet Formats (a) SYNC (b) SYNC-M

The format of the control packets is shown in Figure 4-14. A SYNC packet consists of the ID of the node that sends the packet (NID), the cluster ID (CID), the duration until the next virtual frame (NF) and a one-bit flag (F). The CID is the ID of the node that created the schedule in the cluster. Nodes set F to one when they are in a merging process. A SYNC-M packet consists of Node ID (NID), the merging cluster ID ($CID_{CURRENT}$), the duration until the next frame in the merging cluster ($NF_{CURRENT}$), the destination cluster ID (CID_{NEW}), the duration until the next frame in the destination cluster (NF_{NEW}) and the number of hops (HOP). The HOP starts at one and is increased every time the packet is rebroadcasted.

4.6.3 The Merging Algorithm

In the proposed merging scheme, the nodes in a cluster merge at the same time. If a node decides to merge into a new cluster, the node creates a special packet named SYNC-M to notify its own cluster and the destination cluster about the merge. Nodes in both the sender's cluster and the destination rebroadcast the SYNC-M for the following reasons:

(1) In the merging cluster, the SYNC-M notifies the other nodes in the cluster about the ongoing merging process.
(2) In the destination cluster, the SYNC-M notifies the nodes in the cluster regarding the intention of another cluster to join their cluster. After receiving the SYNC-M, the nodes are not allowed to initiate another merging process for the duration of the ongoing merging process.
(3) The nodes in the other neighbouring clusters, after receiving a SYNC-M, will not initiate a merging process with the merging cluster for the duration of the ongoing merge.

A merging process of a node starts if a node receives a SYNC or a SYNC-M advertising another cluster's schedule and the offset between its current schedule and the advertised schedule falls into $[\frac{1}{2} L + \delta_{max}, L]$ as detailed in Table 4-2. On receiving a SYNC, a node abides by following steps.

(1) If the node receives the SYNC before it has chosen its schedule, it adopts the advertised schedule in the SYNC as its own schedule and announces this schedule in its SYNC.
(2) If the node receives the SYNC advertising a schedule of a different cluster after it has chosen a schedule, it calculates the offset (d) between its schedule and the newly received schedule and takes action based on the merging rules in Table 4-2. There are two cases to consider:
    (i) If the node decides to merge with the newly discovered cluster, it creates and broadcasts a SYNC-M to notify the members of its current and destination cluster of its decision. It then sets a timer t_{wait}. At the end of t_{wait}, if there is no interruption (detailed in step 3), it merges with the destination cluster by changing its cluster ID and its schedule. The waiting period, t_{wait}, is defined as the maximum time needed to ensure that (1) all nodes in the cluster aware of the merging process, and (2) there is no other member of the cluster initiating another merging process. Each node uses equation (4-7) detailed in section 4.7 to calculate its specific waiting period.
    (ii) If the node does not decide to merge to the newly discovered cluster, it broadcasts a SYNC advertising its schedule during the next active time of the other cluster.

(3) If the node goes with the condition in step 2(i), during the waiting period t_{wait}, it ignores all the received SYNCs announcing other clusters and operates on both its own schedule and the newly received schedule. If the node receives a SYNC-M, there are two cases to consider during the waiting time t_{wait}.

   (i) If (a) the newly received SYNC-M packet has the same cluster ID as the node and advertises that another cluster has decided to merge with its current cluster, and (b) the destination cluster ID in the newly received SYNC-M is smaller than the destination cluster ID in its previously broadcasted SYNC-M it cancels its timer.

   (ii) Otherwise, the node merges with the destination cluster at the end of t_{wait}.

On receiving a SYNC-M packet, there are three cases that determine what action a node will take.

(1) If the SYNC-M packet is sent by another member of its cluster ($CID_{CURRENT} = CID$ and $NF_{CURRENT} = NF \pm \delta_{max}$), it rebroadcasts the SYNC-M and sets the t_{wait} timer. At the end of t_{wait}, the node merges with the newly found cluster.

(2) If the node belongs to the destination cluster ($CID_{NEW} = CID$ and $NF_{NEW} = NF \pm \delta_{max}$), the node sets t_{wait} timer and rebroadcasts the SYNC-M packet. During t_{wait}, the nodes in the destination cluster know that another cluster is merging with their cluster, thus, they will not decide to merge with any other cluster.

(3) Any other cluster that listens to the SYNC-M packet will not consider merging with the cluster in $CID_{CURRENT}$ for the duration of t_{wait}.

We summarize the algorithm of receiving SYNC and SYNC-M procedures in Figure 4-15.

Receiving SYNC Procedure

```
(1)  Calculate schedules offset d
(2)  If d > ½ L + δ_max Then
(3)      CID_TEMP ← CID_SYNC, d_TEMP = d
(4)      Generate and broadcast SYNC-M
(5)      Set t_wait
(6)      While t_wait do
(7)          Implement active time of both schedules
(8)          If receive another SYNC-M Then
(9)              Calculate d
(10)             If (d < ½L - δ_max and CID_SYNC-M < CID_TEMP) Then
(11)                     Call ReceivingSYNC-M Procedure
(12)     Merging: CID ←CID_temp, NF ←NF + d
(13)     Broadcast SYNC
(14) Else If d < ½ L - δ_max Then
(15)     Save the new schedule at schedule table
(16)     Broadcast SYNC at
(17) Else
(18)     If CID > CID_SYNC Then CID ← CID_SYNC
(19) End
```

Receiving SYNC-M Procedure

```
(1)  If CID_CURRENT == CID and NF_CURRENT == NF ± δ_max
(2)      CID_TEMP ← CID_SYNC, d_TEMP = d
(3)      Generate and broadcast SYNC-M
(4)      Set t_wait
(5)      While t_wait Do
(6)          If receives another SYNC Then
(7)              Calculate d
(8)              If (d < ½ L - δ_max and CID_SYNC < CID_TEMP) Then
(9)                  Go To Step 1
(10)     Merging: CID ←CID_temp, NF ←NF + d
(11)     Broadcast SYNC
(12) Else IfCID_NEW == CID and NF_NEW == NF ± δ_max
(13)     Generate and broadcast SYNC-M
(14)     Set t_wait
(15)     While t_wait Do
(16)         If receives another SYNC Then
(17)             Calculate d
(18)             If (d > ½ L + δ_max and CID_SYNC < CID_TEMP) Then
(19)                 Go To Step 1
(20)             Else If (d < ½ L - δ_max and CID_SYNC < CID_TEMP)
                 Then
(21)                 Generate and Broadcast SYNC-M
(22)                 Set t_wait
(23) End
```

Figure 4-15 Receiving SYNC and SYNC-M Procedures

Merging interruption occurs when there is more than one SYNC-M propagating through the network, i.e., during its waiting time, a node receives another SYNC-M informing of another merging process. In this case, the node will cancel the SYNC-M that has a larger destination cluster ID and proceeds with the one with a smaller destination cluster ID. Figure 4-16 illustrates this condition.

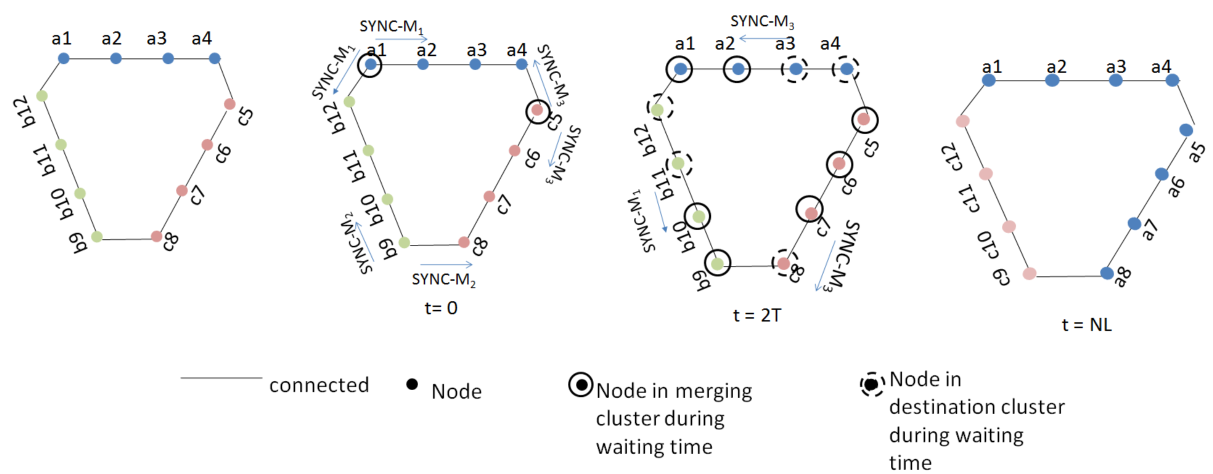

Figure 4-16 An Interrupted Merging Process

We consider clusters *A*, *B* and *C* which are connected is such a way as shown in Figure 4-16. The offset between the clusters $d_{AB} = d_{BC} = d_{CA} = 2/3\ L$, thus, according to the rules in Table 4-2, nodes in cluster *A* needs to merge with cluster *B*, nodes in cluster *B* needs to merge with cluster *C* and nodes in cluster *C* needs to merge with cluster *A*. We investigate the worst scenario, where the border nodes in the three clusters starting their synchronization cycles at approximately a same time. Assuming that $CID_A < CID_B < CID_C$ then the merging process follow these certain steps

(1) At $t = 0$

- Node a_1 discovers cluster *B* and broadcasts *SYNC-M*$_1$ announcing that cluster A needs to merge with cluster B. Nodes in cluster A (the merging cluster) and cluster B (the destination cluster) that receive *SYNC-M*$_1$ perform the same actions.
- Node b_9 discovers cluster C and broadcasts *SYNC-M*$_2$, announcing that cluster B needs to merge with cluster C. Nodes in the merging and destinations clusters perform the same actions.
- Node c_5 discovers cluster *A* and broadcasts *SYNC-M*$_3$ announcing that cluster C needs to merge with cluster A. Nodes in the merging and destinations clusters perform the same actions.

(2) At $t = 2T$

- Node a_3 receives both *SYNC-M*$_1$ and *SYNC-M*$_3$, because $CID\_NEW_{SYNC\text{-}M3} < CID\_NEW_{SYNC\text{-}M1}$, it cancels SYNC-M$_1$ and proceeds with *SYNC-M*$_3$.
- Due to the same reason, node b_{11} cancels *SYNC-M*$_2$ and proceeds with *SYNC-M*$_1$, and node c_7 cancels *SYNC-M*$_2$ and proceeds *SYNC-M*$_3$.

(3) At the end of waiting period, t = NL

- The nodes in cluster *C* adopt the schedule and the cluster ID of cluster *A*.
- The nodes in cluster B terminate their waiting period and resume a normal operation (i.e. in discovering another cluster, *X*, where $d_{BX} > 1/2L+\delta_{max}$, nodes in cluster B can start a merging process with cluster *X*)
- The algorithm reduces the number of clusters in the network from three clusters to two clusters.

As shown in the previous example, even in the worst case scenario that potentially results in a schedule oscillations (detailed in 4.3.1), the algorithm can break the cycle and reduce the number of cluster in the network. In a bigger network that consists of *k* clusters, every time two or more *SYNC-Ms* are propagating in a cluster, the nodes that receive multiple *SYNC-Ms* will discard the *SYNC-M* that has a bigger destination cluster ID and proceed with the one with smaller destination cluster ID. By reducing the number of clusters in the network by at least one at a time, eventually the network operates under a single cluster (i.e. a common sleeping schedule).

4.7 Estimated Time and Energy Spent in a Merging Process

As mentioned earlier, in our proposed protocol, nodes in a cluster merge at the same time. When a node at the border of a cluster discovers another cluster and decides to merge with the cluster, it broadcasts a special packet (SYNC-M). The node then waits for a specific amount of time, t_{wait}, before actually changing its schedule and cluster ID. The duration t_{wait} is the duration needed to ensure (1) all nodes in the merging cluster are aware of the merging, (2) all the nodes in the destination cluster are aware of the merging and (3) all the nodes in the merging cluster are aware if there is an interruption to cancel the merging. At the end of t_{wait}, all the nodes in a cluster make the same decision about the merging. There are two main motivations behind the waiting:

(1) The waiting enables the whole cluster to be aware of the merging and then the nodes in the cluster perform the merging at the same time. It results in lower convergence time compared to the scheme that proposed individual node merging.

(2) It eliminates the oscillation problem, one of the challenges discussed in section 4.3., caused by the inconsistent decisions of the nodes in a common cluster regarding a merging.

The duration of t_{wait} is twice the maximum time needed for the SYNC-M to propagate to another furthest end of the cluster. After receiving a SYNC-M, a node needs to wait until the next frame to rebroadcast the packet. Therefore, the time to propagate the packet to the whole cluster equals the maximum number of hops between two furthest nodes in the cluster times the duration of a frame. In the SYNC-M, there is a field called *HOP* that records how many hops the packet has travelled in the network. In every hop the packet has travelled, we exclude that hop in calculating the waiting time. Each node in the network uses equation (4-7) to calculate its specific waiting time t_{wait}, where N is the maximum number of hops in a network (see section 4.3 for detail), HOP is the numb er of hops the packet has been travelled , T is the duration of a frame and L is the duration a virtual frame.

$$t_{wait} = 2 * (N - HOP) * T = (N - HOP) * L \quad (4\text{-}7)$$

Having derived the waiting time in each node, we can derive the upper bound of convergence time (the merging time of two clusters) for our proposed protocol. We then compare it to the convergence times of the other two existing global schedule protocols, S-MACL [22, 23] and GSA [24], as shown in equation (4-8).

In the offset-based global sleeping schedule, after a border node makes a merging decision, it communicates the decision to its cluster by propagating a SYNC-M. The convergence time is the waiting time of the border node that discoveres the destination cluster and initiates the SYNC-M. In the existing global sleeping schedule protocols, a node could only make a merging decision if it receives a SYNC packet from another cluster. Let us consider a network in that, based on their schedule selection rules, nodes in cluster *A* need to merge with cluster *B*.

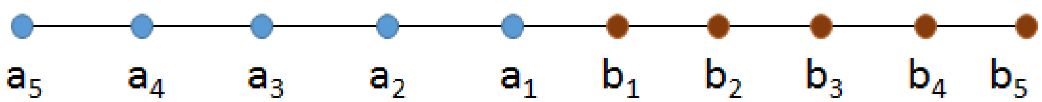

Figure 4-17 A Network Convergence Time

If node a_1 discovers cluster B at $t = 0$, then based on the protocol implemented in the network, the merging process is as follow:

- Offset-based global sleeping schedule protocol
  At t=0, the border node a_1 discovers cluster B. It then broadcasts a SYNC-M to announce the merging and , according to equation, waits for $t_{wait} = N*L$ before joining the destination cluster. The next hop node a_2 takes the same action and waits for $t_{wait} = (N-1)*L$ before merging with the destination cluster, and so on. At the end of $t=N*L$, all the node in cluster A merge with cluster B.
- The existing global sleeping schedule protocols

At $t = 0$, node a_1 discover cluster B and merge with cluster B. In the worst case, it takes until the next synchronization cycle before another node in cluster A, a_2, discovers cluster B and takes the same action. The synchronization cluster on average occurs once every 10 cycles and each cycle consists of 10 frames. Hence, in the worst case scenario, the merging duration equals to the duration of 10 cycles times the maximum number of hops in the cluster.

Let N be the maximum number of hops in the network, L be the duration of a virtual cluster, and C be the duration of a cycle, then equation (4-8) shows the convergence time in the proposed protocol and in the existing protocols.

$$t_{conv_{offset}} = O[N*L]$$
$$t_{conv_{other}} = O[N*10C] = O[50*N*L] \qquad (4\text{-}8)$$

During the merging process, the nodes exchange their control packets including the periodical SYNCs (in the existing protocols and our proposed protocol) and SYNC-Ms (in our proposed protocols). The energy spent to send the control packets during the merging process is as follows

- A SYNC packet is in average sent once every 10 frames (5 virtual frames). The energy spent in sending a SYNC packet equals to the energy for sending one bit (e), multiplied the size of a SYNC packet (*SYNC*)
- A SYNC-M packet is broadcasted by every node in the merging and the destination cluster (N). The energy spent in sending a SYNC-M is the energy for sending one bit *(e),* multiplied by the size of a SYNC-M packet ($SYNC_M$).

The energy spent in a merging process of offset-based global schedule and the other existing global schedules are shown in equation (4-9).

$$O[E_{offset}] = \frac{1}{5L} O[t_{conv_{offset}}] * SYNC * e + N * SYNC_M * e$$
$$O[E_{other}] = \frac{1}{5L} O[t_{conv_{other}}] * SYNC * e \qquad (4\text{-}9)$$

4.8 Simulation and Analysis

4.8.1 Simulation Environment

To validate our proposed model, we run multiple simulations in the MATLAB 2014 environment using the lists of parameters shown in Table 4-4 Simulation Parameter. The protocol parameters are based on S-MAC implementation in Mica motes described in [21]. The size of SYNC and SYNC-M packets are assumed based on the size of information contained in the packets.

Table 4-4 Simulation Parameter

Parameter	Value
Contention Slot (σ)	2.5×10^{-3} s
Listen Time	15 slots SYNC and 31 slots DATA
Duty cycle	10%
Tx Power	36mW
Data Rate	100 kbps
Packet Length	500 bytes
RTS/CTS length	10 bytes
SYNC	4 bytes
SYNC-M	10 bytes

To investigate the effect of the chain length in the network on the convergence time of the algorithm, we use chain topology with various numbers of nodes in the network.

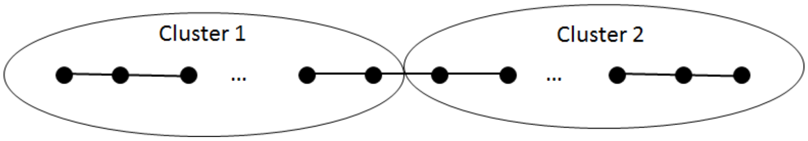

Figure 4-18 Chain Topology

We compare the performance of our proposed protocol with the performance of the ID-based global schedule protocol (SMAC-L[22, 23]).

4.8.2 Results and Discussion

Figure 4-19 shows the convergence time (i.e. the merging time) of two clusters. It is the duration between the first time a node in the merging cluster receives the SYNC of the destination cluster and the time the merging concludes.

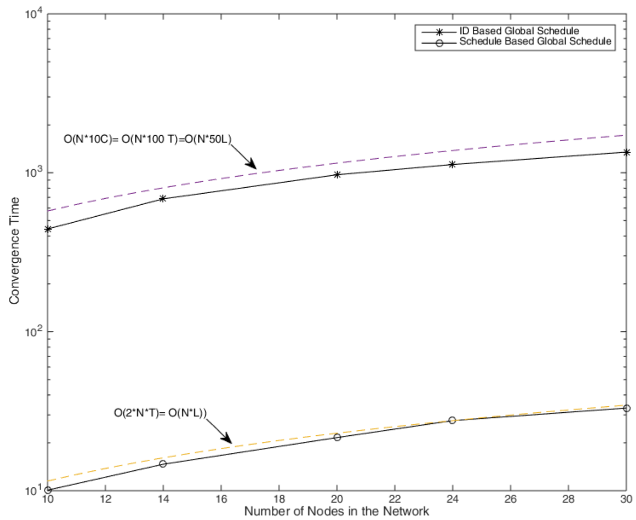

Figure 4-19 The Convergence Time in the Chain Topology Network

The solid lines represent the simulation results and the dashed lines represent the upper bound of the convergence time as shown in equation (4-8). As in the simulation, we have a chain topology (the number of hops between the two furthest nodes is always the maximum), and the simulation result closely follows the theoretical one. The results show that our proposed protocol has much smaller convergence time (roughly 50 times smaller) compared to the ID-based global schedule protocol.

In Figure 4-20, we plot the energy spent by the network to send the control packet during the merging process. Similar to the previous figure, the solid lines represent the simulation results and the dashed lines represent the expected value as shown in equation (4-9). The results show that even though the offset-based protocol has larger control packet sizes compared to the ID-based protocol, due to the smaller convergence time during the merging process, it spends less energy than the ID-based protocol. On average, in a network with chain topology, the energy spent to send the control packet in our proposed protocol is about 10 % lower than the ID-based global schedule protocol.

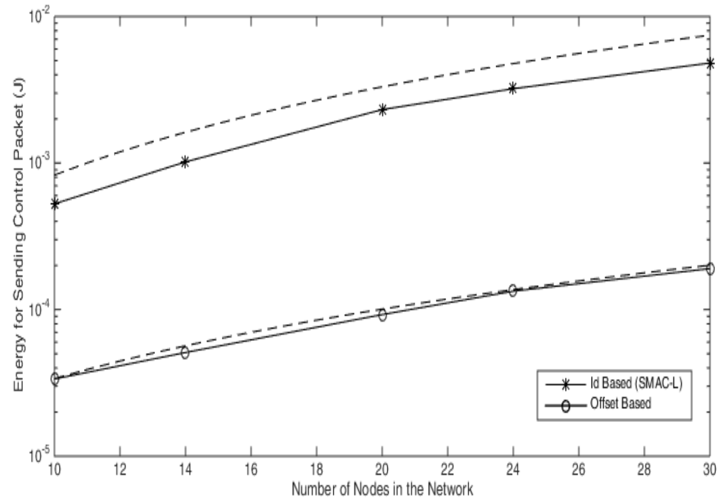

Figure 4-20 The Energy Spent during Convergence Time in Chain Topology

4.9 Summary

In this chapter, we described the development of a global sleeping schedule protocol for a self-organizing WSN to provide an answer to the second main research question (RQ. 2) and all of its sub-questions. The work in this chapter can be grouped into three main parts. In the first part, as the response to RQ 2.1, we present ways to derive the maximum synchronization drift in the network. The second part of the work concerns the development of our proposed global sleeping scheduled protocol as the solution to RQ 2.2. The third part of the work provides a solution to RQ 2.3. In this last part of the chapter, we provide a mathematical estimation and run a MATLAB simulation to compare the performance of proposed protocol compared to an existing protocol, S-MACL[23], in terms of the merging time and the energy spent during the merging process.

The upper bound of schedule drift in the network is the function of the network size and the nodes' transmission range. We use the space filling curve algorithm to derive equation (4-2) and (4-3), which shows the relation of the network size and the transmission range to the maximum schedule drift in a network. While giving the worst possible schedule drift in the network, the results derived from the space filling curve algorithm do not give a good picture of the actual network condition. We then run a Monte Carlo simulation to investigate the relation of the network size and density to the number of hops between the two furthest nodes in a network. We finally derive the estimated maximum chain length in a dense network.

As the response to RQ 2.2, we presented our proposed global sleeping schedule protocols in section 4.5 and 4.6. We proposed the use of the offset between two schedules as the winning criterion in the merging process to overcome the problems resulting from the use of schedule ID and schedule age as the winning criteria as proposed by the existing global schedule protocols. We use the results of the work in section 4.3 to derive the critical range of schedule offset and we proposed the use of virtual frames to deal with this problem.

Section 4.7 and 4.8 dealt with the performance of the proposed protocol in term of the convergence time and the energy spent during the merging process. The results show that the merging time in our proposed protocol is 50 times less than the existing global schedule protocol (S-MAC L [52]). Moreover, during the merging process, our proposed protocol saves up to 90% more energy than S-MACL.

Chapter 5 Summary and Future Work

5.1 Summary and Significant Outcomes

Energy conservation is one of the most serious challenges in developing a protocol for wireless sensor networks (WSNs), especially the ones that have a self-organizing nature. The limited energy resources of the nodes and the difficulty of recharging or replacing the nodes' batteries make reducing the occurrence of energy waste an ideal way to prolong the life span of the networks. Studies show that one of the major sources of energy waste in WSN is a condition called idle listening, in which the nodes listen to an idle channel. To alleviate this problem, the duty cycle scheme in WSN allows the node to sleep, i.e. turning off their radio module, periodically according to a certain schedule.

While it is desirable that all the nodes in a network operate under a single sleeping schedule, the existing duty cycle protocol enables the formation of multiple schedules in the network, thus dividing the network into virtual clusters. The border nodes of the clusters need to have special arrangements to enable communication between the clusters, which results in a higher energy depletion and/or higher packet collision rate in the border nodes. Due to the important role played by the border nodes in maintaining the connectivity of the network, this in the end severely affects the network performance.

Evaluating the performance of the existing network protocols plays a significant part in designing a mechanism to improve the performance of the protocols. In this thesis, we present a quite simple but novel model to evaluate the effects of having a multi-cluster network on the degradation of the performance of the border nodes. We started by developing a Markov model of the medium access control (MAC) state of a node in a synchronized duty cycle-based network. From the model, we derived the performance evaluation parameters, such as the packet delivery rate (PDR) and the throughput in a 1-hop network as functions of the network density and the packet arrival rate. We then extend the model to investigate additional data packet collision due to the hidden terminal-like problem in a multi-cluster network. Both the mathematical models and the simulation show that the border nodes in a multi-cluster network have a much higher energy depletion rate due to having to retransmit the collided data packets.

For instance, for a border node that can listen to three other different clusters, the energy waste due to packet retransmission is up to 100 times larger than the nodes of a single cluster network.

In developing a protocol for a self-organizing network, part of the work is to calculate the synchronization drift in the network. Since the duty cycle protocols achieve local synchronization by broadcasting a control packet, called SYNC, the synchronization drift is proportional to the propagation delay of SYNC packets. We investigated and presented the relation between network size, network density and the transmission range of the nodes to the maximum synchronization drift in the network. We also calculated the longest propagation path length in the network and the number of hops between the two furthest nodes in the network.

Having investigated the disadvantageous results of a multi-cluster network, we proposed a global sleeping schedule protocol to improve the energy conservation mechanism in WSNs. In our proposed protocol, a node, upon discovering a new schedule, employs a schedule selection algorithm to decide the winning schedule. We proposed the offset between two schedules as the winning criterion in our schedule selection algorithm. After discovering a new schedule and deciding to adopt the schedule, a border node informs the other nodes in the cluster of its decision by broadcasting a special control packet containing the information of the newly discovered schedule. This mechanism reduces the convergence time in the network since all the nodes in a cluster perform the schedule selection algorithm at approximately the same time. Both the mathematical estimation and the simulation results show that our proposed protocol has approximately 50 times smaller convergence time and saves up to 90% more energy compared to the other global sleeping protocols during the convergence time.

5.2 Key Achievements

The study yields the following key achievements:

(1) The development of a model that provides a simple and novel solution to measure the performance of the synchronized duty cycle-based wireless sensor networks. The model provides the relationship between the data packet arrival and the density of a network to the throughput and packet data delivery (PDR) in a 1-hop network and is validated by simulations.

(2) The development of a mathematical model to analyze the effects of the hidden terminal problem in the border nodes of a multi-cluster network, which has not been investigated by any of the current literature. The model is an extension of the model in (1) that provides the relationship between the packet arrivals, the packet size and the number of bordering clusters to the data packet collision probability and the energy waste rates in the border nodes and validated by simulation. The results show that, the border nodes spend up to 100 times more energy for retransmitting collided packets than the nodes in a single cluster network.

(3) The development of a global sleeping schedule protocol as a solution to the problems caused by multi-cluster WSNs. In the current literature, there are only two other works that investigate the problem, both serving as preliminary studies as they have not deeply investigated the challenges of implementing a global schedule in a self-organizing environment. The performance evaluation shows that the proposed protocol has a significantly smaller (up to 50 times smaller) convergence time and consequently saves a significant amount (up to 90% more than the other global schedule protocols) of the convergence process energy.

5.3 Limitations and Future Work

The study has several limitations that could serve as motivations of future work.

(1) The model assumes a low data traffic network and each node has a transmission buffer size of exactly one packet. When the network traffic increases, the nodes need to buffer more than one packet, which affects the model's ability to correctly measure the performance of the network. It would be interesting, as future work, to investigate the performance of a network during a time crucial event detection. It would also be interesting to investigate a model with different packet arrival distributions.

(2) Another interesting research topic would be to investigate quality of service (QoS) in WSNs during time critical event detection. Generally, QoS takes a second priority after energy conservation in WSNs. In the detection of time critical events, however, QoS becomes crucial, since the reporting packets have to be delivered as soon as possible. One of the proposed solutions is the use of a dynamic duty cycle, in which a node increases its duty cycle, i.e. shortens its sleeping period if it has a full transmission buffer. However,

the duty cycle change needs to be propagated in the network, otherwise, the other nodes, e.g. the intended receiver, will not be aware of the change. Moreover, during critical event detection, the nodes located near the event compete to send their packet at the same time, which could result in network congestion.

Bibliography

[1] R. Yadav, S. Varma, and N. Malaviya, "A survey of MAC protocols for wireless sensor networks," *UbiCC journal,* vol. 4, pp. 827-833, 2009.

[2] S.-H. Yang. (2013). *Wireless Sensor Networks : Principles, Design and Applications (1 ed.)*. Available: http://latrobe.eblib.com.au/patron/FullRecord.aspx?p=1538699

[3] M. Nabi, M. Blagojevic, M. Geilen, and T. Basten, "Dynamic data prioritization for quality-of-service differentiation in heterogeneous wireless sensor networks," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society Conference on*, 2011, pp. 296-304.

[4] A. Milenković, C. Otto, and E. Jovanov, "Wireless sensor networks for personal health monitoring: Issues and an implementation," *Computer communications,* vol. 29, pp. 2521-2533, 2006.

[5] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser*, et al.*, "Health monitoring of civil infrastructures using wireless sensor networks," in *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, 2007, pp. 254-263.

[6] M. P. Đurišić, Z. Tafa, G. Dimić, and V. Milutinović, "A survey of military applications of wireless sensor networks," in *2012 Mediterranean conference on embedded computing (MECO)*, 2012, pp. 196-199.

[7] N. K. Suryadevara and S. C. Mukhopadhyay, "Wireless sensor network based home monitoring system for wellness determination of elderly," *Sensors Journal, IEEE,* vol. 12, pp. 1965-1972, 2012.

[8] S. D. T. Kelly, N. K. Suryadevara, and S. C. Mukhopadhyay, "Towards the implementation of IoT for environmental condition monitoring in homes," *Sensors Journal, IEEE,* vol. 13, pp. 3846-3853, 2013.

[9] A. Khadivi, L. Georgopoulos, and M. Hasler, "Forest fire localization using distributed algorithms in wireless sensor networks," in *Proceedings of PIERS*, 2009, pp. 452-455.

[10] J. Lloret, M. Garcia, D. Bri, and S. Sendra, "A wireless sensor network deployment for rural and forest fire detection and verification," *sensors,* vol. 9, pp. 8722-8747, 2009.

[11] Y. E. Aslan, I. Korpeoglu, and Ö. Ulusoy, "A framework for use of wireless sensor networks in forest fire detection and monitoring," *Computers, Environment and Urban Systems,* vol. 36, pp. 614-625, 2012.

[12] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees*, et al.*, "Deploying a wireless sensor network on an active volcano," *Internet Computing, IEEE,* vol. 10, pp. 18-25, 2006.

[13] N. Meenakshi and P. Rodrigues, "Tsunami Detection and forewarning system using Wireless Sensor Network-A Survey," 2014.

[14] R. Margesson, "Indian Ocean Earthquake and Tsunami: Humanitarian Assistance and Relief Operations," Congressional Research Service2005.

[15] I. M. o. E. a. Forestry. (2016, 20 May 2016). *Data Luas Kebakaran (Forest Fire Affected Area) 2011-2016*. Available: http://sipongi.menlhk.go.id/hotspot/luas_kebakaran

[16] M. o. H. R. I. Information and Data Center. (2015, 10 May 2016). *Masalah Kesehatan Akibat Kabut Asap Kebakaran Hutan dan Lahan Tahun 2015 (Health Related Problems Caused by Land and Forest Fire Smoke in 2015)*. Available: http://www.depkes.go.id/resources/download/pusdatin/infodatin/infodatin-asap.pdf

[17] J. M. Kahn, R. H. Katz, and K. S. Pister, "Next century challenges: mobile networking for "Smart Dust"," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, 1999, pp. 271-278.

[18] I. C. S. L. M. S. Committee, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," ed: IEEE Std, 1997.

[19] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, 2004, pp. 95-107.

[20] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2002, pp. 1567-1576.

[21] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *Networking, IEEE/ACM Transactions on,* vol. 12, pp. 493-506, 2004.

[22] L. Zhang, S. Ghosh, P. Veeraraghavan, and S. Singh, "An energy efficient wireless sensor MAC protocol with global sleeping schedule," in *Computer Science and its Applications, 2008. CSA'08. International Symposium on*, 2008, pp. 303-308.

[23] S. Ghosh, P. Veeraraghavan, S. Singh, and L. Zhang, "Performance of a wireless sensor network mac protocol with a global sleep schedule," *International Journal of Multimedia and Ubiquitous Engineering,* vol. 4, pp. 99-114, 2009.

[24] Y. Li, W. Ye, and J. Heidemann, "Energy and latency control in low duty cycle MAC protocols," in *Wireless Communications and Networking Conference, 2005 IEEE*, 2005, pp. 676-682.

[25] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks,* vol. 38, pp. 393-422, 2002.

[26] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie, "A self organizing wireless sensor network," in *PROCEEDINGS OF THE ANNUAL ALLERTON CONFERENCE ON COMMUNICATION CONTROL AND COMPUTING*, 1999, pp. 1201-1210.

[27] G. Zhou, L. Huang, W. Li, and Z. Zhu, "Harvesting ambient environmental energy for wireless sensor networks: a survey," *Journal of Sensors,* vol. 2014, 2014.

[28] S. Gao, L. Qian, D. R. Vaman, and Q. Qu, "Energy efficient adaptive modulation in wireless cognitive radio sensor networks," in *2007 IEEE International Conference on Communications*, 2007, pp. 3980-3986.

[29] J. Garzas, C. B. Calzón, and A. G. Armada, "An energy-efficient adaptive modulation suitable for wireless sensor networks with SER and throughput constraints," *EURASIP Journal on Wireless Communications and Networking,* vol. 2007, pp. 23-23, 2007.

[30] I. Khemapech, A. Miller, and I. Duncan, "A survey of transmission power control in wireless sensor networks," in *Proceedings of the 8th Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting (PGNet'07)*, 2007, pp. 15-20.

[31] N. A. Pantazis and D. D. Vergados, "A survey on power control issues in wireless sensor networks," *IEEE Communications Surveys and Tutorials,* vol. 9, pp. 86-107, 2007.
[32] F. Palumbo, P. Barsocchi, C. Gallicchio, S. Chessa, and A. Micheli, "Multisensor data fusion for activity recognition based on reservoir computing," in *International Competition on Evaluating AAL Systems through Competitive Benchmarking*, 2013, pp. 24-35.
[33] A. R. Pinto, C. Montez, G. Araújo, F. Vasques, and P. Portugal, "An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms," *Information Fusion,* vol. 15, pp. 90-101, 2014.
[34] R. Tan, G. Xing, B. Liu, J. Wang, and X. Jia, "Exploiting data fusion to improve the coverage of wireless sensor networks," *IEEE/ACM Transactions on networking,* vol. 20, pp. 450-462, 2012.
[35] X. Liu, "A survey on clustering routing protocols in wireless sensor networks," *Sensors,* vol. 12, pp. 11113-11153, 2012.
[36] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: A survey," *IEEE Communications Surveys & Tutorials,* vol. 15, pp. 551-591, 2013.
[37] A. Liu, Z. Zheng, C. Zhang, Z. Chen, and X. Shen, "Secure and energy-efficient disjoint multipath routing for WSNs," *IEEE Transactions on Vehicular Technology,* vol. 61, pp. 3255-3265, 2012.
[38] V. K. Tiwari and K. Upadhyay, "ENERGY EFFICIENT MULTIPATH ROUTING PROTOCOL FOR MOBILE SENSOR NETWORK," *ENERGY,* vol. 1, 2013.
[39] X. Li, J. Yang, A. Nayak, and I. Stojmenovic, "Localized geographic routing to a mobile sink with guaranteed delivery in sensor networks," *IEEE Journal on Selected Areas in Communications,* vol. 30, pp. 1719-1729, 2012.
[40] C. Tunca, S. Isik, M. Y. Donmez, and C. Ersoy, "Distributed mobile sink routing for wireless sensor networks: a survey," *IEEE Communications Surveys & Tutorials,* vol. 16, pp. 877-897, 2014.
[41] P. Lin, C. Qiao, and X. Wang, "Medium access control with a dynamic duty cycle for sensor networks," in *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE*, 2004, pp. 1534-1539.
[42] S.-C. Choi, J.-W. Lee, Y. Kim, and H. Chong, "An energy-efficient MAC protocol with random listen-sleep schedule for wireless sensor networks," in *TENCON 2007-2007 IEEE Region 10 Conference*, 2007, pp. 1-4.
[43] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: a survey," *IEEE Communications Magazine,* vol. 44, pp. 115-121, 2006.
[44] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," *Wireless Networks,* vol. 12, pp. 63-78, 2006.
[45] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks,* vol. 52, pp. 2292-2330, 2008.
[46] C. C. Enz, A. El-Hoiydi, J.-D. Decotignie, and V. Peiris, "WiseNET: an ultralow-power wireless sensor network solution," *Computer,* vol. 37, pp. 62-70, 2004.
[47] W. Ye, F. Silva, and J. Heidemann, "Ultra-low duty cycle MAC with scheduled channel polling," in *Proceedings of the 4th international conference on Embedded networked sensor systems*, 2006, pp. 321-334.

[48] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*, 2006, pp. 307-320.

[49] Y. Sun, O. Gurewitz, and D. B. Johnson, "RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, 2008, pp. 1-14.

[50] L. Tang, Y. Sun, O. Gurewitz, and D. B. Johnson, "PW-MAC: An energy-efficient predictive-wakeup MAC protocol for wireless sensor networks," in *INFOCOM, 2011 Proceedings IEEE*, 2011, pp. 1305-1313.

[51] T. Van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, 2003, pp. 171-180.

[52] P. C. Uzoh, J. Li, Z. Cao, J. Kim, A. Nadeem, and K. Han, "Energy Efficient Sleep Scheduling for Wireless Sensor Networks," in *Algorithms and Architectures for Parallel Processing*, ed: Springer, 2015, pp. 430-444.

[53] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An adaptive energy-efficient and low-latency MAC for data gathering in wireless sensor networks," in *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*, 2004, p. 224.

[54] S. W. Hussain, T. Khan, and S. H. Zaidi, "Latency and energy efficient MAC (LEEMAC) protocol for event critical applications in wsns," in *Collaborative Technologies and Systems, 2006. CTS 2006. International Symposium on*, 2006, pp. 370-378.

[55] L. Choi, S. H. Lee, and J.-A. Jun, "SPEED-MAC: Speedy and energy efficient data delivery MAC protocol for real-time sensor network applications," in *Communications (ICC), 2010 IEEE International Conference on*, 2010, pp. 1-6.

[56] N. Vasanthi and S. Annadurai, "Energy efficient sleep schedule for achieving minimum latency in query based sensor networks," in *Sensor Networks, Ubiquitous, and Trustworthy Computing, 2006. IEEE International Conference on*, 2006, pp. 214-219.

[57] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *Selected Areas in Communications, IEEE Journal on*, vol. 18, pp. 535-547, 2000.

[58] X. Ma and X. Chen, "Performance analysis of IEEE 802.11 broadcast scheme in ad hoc wireless LANs," *Vehicular Technology, IEEE Transactions on*, vol. 57, pp. 3757-3768, 2008.

[59] J. C.-P. Wang, M. Abolhasan, D. R. Franklin, and F. Safaei, "Characterising the behaviour of IEEE 802.11 broadcast transmissions in ad hoc wireless LANs," in *Communications, 2009. ICC'09. IEEE International Conference on*, 2009, pp. 1-5.

[60] B. Jang and M. L. Sichitiu, "IEEE 802.11 saturation throughput analysis in the presence of hidden terminals," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, pp. 557-570, 2012.

[61] R. Singh, "Performance Analysis of IEEE 802.11 in the Presence of Hidden Terminal for Wireless Networks," in *Computational Intelligence in Data Mining-Volume 1*, ed: Springer, 2015, pp. 665-676.

[62] O. Yang and W. B. Heinzelman, "Modeling and performance analysis for duty-cycled MAC protocols with applications to S-MAC and X-MAC," *Mobile Computing, IEEE Transactions on*, vol. 11, pp. 905-921, 2012.

[63] D. Dutta, A. Karmakar, and D. K. Saikia, "An Analytical Model for IEEE 802. 15. 4/ZigBee Wireless Sensor Networks with Duty Cycle Mechanism for Performance Prediction and Configuration of MAC Parameters to Achieve QoS and Energy Efficiency," *International Journal of Computer Applications,* vol. 102, 2014.

[64] I. Al-Anbagi, M. Khanafer, and H. T. Mouftah, "A realistic and stable Markov-based model for WSNs," in *Computing, Networking and Communications (ICNC), 2013 International Conference on*, 2013, pp. 802-807.

[65] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *ACM SIGOPS Operating Systems Review,* vol. 36, pp. 147-163, 2002.

[66] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, 2003, pp. 138-149.

[67] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, 2004, pp. 39-49.

[68] K. S. Yildirim and A. Kantarci, "Time synchronization based on slow-flooding in wireless sensor networks," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 25, pp. 244-253, 2014.

[69] W. Su and I. F. Akyildiz, "Time-diffusion synchronization protocol for wireless sensor networks," *Networking, IEEE/ACM Transactions on,* vol. 13, pp. 384-397, 2005.

Glossary

Active (mode)	A fashion of nodes in WSNs operating with their radio on
Border node	Nodes that have at least a one-hop neighbour that is a member of a different cluster
Cluster	A group of neighbouring nodes that operates on a same schedule
Convergence time	The duration of a merging process
Cycle	A group of frames. Nodes send their SYNC once in every cycle
Destination cluster	A cluster that is the destination of a meging process
Frames	A duration that covers an active period and a sleeping period in synchronized duty cycle protocol
Merging cluster	A cluster that is in merging with a destination cluster
Merging process	A process in which nodes from a cluster join a newly discovered cluster based on a schedule selection algorithm
Sleep (mode)	A fashion of nodes in WSN operating with their radio off
Sleep Schedule	A schedule that determines the time a node switch from active to sleep mode and the other way around
SYNC	A control packet in synchronized duty cycle protocols that contains the information of a schedule
SYNC frame	A randomly picked frame in each cycle, in which each node contends for sending its SYNC
Synchronization cycle	A randomly picked cycle (with a group of 10 cycles), in which each node operates in active mode during the whole cycle duration.
SYNC-M	A control packet in the that contains the information of a schedule
Virtual cluster	See Cluster
Virtual frame	A group of two frames
Wake-up (mode)	See Active mode