A Study into Prolonging Wireless Sensor Network Lifetime during Disaster Scenarios

by

Ansar Jamil

A Doctoral Thesis

Submitted in partial fulfilment
of the requirements for the award of

Doctor of Philosophy
of
Loughborough University

16th September 2014

Abstract

A Wireless Sensor Network (WSN) has wide potential for many applications. It can be employed for normal monitoring applications, for example, the monitoring of environmental conditions such as temperature, humidity, light intensity and pressure. A WSN is deployed in an area to sense these environmental conditions and send information about them to a sink. In certain locations, disasters such as forest fires, floods, volcanic eruptions and earth-quakes can happen in the monitoring area. During the disaster, the events being monitored have the potential to destroy the sensing devices; for example, they can be sunk in a flood, burnt in a fire, damaged in harmful chemicals, and burnt in volcano lava etc. There is an opportunity to exploit the energy of these nodes before they are totally destroyed to save the energy of the other nodes in the safe area. This can prolong WSN lifetime during the critical phase. In order to investigate this idea, this research proposes a new routing protocol called Maximise Unsafe Path Routing Protocol (MUP) routing using IPv6 over Low power Wireless Personal Area Networks (6LoWPAN). The routing protocol aims to exploit the energy of the nodes that are going to be destroyed soon due to the environment, by concentrating packets through these nodes. MUP adapts with the environmental conditions. This is achieved by classifying four different levels of threat based on the sensor reading information and neighbour node condition, and represents this as the node health status, which is included as one parameter in the routing decision. High priority is given to a node in an unsafe condition compared to another node in a safer condition. MUP does not allow packet routing through a node that is almost failed in order to avoid packet loss when the node fails. To avoid the energy wastage caused by selecting a route that requires a higher energy cost to deliver a packet to the sink, MUP always forwards packets through a node that has the minimum total path cost. MUP is designed as an extension of RPL, an Internet Engineering Task Force (IETF) standard routing protocol in a WSN, and is implemented in the Contiki Operating System (OS). The performance of MUP is evaluated using simulations and test-bed experiments. The results demonstrate that MUP provides a longer network lifetime during a critical phase of typically about 20% when compared to RPL, but with a trade-off lower packet delivery

ratio and end-to-end delay performances. This network lifetime improvement is crucial for the WSN to operate for as long as possible to detect and monitor the environment during a critical phase in order to save human life, minimise loss of property and save wildlife.

Acknowledgements

First and Foremost, thanks to Allah, the Most Gracious, the Most Merciful for giving me strength and calmness during my life as a PhD student.

My sincere thanks and appreciation go to my supervisors, Prof. David Parish, Dr Iain Phillips, and Prof. Raphael Phan for their support, encouragement and guidance throughout the journey of my PhD research. During the three and a half years of this study they have provided me with constructive criticism, comments, and advice on my research work. They are my inspiration to be successful in my career as an academician and researcher.

I would also like to thank Dr John Whitley and Dr George Oikonomou for their help and advice in this research project, particularly their involvement in research discussions, Sensinode matters, the Contiki OS and the usage of LaTeX. I am also very thankful to my colleagues in the High Speed Networks group and the NETS group for their help and knowledge sharing.

Lastly, I would like to show appreciation to my family: my parents for always praying for my success in my PhD study, my children who kept me smiling during the tough times in the pursuit of a PhD, and especially my beloved wife, who has been supportive and encouraging throughout my PhD journey even though we have to stay thousand miles away from our home country.

Contents

List of Figures

List of Tables

List of Abbreviations

6LoWPAN IPv6 over Low power Wireless Personal Area Networks.

BH Back-up Cluster Header.

CCA Clear Channel Assessment.

CH Cluster Head.

CPU Central Processing Unit.

CSMA/CA Carrier Sense Multiple Access Collision Avoidance.

CTS Clear-To-Send.

DAO Destination Advertisement Object.

DAO-ACK Destination Advertisement Object Acknowledgement.

DGRM Directed Graph Radio Medium.

DIO DODAG Information Object.

DIS DODAG Information Solicitation.

DODAG Destination Oriented Directed Acyclic Graph.

DSSS Direct Sequence Spread Spectrum.

ETX Estimation Transmission Count.

FFD Full Function Device.

GPS Global Positioning System.

ICMP Internet Control Message Protocol version 6.

IEEE Institute of Electrical and Electronic Engineering.

IETF Internet Engineering Task Force.

IP Internet Protocol.

IPv4 Internet Protocol version 4.

IPv6 Internet Protocol version 6.

IR Infra-Red.

LLN Low Power and Lossy Network.

LPM Low Power Mode.

LR-WPAN Low-Rate Wireless Personal Area Network.

MAC Medium Access Control.

MCU Micro Controller Unit.

MEMS Micro-Electromechanical Systems.

MP2P Multipoint-to-Point.

MRHOF Minimum Rank Objective Function with Hysteresis.

MTU Maximum Transmission Unit.

MUP Maximise Unsafe Path Routing Protocol.

OF Objective Function.

OS Operating System.

P2MP Point-to-Multipoint.

P2P Point-to-Point.

PAN Personal Area Network.

PHY Physical Layer.

QoS Quality of Service.

RAM Random Access Memory.

RDC Radio Duty Cycle.

RFD Reduced Function Device.

RPL IPv6 Routing Protocol for Low Power and Lossy Networks.

RSSI Received Signal Strength Indicator.

RTS Request-To-Send.

SNR Signal-to-Noise Ratio.

SRAM Static Random Access Memory.

TCP Transport Control Protocol.

UAV Unmanned Ariel Vehicles.

UDGM Unit Disk Graph Medium.

UDP User Datagram Protocol.

WSN Wireless Sensor Network.

Chapter 1

Introduction

1.1 Background

A WSN is defined as a large collection of small wireless devices called sensor nodes that can organise themselves into an ad-hoc network. They have constraints of energy, processing and communication resources [75]. These wireless devices are deployed in close proximity to the phenomenon to gather information about the physical world and send it to a sink or base station. This technology is suitable for environmental data collection systems that enable a user to monitor environmental conditions effectively from a distance. Several studies have been carried out to integrate WSNs with the Internet which will provide more functionality such as requesting data, sending notifications, and monitoring sensor readings via email [44] and the web [17].

Sensor nodes can be attached with many different types of sensor, such as magnetic, seismic, infrared, thermal, acoustic, visual, and radar, which are able to monitor a wide variety of ambient conditions [29]. In addition, the self-organising feature makes WSN technology suitable to be deployed in many applications such as agriculture, animal tracking, the military, petroleum pipeline monitoring, patient-health monitoring and forest fire monitoring systems. The deployment of a WSN needs to be done carefully to meet the desired performance of an application. This is because a WSN has many constraints compared to a traditional computer network.

Sensor nodes in a WSN have two obvious resource limitations in terms of energy sources and memory storages. The sensor nodes have a limited amount of energy because they are reliant on batteries as an energy source. Once sensor nodes are deployed in an area, their batteries cannot be replaced or recharged easily. Therefore, the battery charge must be conserved to extend the life of the individual sensor node and hence the entire network. The sensor nodes have a

small amount of memory and storage space for program codes. For example, a MICAZ node has an 8-bit, 7.37 MHz CPU with only 4 kB SRAM and 128 kB program Flash memory. Other than this, a N740 Nano-Sensor node has an 8-bit MCU with 128 kB Flash and 8 kB RAM. The size of the developed program codes for the sensor nodes must also be quite small.

Normally, WSNs use unreliable connectionless communication when delivering a packet from a node to the sink to avoid the energy wastage from dedicated routes in connection-oriented communication. WSNs provide their best effort to deliver a packet to the sink successfully. Due to the unreliable wireless communication link, a packet may get corrupted due to channel errors. This unreliable wireless communication link is highly probabilistic and asymmetric. It depends on the transmission power and the distance travelled by a packet [106]. Other than this, a packet can also become corrupted due to transmission interference, which occur when two nodes within the same coverage area transmit packets simultaneously. In both cases, the packet transfer fails and the sender node needs to retransmit the packet. If the number of retransmissions has reached the maximum value, the sender node simply drops the packet. A packet may also be dropped due to congestion at a highly congested node. It is very important in a WSN to have appropriate mechanisms to handle these situations in order to reduce the probability of packet loss.

1.2 Problem Statement

A WSN is suitable for use in a wide area of applications. A WSN is deployed to fulfil a specific task, which may be different for each application: for example, detecting forest fire, monitoring soil humidity, monitoring room temperature, and tracking wild animals. Each application has specific requirements in terms of security level, Quality of Service (QoS) and the type of data collected. Thus, a WSN should be designed specifically to meet the individual application requirements. In order to achieve this, a large number of communication protocols and network solutions should be examined in the process of constructing an optimal WSN infrastructure before practical deployment [30]. The main challenge is to achieve an acceptable network performance for a desired task, given the limitations and constraints of WSN technology.

A major issue which has been frequently emphasised in many research studies in this field is the problem of limited energy. It is important for the WSN to be available throughout the intended time of deployment. For example, in critical situations, a WSN must always be ready to detect any critical event. If the sensor network becomes non-functional, the network can no longer detect and monitor

the critical event. This situation can cause many unwanted events to happen, such as loss of human life, property, public infrastructure and valuable wildlife when the critical event becomes uncontrollable. It is crucial for researchers in this field to develop new protocols that are able to prolong the network lifetime of a WSN. Since network lifetime is directly influenced by energy, the power efficiency often turns out to be a major performance metric. In the WSN, the power consumption of a sensor node can be divided depending on each component: sensing, communication and data processing. Among these components, the communication components have the highest energy consumption, which includes the activities of transmitting and listening for a packet.

Besides this, the two performance metrics that should be considered during the deployment of the WSN are the packet delivery ratio and end-to-end delay. It is expected that the WSN to have a high packet delivery ratio performance in order to provide sufficient information to achieve the monitoring aims. If the node is located further away and does not have a direct link of communication with the sink, the network must deliver it using multihop communication. This is a challenging task because the WSN has an unreliable communication link with limited bandwidth, which is heavily influenced by its lossy environment [106, 13]. This problem with the communication link can be solved by implementing the hop-by-hop basis of data transfer, where for each successful transmission the receiving node must send an acknowledgement message to the sender node. Because of this, the end-to-end delay performance in the WSN becomes unpredictable. The WSN must be able to deliver the packet as soon as possible because, for certain applications, the data is only valid for a short period of time.

1.3 Objectives

A common application of a WSN is environment monitoring. Here, a WSN is deployed in an area to sense environmental conditions, such as temperature, humidity, light intensity and pressure, and collect the sensed information. In certain deployment areas, sudden disasters such as forest fires, floods, volcanic eruptions and earth-quakes can happen during monitoring. During such a disaster phase, the events being monitored have the potential to destroy the sensing devices; for example, they can be sunk in a flood, burnt in a fire, damaged in harmful chemicals, and burnt in volcano lava etc. There is an opportunity to exploit the energy of these doomed nodes before they are totally destroyed to save the energy of other nodes and prolong network lifetime during the disaster phase.

In order to investigate this idea, the present research proposes the Maximise Unsafe Path (MUP) routing protocol. MUP aims to exploit the energy of these

dying nodes by concentrating packets through the nodes before they are totally destroyed in order to save the energy of the other nodes, to provide a longer network lifetime during disaster situations. MUP is also expected to provide a similar packet delivery ratio and end-to-end delay performance. MUP is designed specifically for normal monitoring applications. MUP adapts its routes to the environments by using sensing information from the sensors, which is included as one of the parameters during the selection of routes in the networks. The research work is focused on the development and evaluation of MUP based on IEEE 802.15.4 beaconless operation and 6LoWPAN architecture, which becomes the popular choice for various monitoring applications.

1.4 Scopes

The scope of the research is divided into three technical phases which include the design of MUP, a simulation study of MUP and an experimental study of MUP in WSN test-beds.

1.4.1 Design of MUP

The MUP design concept consists of three functional modules that include routing management, neighbourhood management and critical event detection modules. These functional modules cooperate to prolong the network lifetime of the WSN during disaster situations.

One approach for developing MUP is by reusing an existing routing protocol. MUP is designed as an extension to the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL) [100], an IETF standard routing protocol in WSN. The implementation and development of the MUP design is based on the ContikiRPL platform [48], which is the implementation of RPL in the Contiki OS. The routing management module is implemented by introducing a new Objective Function (OF). The critical event detection is developed as a new module. The neighbourhood management module uses the existing Neighbour Discovery available in RPL. Here, two implementations of MUP are proposed: MUP-single and MUP-adapt.

The characteristics of MUP have been studied to ensure that the implementation is carried out correctly and that the routing protocol performs as expected. This step is very important before intensive measurement to determine the performance of MUP via simulation.

1.4.2 A Simulation Study of MUP

In the development of MUP, COOJA is selected as the design and evaluation tool. COOJA is a simulator for the Contiki OS which enables cross-level simulation, i.e. simultaneous simulation at many levels of the systems [71]. It is flexible and extensible in that all levels of the system can be changed and replaced, including the sensor node platforms, the operating system software, the radio transceivers, and the radio transmission models. MUP is simulated based on the Contiki OS which includes 6LoWPAN functionality. In addition,Carrier Sense Multiple Access Collision Avoidance (CSMA/CA) is chosen as the Medium Access Control (MAC) layer protocol. For the physical layer, the simulation uses the sensor node model which is based on the IEEE 802.15.4 physical layer standard. Other than this, the simulator has a built-in plugin to include any disaster scenario. Since the forest fire scenario is selected as an example of disasters, the plugin loads the scenario that consists of temperature increment information for each node in the network into the simulation. The performance of MUP, such as network lifetime, energy consumption, packet delivery ratio, packet loss and end-to-end delay, is studied and compared with RPL.

1.4.3 An Experimental Study of MUP in WSN Test-beds

MUP is tested in WSN test-beds to determine the routing performance in the real environment. The WSN test-beds consist of sensor nodes which are called N740 Nano-Sensor, manufactured by the SENSINODE LTD company. The sensor node is user programmable and operates at 2.4 GHz frequency using an 8051 MCU, an 8-bit MCU with 128kB Flash and 8kB RAM [83]. The Contiki OS which is used in the simulation can be programmed into the sensor node. In the experiment, two different sizes of WSN testbeds are established consisting of 10 sensor nodes (small testbed) and 25 sensor nodes (large testbed) distributed on the field. Each node sends data periodically to the sink. Based on the captured data, the performance of MUP in the experiment is determined and compared with the simulation.

1.5 Significance of the Research Work

MUP is designed for normal monitoring applications. MUP is able to prolong the network lifetime of a WSN during the disaster phase. It is very important for the network to keep functioning for as long as possible, especially to detect and monitor any disaster events. These disaster events are unpredictable and can happen anywhere at any time in the network. For example, in forest fire detection and monitoring systems, the network must be available to detect any

fire happening and send an alert to fire fighters as soon as possible. The forest fire can be put out easily even with a small squad of fire fighters with just basic equipment. If the network is not available, an undetected fire can grow larger until the stage where it becomes uncontrollable, which is too risky for fire fighters to handle and may endanger their life. Thousands of hectares of valuable forest will be burnt in a fire. This forest can consist of a diversity of small and large trees, which are places where many animal species live. To make things worse, the fire could cause loss of civilian life, houses, properties and other facilities in the affected area.

1.6 Thesis Contributions

The contributions of the thesis are outlined as below:

1. Study a new idea for prolonging the network lifetime in WSN. In certain deployment areas, disasters such as forest fires, floods, volcanic eruptions and earth-quakes can happen in the monitoring area. During the disaster phase, the events being monitored have the potential to destroy the sensing devices; for example, they can be sunk in a flood, burnt in a fire, damaged by harmful chemicals, and burnt in volcano lavaetc. There is an opportunity to exploit the energy of these nodes before they are totally destroyed to save the energy of the other nodes in the safe area. This can prolong the WSN lifetime during the disaster phase. In order to study this idea, a new routing protocol is proposed called the Maximise Unsafe Path (MUP) routing protocol.

2. Design and development of the MUP routing protocol. MUP is a new routing protocol in WSN. It is designed to be used in normal monitoring applications. The main objective of MUP is to prolong the network lifetime of WSNs and still provide comparable network performances during the disaster phase. In order to achieve this, MUP exploits the energy of unsafe nodes that are going to be damaged soon, to save the energy of the other nodes in the network by routing packets through the unsafe nodes. The main feature in MUP is the ability to adapt alongside the environment, which is achieved by taking consideration of the environmental conditions in the routing decision. A new parameter is introduced in the MUP routing design to represent the environmental threat to a sensor node, which is called health status. Determination of health status is based on the available information from the sensor reading and the neighbour nodes' condition. MUP uses the health status and total path cost information as parameters to make routing decisions. First, a MUP node finds neighbours with the lowest total path

cost. If two or more neighbours are found, the node selects from them based on their health status by giving the highest priority to the neighbour in an unsafe condition rather than other neighbours in a safe condition. If they are still tied, the routing algorithm remains with the current best neighbour as the forwarding node. In order to avoid packet loss when a neighbour is just damaged, the node removes the almost failed neighbour as its forwarding node. MUP routing is designed for 6LoWPAN network. MUP is implemented as an extension to RPL.

3. Study of MUP in simulation and experiment. A forest fire scenario is taken as an example of a disaster situation. The simplest circular shape of the forest fire growth model for a flat terrain without wind is used in both the simulation and the experiment. When a node becomes exposed to fire, a linearly growing offset is added to the node's temperature value. In the simulation, the characteristics and performance of MUP are studied in perfect and lossy network conditions. After that, the simulation findings are verified by experiment. In the experiment, MUP is tested using a WSN test-bed, which is deployed on a field. Through this study, the findings demonstrate that MUP prolongs the network lifetime of the WSN during the disaster phase when compared to RPL. The findings also indicate that MUP suffers a lower packet delivery ratio and end-to-end delay performance when compared to RPL.

1.7 Thesis Organisation

This thesis consists of eight chapters. Chapter 1 serves as an introduction to the thesis. It covers topics such as the statement of the research problem, the objectives of the research, the scope of the research and the significance of the research.

Chapter 2 provides the relevant background about the WSN, IEEE 802.15.4 and 6LoWPAN. The chapter introduces the applications of the WSN, the constraints in the WSN, the routing challenges in the WSN and the routing categories in the WSN.

Chapter 3 describes the MUP routing protocol design which is based on RPL. MUP introduces two new modules which are the OF and the critical event detection module. The OF module defines how nodes select and optimise routes within the network based on the routing algorithm. The critical event detection module is very important to determine node health status, which represents the environmental threat to the sensor node. Other than these modules, MUP continues

using the available modules in RPL such as the neighbour discovery mechanism and the routing update mechanism.

Chapter 4 describes the Contiki OS that is selected as the operating system for sensor nodes in both simulation and experiment. This chapter also explains the COOJA simulator, the simulation tool for developing and designing MUP. It includes an explanation of the forest fire scenario that has been introduced in the simulation. The basic network configuration is also described in detail in this chapter. In addition, this chapter defines the performances metrics used in the simulation and experiment.

Chapter 5 describes the simulation results and the analysis for MUP. It includes the performance for both implementations of MUP: MUP-single and MUP-adapt. For performance comparison, this chapter includes the simulation results and the analysis for SAFEST and RPL.

Chapter 6 describes the hardware implementation of MUP and RPL using N740 Nano-Sensor. Both routing protocols have been tested in two different sizes of WSN test-beds consisting of 10 nodes (small test-bed) and 25 nodes (large test-bed). In order to achieve multi-hop communication, the transmission range of each node is limited to its adjacent neighbours. Experimental results from the hardware implementation are compared with the simulation results.

Finally, Chapter 7 concludes the thesis with a summary of the work that has been done, along with suggestions for future work.

Chapter 2

Background and Related Work

2.1 Introduction of WSN

Recent advancement in micro-electromechanical systems (MEMS) technology, wireless communications, and digital electronics have enabled the development of low-cost, low-power, multi-functional sensor nodes that are small in size and communicate in short distances [2]. These tiny sensor nodes, which consist of sensing, data processing, and communicating components, leverage the idea of sensor networks based on the collaborative effort of a large number of nodes [2]. Also, the low cost of the sensors makes it possible to have a network of hundreds or thousands of these wireless sensors, thereby enhancing the reliability and accuracy of the data and the area coverage as well.

WSN networks are affected by many challenging issues such as sensor nodes deployment, data processing and routing, data security, fault tolerance, data aggregation and connectivity [4, 16]. These challenges arise primarily due to the large number of constraints such as limited energy, memory, computational speed and bandwidth [77].

2.2 Architecture of WSN

Figure 2.1 shows the WSN architecture. The main entities that build up the WSN architecture are described below [90]:

- *The sensor nodes*: These are small devices that form the sensor network. The main functionalities of a sensor node are sensing the phenomenon within its coverage area, forming a network by communicating with other nodes over a wireless medium and routing the sensor reading data to the user via a base station or sink.

- *The base station (sink)*: The base station communicates with the user via internet or satellite communication. It is located near the sensor field. The data collected from each sensor node is sent to the user by a multi-hop communication method (due to the low power of the sensor nodes) through the base station.
- *Phenomenon*: This is an entity that is of interest to the user. The sensor node has the capability to sense the phenomenon and also to do simple analysis on it before sending the data to the user.
- *The user*: This is a person that is interested in obtaining information about a specific phenomenon to measure or monitor its behaviour.

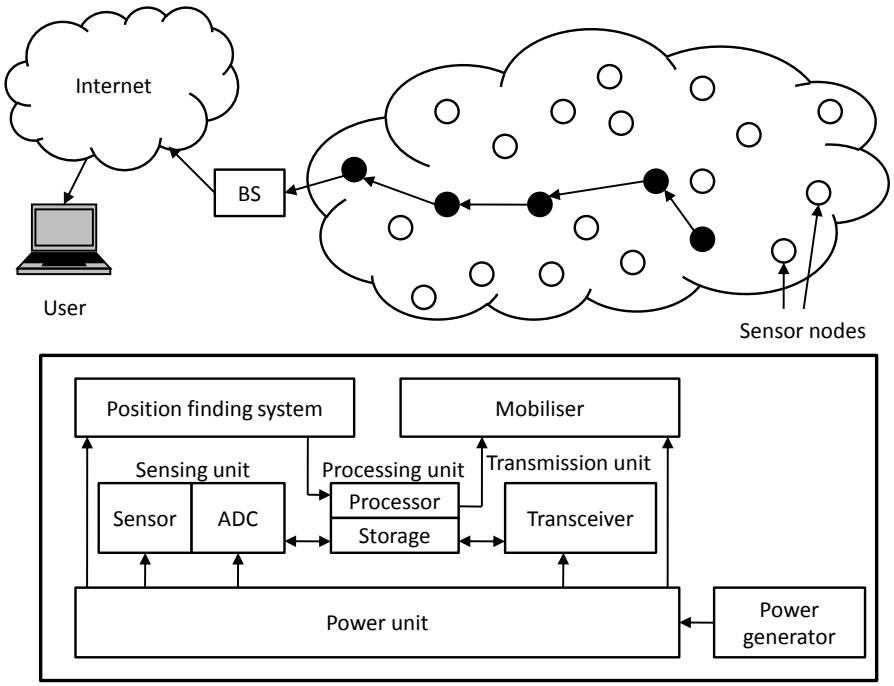


Figure 2.1: WSN architecture [2].

2.2.1 IEEE 802.15.4 Specification

IEEE 802.15.4 is a standard which defines the Physical Layer (PHY) and MAC sub-layer specifications for Low-Rate Wireless Personal Area Network (LR-WPAN) [41]. This standard was not specifically developed for WSN, but WSN can be built up from the LR-WPAN. The IEEE 802.15.4 protocol targets low data rate, low power consumption, low cost wireless networking which meet the requirement of the WSN.

Based on the IEEE 802.15.4 standard, a LR-WPAN can support two different types of device: the Full Function Device (FFD) and the Reduced Function Device

(RFD). The FFD is a device that supports three operation modes; one of these acts as a Personal Area Network (PAN) coordinator. A PAN coordinator identifies its own network such that other devices may be associated. A LR-WPAN must include at least one FFD acting as a PAN coordinator. An FFD can also act as a coordinator that provides synchronisation services through the transmission of beacons but does not create its own network. Other than this, an FFD can just be a simple device which does not implement the previously mentioned functionalities. The RFD is a device operating with the minimal implementation of the IEEE 802.15.4 protocol. An RFD supports simple applications such as environment sensing using temperature, light or infra-red sensors, which do not require the device to send large amounts of data.

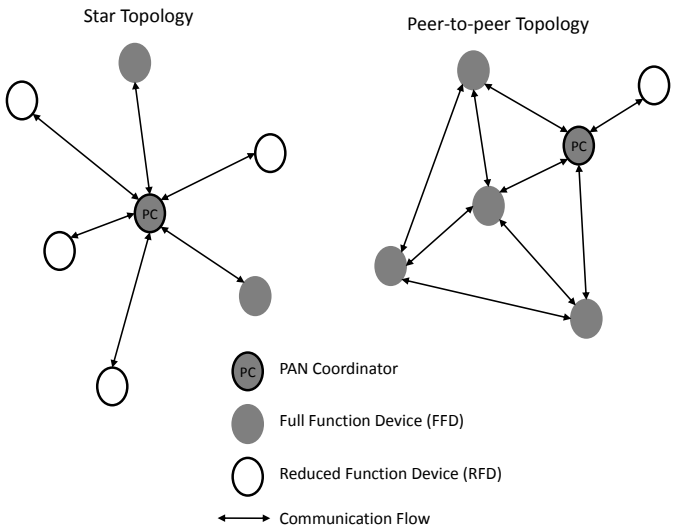


Figure 2.2: Star and peer-to-peer topology [41].

There are two basic types of network topology specified in the IEEE 802.15.4 standard: the star topology and the peer-to-peer topology. Both topology examples are shown in Figure 2.2. In the star topology, a node operates as the PAN coordinator. The PAN coordinator chooses a PAN identifier which is not currently used by any other network. Each of the devices, either FFD or RFD, joining the network can communicate with other devices, and must send its data through the PAN coordinator. The PAN coordinator becomes the centre of communication among the devices in the network. This means that the PAN coordinator may be mains powered due to the power consuming tasks of the PAN coordinator in the star topology, while the other devices are more likely to be battery powered.

In the peer-to-peer topology, a device must be selected as the PAN coordinator, a decision which is based on, for instance, the first device to communicate on the channel. The communication paradigm in the peer-to-peer topology is decentralised. This means that each of the devices can communicate directly

with the other devices in its radio communication range. This topology provides networking flexibility, but it requires an additional complexity for providing an end-to-end connectivity between all the devices in the network. The peer-to-peer topology operates in an ad-hoc manner, and implements multi-hops communication to transfer data from any device to any other device. However, these functions must be defined at the network layer which is not considered in the IEEE 802.15.4. In contrast with the star topology, the resource usage is fairer in the peer-to-peer topology since the communication process does not rely on a particular node [50].

Table 2.1: Physical layer description in IEEE 802.15.4 [41].

Property	Range
Raw data rate	868 MHz: 20 kb/s; 915 MHz: 40 kb/s; 2.4 GHz: 250 kb/s
Range	10-20 m
Latency	Down to 15 ms
Channels	868 MHz: 1 channel; 915 MHz: 10 channels; 2.4 GHz: 16 channels
Frequency band	Two PHYs: 868 MHz/915 MHz and 2.4 GHz
Addressing	Short 16-bit or 64-bit IEEE
Temperature	Industrial temperature range −40 to +85 °C

The IEEE 802.15.4 physical layer offers three operational frequency bands: 868 MHz, 915 MHz and 2.4 GHz. There is a single channel between 868 and 868.6 MHz, 10 channels between 902 and 928 MHz, and 16 channels between 2.4 and 2.4835 GHz as shown in Figure 2.3. The operating frequency bands are defined with specific data rates, which are 20 kb/s at 868 MHz, 40 kb/s at 915 MHz and 250 kb/s at 2.4 GHz. Lower frequencies are more suitable for longer transmission ranges due to lower propagation losses. However, high data rate transmission provides higher throughput, lower latency and lower duty cycles. All these frequency bands are based on a modulation technique called the Direct Sequence Spread Spectrum (DSSS). The specification of IEEE 802.15.4 physical layers are summarised in Table 2.1.

The MAC sub-layer of the IEEE 802.15.4 protocol provides an interface between the physical layer and the higher layer protocols of LR-WPANs. It has many common features with the MAC sub-layer of the IEEE 802.11 protocol, such as the use of CSMA/CA (a channel access protocol), and the support of contention-free and contention-based periods. However, the standard does not include the request-to-send (RTS) and clear-to-send (CTS) mechanism to reduce the probability of collisions, which are more likely to happen in low data rate networks. Figure 2.4 shows a structure for the operational modes for the MAC sub-layer of IEEE 802.15.4. The MAC protocol supports two operational modes that may be

selected by the coordinator as described below:

1. *Beacon-enabled mode*: The coordinator generate beacons periodically to synchronise the attached devices and to identify the PAN. The first part of a superframe is a beacon frame, which also embeds all data frames exchanged between the nodes and the PAN coordinator. Data transmissions between the nodes are also allowed during the superframe duration.

2. *Non beacon-enabled mode*: The devices can simply use unslotted CSMA/CA to send their data by. A superframe structure is not used in this mode.

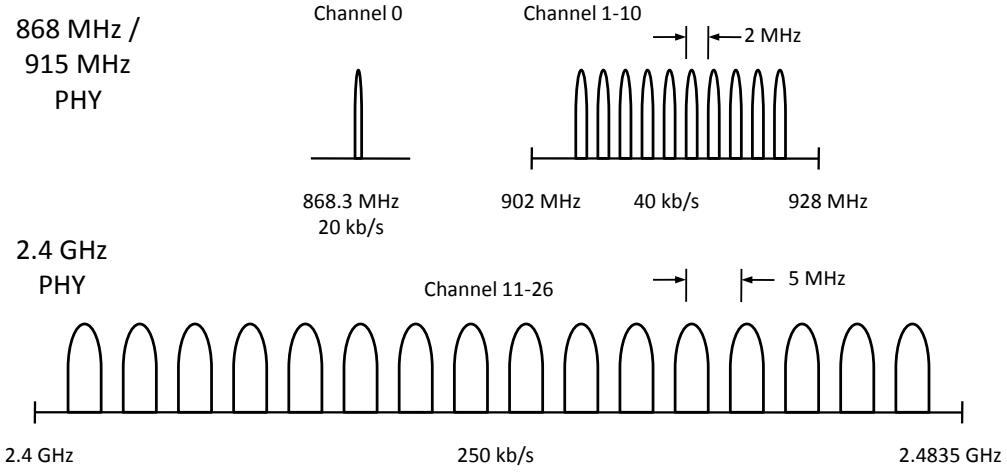


Figure 2.3: Operating Frequency Bands in IEEE 802.15.4 [50].

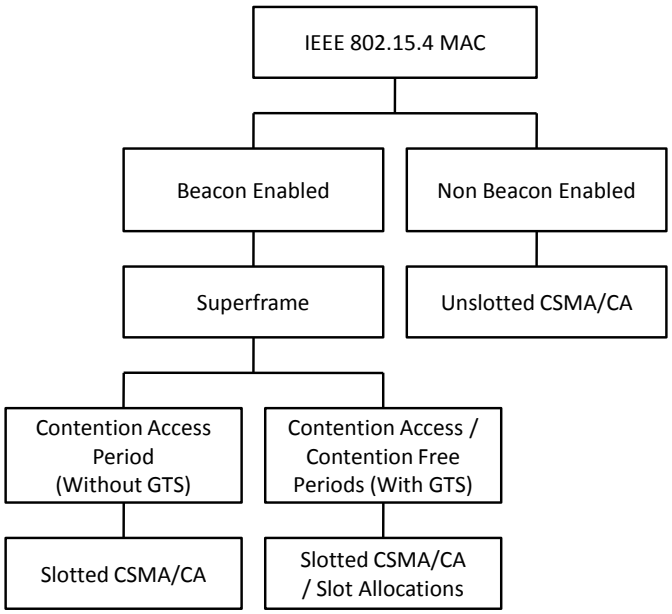


Figure 2.4: Operational modes for MAC sub-layer of the IEEE 802.15.4 [50].

2.2.2 6LoWPAN

The Internet Protocol version 6 (IPv6) is designed to supersede Internet Protocol version 4 (IPv4) and enable the Internet to scale for decades to come. IPv6 expands the IP address from 32 to 128 bits. This means that IPv6 provides about 2^{128} or approximately 3.4×10^{38} address spaces which is significantly more than the IPv4 with 2^{32} or 4, 294, 967, 296 address spaces. Since there is a need to assign thousands of sensor nodes with IP addresses to enable simple interconnectivity to the other IP networks, including the internet [53], IPv6 becomes the best choice for WSN. In addition, the stateless address auto-configuration simplifies the configuration and management of IPv6 devices by enabling sensor nodes to assign themselves meaningful addresses. These features make IPv6 better suited for WSN, especially for large scale deployment.

As described in section 2.2.1, IEEE 802.15.4 is a standard designed specifically for long-lived applications that require numerous low-cost nodes, and has constraints of limited capability of the links and MCU. The data rate is limited to 250 kb/s in the 2.4 GHz band and 20 or 40 kb/s in other frequency bands. The frame length is limited to 128 bytes to ensure a reasonably low packet error. Other than this, it is known to have limited buffering capabilities. The IEEE 802.15.4 defines short 16-bit link addresses in addition to 64-bit addresses to reduce the header overhead and memory requirement. The communication range is short due to its low transmit power. The associated microcontroller typically has about 8 kB of data RAM and 64 kB program ROM.

Because of these resource constraints, supporting IPv6 over WSN presents several challenges. One of them is that IPv6 datagrams are not fit for WSN. The IEEE 802.15.4 frame is one-tenth of the size of the IPv6 minimum MTU requirement, which makes datagram fragmentation and compression essential for efficient operation. It starts from a maximum physical layer packet size of 127 bytes. For the worst case, the link header requires about 46 bytes (25 bytes of physical layer overhead and 21 bytes of overhead in media access control using AES-CCM-128), which leaves 81 bytes for the IPv6 payload. The IPv6 header requires 40 bytes, which leaves 41 bytes. Other than this, the transport layer header must be deducted from the remaining 41 bytes which leaves a very short payload. If UDP is used (which requires 8 bytes header), this leads to 33 bytes for the payload. If TCP is used (which requires 20 bytes), this leaves 21 bytes for the payload.

The adaptation layer must be provided to comply with the IPv6 requirements of a minimum MTU. It is expected that most applications of IEEE 802.15.4 will not use such large packets, and small application payloads in conjunction with the proper header compression will produce packets that fit within a single IEEE

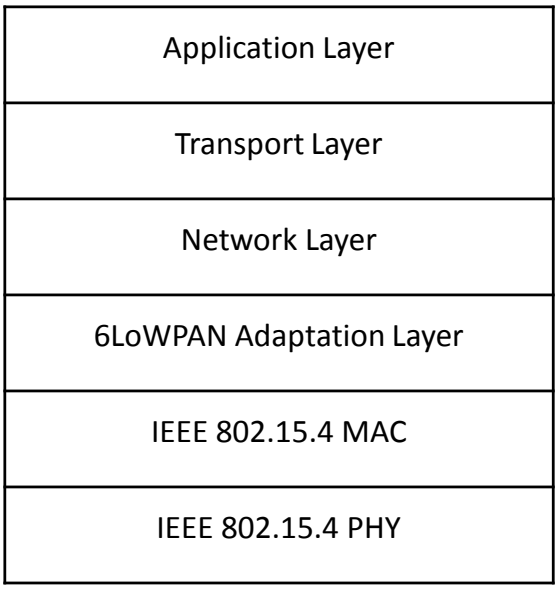


Figure 2.5: 6LoWPAN adaptation layer.

802.15.4 frame [53]. For certain applications, it is quite likely that the packet size requires a small number of fragments. The 6LoWPAN [65] defines the format of how the IPv6 is carried in the IEEE 802.15.4 frames and specifies key elements in the adaptation layer. The 6LoWPAN adaptation layer is illustrated in Figure 2.5. The key concept applied throughout the 6LoWPAN adaptation layer is that it uses stateless compression, which elides the adaptation, network and transport layer header fields by compressing them down to a few bytes. 6LoWPAN has three primary elements:

- **Header compression**: IPv6 header fields are eliminated from a packet when the adaptation layer can derive them from the link-level information carried in the IEEE 802.15.4 frame or based on simple assumptions of shared context.
- **Fragmentation**: IPv6 packets are fragmented into multiple link-level frames to accommodate the IPv6 minimum MTU requirement.
- **Layer-two forwarding**: To support layer-two forwarding of IPv6 datagrams, the adaptation layer can carry link-level addresses for the ends of an IP hop. Alternatively, the IP stack might accomplish intra-PAN routing via layer-three forwarding, in which each IEEE 802.15.4 radio hop is an IP hop.

Similar to IPv6, the 6LoWPAN adaptation layer makes use of header stacking. There are three types of sub-headers which are supported by the 6LoWPAN: *mesh addressing header*, *fragmentation header* and *compression header*. The header

stack is simple to parse and support stateless compression. Each type of sub-header is added only when needed. The fragmentation header is not added for small datagrams, which indicates that a single frame carries the entire payload. Similarly, the mesh header is not added when 6LoWPAN frames are delivered over a single hop radio, since the path source and destination are identical in the link layer header. Figure 2.6 shows typical header stacks and detail for each sub-header.

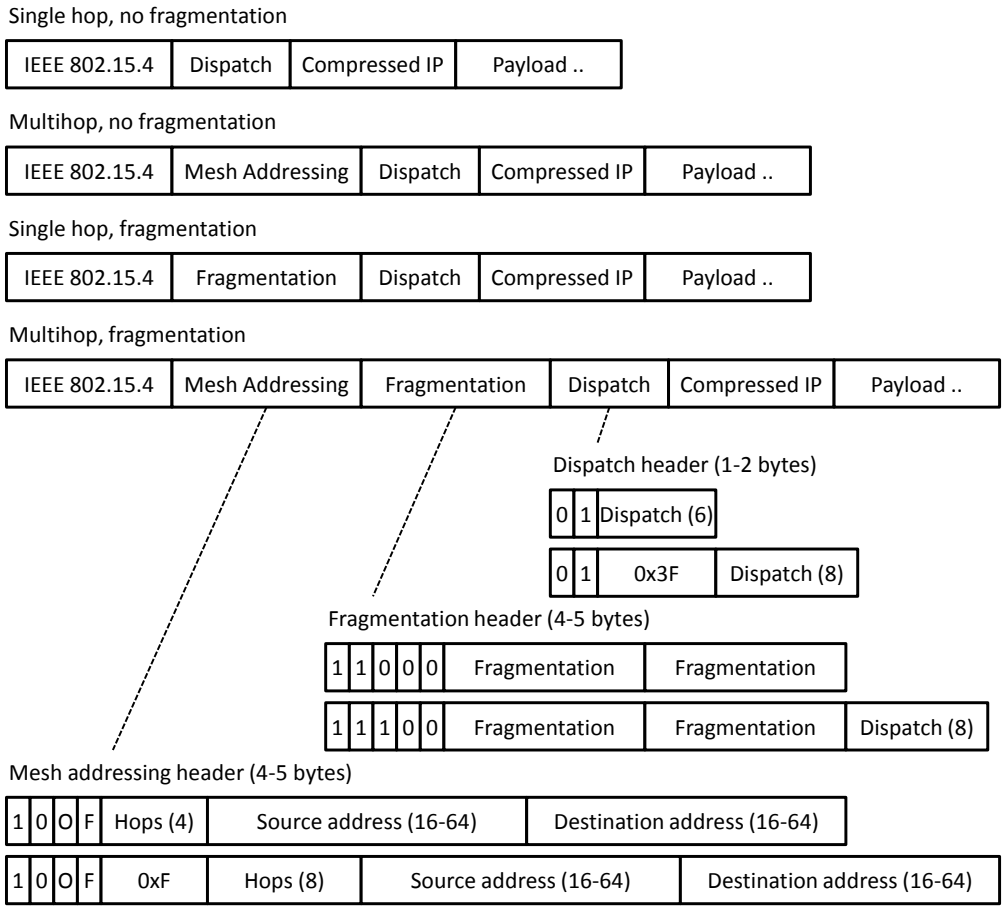


Figure 2.6: 6LoWPAN header stack [40].

2.3 Applications of WSN

The applications of WSN can be categorised into military, environment, home, agriculture and commercial areas [2]. Examples of WSN implementation for military purposes include monitoring, tracking and surveillance of borders; in industry it can be used for factory instrumentation; in a large city it can monitor traffic density and road conditions; in engineering it can monitor building structures; in the environment it can monitor forests, oceans, precision agriculture etc. The next section will explain some examples of the implementation of WSN in real

applications.

2.3.1 Forest Fires Applications

Forest fires present a challenge for forest management because they have the potential to be at once harmful and beneficial. On the one hand, forest fires can destroy vast amounts of timber resources and threaten communities, resulting in costly losses. On the other hand, forest fires are a natural part of the forest ecosystem and are important for maintaining the diversity and health of the forest. In Canada, the forest fire season runs from April through to October. The majority of fires covering the largest areas happen in June, July, and August. During a typical year there are over 9,000 forest fires, burning an average of 25 million hectares [ha] or 25 000 km^2 [19]. The number of fires and the area burned can vary dramatically from year to year. Table 2.2 shows the total number of forest fires and the size of the area burned in 2013. The weekly forest fire occurrence for the same year is illustrated in Figure 2.7. Two-thirds of all forest fires are caused by people, while lightning causes the remaining third. Yet, the lightning fires account for over 85% of the area burned in Canada [19]. This is because the lightning-caused fires usually occur at remote areas which are difficult to reach with fire suppression equipment. However, human-caused fires usually start close to communities, where they are reported quickly and are dealt with by local fire crews.

Table 2.2: Statistics of Forest Fires for the Canadian Forest Fires in 2013 [19]

Metric	2013	10 years avg	% Normal
Number	5,780	6,113	88%
Area (ha)	3,647,589	1,875,617	185%

Traditionally, look-out towers located at high points were used to detect forest fires. A person looks for fires using special devices, such as the Forest Fire Modelling Osborne fire finder [32], to determine the location of the forest fire. However, this approach has a few drawbacks due to the unreliability of human observation and the threat to life for forest fire personnel. Because of this, automatic video surveillance systems [21, 51] have been developed and introduced as part of forest fire detection systems. Automatic surveillance systems mainly use cameras and infrared (IR) detectors installed on top of towers. These systems are not suitable for monitoring large areas of forest. For this reason, aeroplanes or Unmanned Aerial Vehicles (UAV) are used for monitoring large forests [107, 12]. In addition, more advanced forest fire detection systems are based on satellite remote sensing.

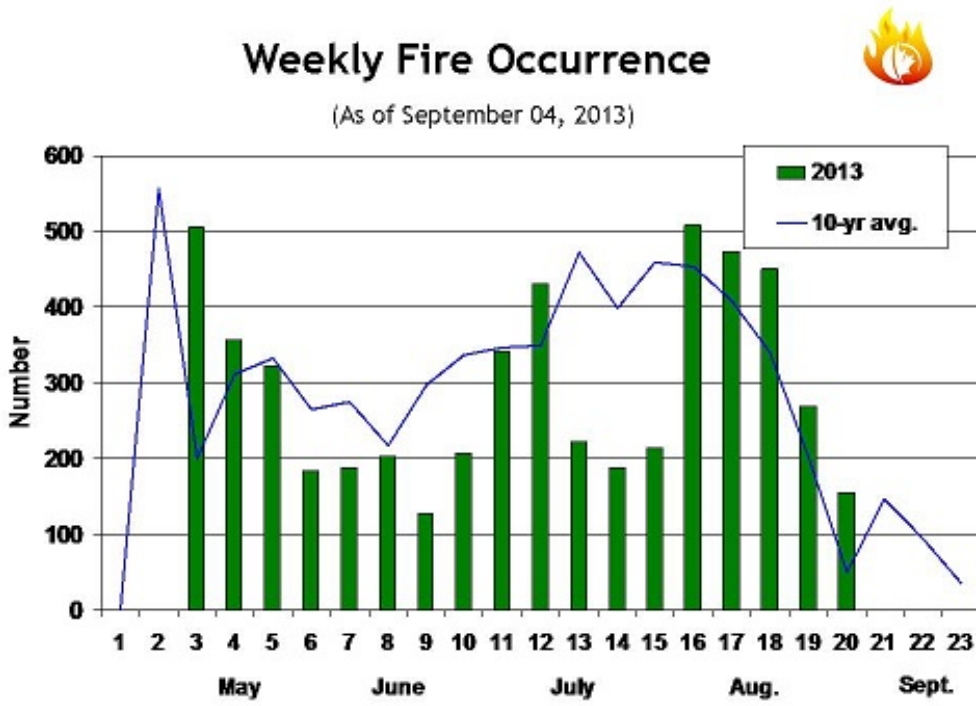


Figure 2.7: Weekly number of Forest Fires in 2013 [19]

These existing forest fire detection systems cannot function efficiently during all types of weather conditions. Their success depends on the time of the day, the existence of a line of sight and other visibility constraints [43]. WSNs can potentially provide solutions to these problems. If the system can be supported by WSNs, it can make a promising framework for a forest fire detection system. The current sensing modules of WSNs can sense a variety of phenomena including smoke, temperature and relative humidity which is helpful in forest fire detection systems. When no fire occurs, the sensor nodes send data in an infrequent manner such as one packet per day, in order to save their energy. If fire is detected, a sensor node must send data rapidly i.e. within a minute or even a second to monitor the unpredictable behaviour of the fire [5]. Moreover, the self-organising feature of sensor nodes to create a network means that the WSN can be easily deployed in a forest.

The most important goals in forest fire surveillance are: the quick and reliable detection and the accurate localisation of the fire [22]. Most forest fires are caught in the early stages before they have chance to grow. In Canada, only 3% of all the forest fires that start each year grow to more than 200 ha in area. However, these fires account for 97% of the total area burned across the country [19]. A WSN can provide timely detection of the forest fire because it is located in close proximity to the phenomenon. Furthermore, a WSN can provide information about the location

of the fire, fire growth, and the environmental conditions which are needed by the fire-fighting management [85]. By having this information, fire-fighting staff can be guided to the critical area to suppress it quickly by utilising the necessary fire-fighting equipment. As a result, there will be a reduction in the number of forest fires that become uncontrolled.

There has been a considerable amount of work carried out to deploy WSNs in forest fire detection systems. Researchers from Civil and Environmental Engineering, University of California [23], have designed a system for wildfire monitoring using WSN and have evaluated the system in a real experiment during prescribed test burns near San Francisco, California. In the experiment, the system was established using 10 sensor nodes with GPS capability which collected temperature, humidity and barometric pressure data. The collected data was sent to a base station to be stored on a database server, which could be accessed using a browser-based web application or any other application capable of communicating with the database server. The experiment showed that most of the motes in the burned area were capable of reporting information about the fire event, fire front spreading, increasing temperature, decreasing barometric pressure and decreasing humidity during the fire growth before they became burnt.

FireWxNet [35], a multi-tiered portable wireless system, is introduced for monitoring forest fire environments. The main objective of the system is to support the fire fighting community in safely viewing the fire and measuring the weather conditions to determine the behaviour of the fire rather than its detection. The system uses a tiered structure which consists of directional radios to provide long distance communication, a sensor network to provide environmental data, and web-enabled surveillance cameras to provide visual data. Data gathered from the sensor nodes and the web-enabled cameras are aggregated at a base station which has the capability of providing long distance communication using satellite technology. The sensor network measures temperature, wind speed, wind direction and relative humidity periodically every half an hour. However, the cameras monitor the fire zones continuously.

Another forest fire surveillance system based on WSN has been developed to monitor the South Korea Mountains. This system is called Forest-Fires Surveillance System (FFSS) [86]. The developed FFSS consists of WSNs, middleware and a web application. The sensor nodes are attached with the temperature, humidity and smoke sensors. The WSN uses a flat routing protocol based on the minimum cost path forwarding method to deliver data to the sink. The middleware program and the web application analyse the collected data and produce the fire risk level. The FFSS is able to provide a real-time alarm when a forest fire occurs.

2.3.2 Military and Safety Applications

WSN can be used in the military for a number of purposes such as monitoring or tracking enemies and force protection [55]. Unlike the commercial WSN, a tactical military sensor network has different priority requirements for military usage. Especially in the remote large-scale network, topology, self-configuration, network connectivity, maintenance, and energy consumption are the challenges [16].

An example of WSN deployment for military application is the Sensor Information Technology (SensIT) program [52]. This program was sponsored by the Information Technology Office (ITO) and was conducted by the Defence Advanced Research Projects Agency (DARPA). The primary goal of the SensIT program is to develop new software for distributed micro-sensors. The SensIT team pursued two key research and development thrusts. The first thrust is the development of new networking techniques. In the battlefield context, sensor devices or nodes should be ready for rapid deployment, in an ad-hoc fashion, and in highly dynamic environments. The second thrust is networked information processing, for example how to extract useful, reliable, and timely information from the deployed sensor network. This implies leveraging the distributed computing environment created by these sensors for signal and information processing in the network, and for dynamic and interactive querying and tasking the sensor network. The SensIT software was tested in the field, with the assistance of the United States Marine Corps and other service entities.

The Remotely Monitored Battlefield Sensor System (REMBASS) [68] is another implementation of a WSN in a military application. REMBASS is a ground-based battlefield surveillance system designed to detect, locate, classify, and report personnel and vehicular activities in real-time within the area of deployment. It uses remotely monitored sensors placed in position along likely enemy avenues of approach. These sensors respond to infrared energy, magnetic field changes and seismic-acoustic energy to detect enemy activities. The collected data is processed by the sensor nodes to provide detection and classification information, which is sent to the system monitoring set (SMS) wirelessly using radio communication. The messages are demodulated, decoded, displayed, and recorded to provide a time-phased record of enemy activity. REMBASS can be used to complement other manned/unmanned surveillance systems such as ground surveillance radar, unmanned aerial vehicles and night observation devices.

Smart Dust is envisioned to combine sensing, computing, and wireless communication capabilities in an autonomous, dust-grain-sized device [45]. A dense network of Smart Dust should then be able to unobtrusively monitor real-world processes with unprecedented quality and scale. The researchers in [80] had presen-

ted and evaluated a prototype implementation of a tracking system to find the exact location of real-world phenomena (using a toy car as an example) with Smart Dust. The toy car is equipped with an omnidirectional infra-red (IR) light emitter consisting of eight IR LEDs, which is mounted on top of the car. Accordingly, the sensor nodes are equipped with an omnidirectional IR light detector consisting of three IR photo diodes. The system includes techniques for node localisation, time synchronisation, and for message ordering specifically tailored for large networks of tiny Smart Dust devices.

2.3.3 Environmental Applications

Some environmental applications of sensor networks include: monitoring the environmental conditions that affect crops and livestock; tracking the movements of birds, small animals, and insects; biological, earth, and environmental monitoring in marine, soil, and atmospheric contexts; chemical/biological detection; irrigation; macro-instruments for large-scale earth monitoring and planetary exploration; precision agriculture; meteorological or geophysical research; bio-complexity mapping of the environment; forest fire detection; flood detection; and pollution study [2].

Habitat monitoring: A WSN architecture was proposed for habitat monitoring on Great Duck Island (GDI), a small island located 15 km south of Mount Desert Island, Maine [59]. The WSN architecture was designed for monitoring the nesting environments and behaviours of Leachs Storm Petrel. In the actual deployment, a WSN was established consisting of 32 Mica nodes. The network monitored underground nesting burrows and surface micro-climates for biologists and ecologists. The data, consisting of temperature, humidity, occupancy, and pressure, was used to correlate nesting patterns with micro-climates. Live data from the sensors can be viewed on the web.

Landslide Detection: The Calita [81] is a WSN infrastructure for landslide monitoring. In May 2009, the infrastructure was deployed in the Emilia Romagna Apennines. The deployment exploited 13 Crossbow Micaz motes with TinyOS software and covered a surface of about 500 m^2. Nodes are embedded with accelerometer sensor boards for capturing slope movements, and environmental boards for the monitoring of ambient parameters like temperature, pressure, humidity and light depth. The sensor nodes send the collected data to the base station, which is cable-connected to a laptop. The laptop sends the data by exploiting File Transfer Protocol (FTP) over a Universal Mobile Telecommunications System (UMTS) connection to the server; the data can be viewed via a Web graphical user interface. Another landslide detection system using WSN has been designed

and deployed at the Anthoniar Colony, Munnar, Idukki (District), Kerala (State), India [78]. The system provides information about the soil condition using selected geophysical sensors: pore pressure transducers, soil moisture sensors, geophones, stain gauges and tilt-meters. The geological sensors were placed inside a sensor column and connected to the wireless sensor node via a data acquisition board. The pilot deployment consists of two sensor columns with ten sensors, deployed in the field along with six wireless sensor nodes.

Flood monitoring: Flooding is a disaster that can cause loss of life, and damage to buildings and other structures including bridges, sewerage systems, roadways, and canals. The cost of damage caused by flooding is dependent on the warning time given before a flood event, making flood monitoring and prediction critical to minimising the cost of flood damage. The GridStix system [39] is an example of the implementation of a WSN in a flood monitoring system. The GridStix sensor platform uses powerful embedded hardware, heterogeneous wireless networking technologies and next generation grid middleware to implement an adaptable WSN. This allows nodes not only to send data to remote fixed grids but also to perform local grid computations to improve the support for flood monitoring and provide more timely warnings to local stakeholders in a range of formats including on-site audio/visual warnings, a public Web site and SMS alerts. For evaluating the system under real world conditions, fifteen GridStix nodes based on the Gumstix embedded computing platform are deployed to perform flood monitoring on a 3 km stretch of the River Ribble in the Yorkshire Dales, UK, which is prone to flooding for much of the year. All of these nodes are equipped with pressure-based depth sensors and a subset is equipped with ultrasound-based flow measurement equipment and cameras for image-based flow measurement. In addition, each node is equipped with 802.11bg1 and Bluetooth network hardware, which are used to provide an ad-hoc communications infrastructure. A single node is equipped with a GPRS uplink and a DVB satellite down-link. Each node is attached with a power solar array to address the energy issue caused by power-consuming devices, which continually powers a GridStix system even during the dark, British winter months.

Active volcano monitoring: A research group from Harvard University began collaborating with volcanologists at the University of North Carolina, the University of New Hampshire and the Instituto Geofísico n Ecuador in order to study active volcanoes. This group was amongst the first to investigate the use of a WSN for the purpose of geophysical studies. In 2004, they deployed a small WSN on Volcán Tungurahua in central Ecuador as a proof of concept using three nodes equipped with microphones collecting continuous data from the erupting volcano for three days [98]. August 2005, they deployed a larger and more capable net-

work on Volcán Reventador in northern Ecuador [99]. The network consisted of 16 sensor nodes, with each of the sensor nodes equipped with a microphone and seismometer, collecting seismic and acoustic data on volcanic activity. The nodes relayed data via a multi-hop network to a gateway node connected to a long-distance FreeWave modem, providing radio connectivity with a laptop at the observatory. A GPS receiver was used along with a multi-hop time-synchronisation protocol to establish a network-wide timebase. Over the three weeks of the deployment, the network captured 230 volcanic events.

2.3.4 Agriculture Monitoring Systems

The Precision Agriculture Monitor System (PAMS) is an intelligent system which can monitor the agricultural environment of crops and provide services to farmers [57]. The system has been deployed successfully in many places in the Shaanxi province of China such as Yangling, Ansai Apple Park, etc. The main objective of the system was to improve crop output by managing and monitoring the crops' growth period. Examples of monitored environment factors include: air temperature and humidity, soil temperature and moisture, carbon dioxide concentration and illumination intensity. Information about these environmental factors is sent to the control unit to be analysed. The system sends an alarm message to the user when it has detected an abnormal situation affecting the crops growth. Besides this, the system can prevent water wastage.

Another example of a WSN implementation in precision agriculture is the LOFAR-agro project [54]. This project is the first large-scale experiment in precision agriculture in the Netherlands. The project concerns protection of a potato crop against phytophthora, a fungal disease that can spread easily amongst plants and destroy a complete harvest within a large region. The development of the fungus and its associated attack on the crop depends strongly on the climatological conditions within the field. In particular, the humidity and temperature within the crop canopy are important factors in the development of the disease. To monitor these critical factors, a potato field was instrumented with a WSN. A close monitoring of the micro-climate can reveal when the crop is at risk of developing phytophthora and allows the farmer to treat the field, or parts of it, with fungicide only when absolutely needed. This precise treatment saves time, reduces costs, and limits the use of environmentally unfriendly substances as opposed to traditional treatment based on information from a remote weather station.

2.3.5 Commercial Building Applications

WSN could be used in commercial building applications. One example of these applications is a structural health monitoring system designed to seek, detect, and localise damage within buildings, bridges, ships and aircraft. Structural health monitoring is the collection and analysis of the structural response to ambient or forced excitation. Wisden [101] is one example of the implementation of a WSN in structural health monitoring. Wisden collects structural response data from a multihop network of sensor nodes, and displays and stores the data at a base station. In terms of hardware, Wisden used Mica-2 motes with a 16-bit vibration card designed specifically for high-quality vibration sensing. The researchers deployed 10 nodes of Wisden on a test structure resembling the frame of a hospital ceiling. The structure was repeatedly hit with a 2-by-411 for 20 s. The system collected data and displayed it at the base station successfully. Another deployment has also been carried out to further evaluate Wisden system performance. In this deployment, a 14 MicaZ node WSN was deployed on a large seismic test structure used by civil engineers [73]. The experiment indicates that Wisden can deliver time-synchronised structural vibration data reliably across multiple hops with low latencies.

A group of researchers from the University of California has successfully implemented WSN for structural health monitoring on a bridge [74]. They have designed, implemented, deployed and tested a WSN on the 1280 m (4200 feet) long main span and the south tower of the Golden Gate Bridge (GGB). Ambient structural vibrations are reliably measured at a low cost and without interfering with the operation of the bridge. In the GGB WSN deployment, 64 nodes are distributed over the main span and the tower, collecting ambient vibrations synchronously at 1kHz rate. The sampled data is collected reliably over the 46-hop network.

2.4 Constraints in WSN

A WSN has many constraints compared to a traditional computer network. Due to these constraints which are discussed below, the deployment of a WSN needs to be managed carefully to meet the desired performance which depends on the requirements of the applications. The constraints are classified into two categories: sensor node constraints and networking constraints [11].

2.4.1 Sensor Node constraints

One major constraint in a WSN is that the network has very limited resources. A sensor node has two obvious resources limitations, which are its energy sources and memory storage. These resources must be managed efficiently in order for the WSN to operate properly within an intended period of deployment.

Energy Limitation: Energy is a main constraint to a WSN's capabilities. Sensor nodes depend on batteries (normally size AA) as their energy source. Once sensor nodes are deployed, their battery cannot be replaced or recharged easily especially due to the nature of the deployment area, such as it being a remote area in habitat monitoring, a hazardous area in chemical plant monitoring, a dangerous area in disaster monitoring etc. Therefore, the battery charge taken with them to the deployment area must be conserved to extend the life of the individual sensor node and at the same time the entire sensor network.

Limited Memory Storage: A sensor is a tiny device with only a small amount of memory and storage spaces for program code. It is very important to limit the code size of the entire program to fit the memory spaces available, which is different depending on type of the sensor node. For example, a MICAZ sensor node has an 8-bit, 7.37 MHz CPU based on Atmel ATmega128L with only 4 kB EEPROM and 128 kB program flash memory. A Sensinode uses 8051 MCU, which is an 8-Bit MCU with 128 kB flash memory and 8 kB RAM. With such a limitation, the software built for the sensor must also be quite small.

2.4.2 Networking Constraints

A WSN operates in a lossy environment. This lossy environment contributes to the unreliable communication among the sensor nodes which limits the networking capabilities of the network. This unreliable communication is influenced by many factors which include [96]:

Unreliable Transfer: Normally, packet-based routing in WSN is connection-less and thus inherently unreliable. A sensor node can send a packet to another sensor node in the network without prior arrangement to establish a dedicated end-to-end connection. This means that the sender node simply transmits the packet without ensuring that the destination node is ready to receive the packet. Furthermore, the sender node does not know whether the packet has reached the destination node or not. The packet may get damaged due to channel errors and interferences, or dropped at highly congested nodes. It is very important for the WSN to have appropriate error handling to reduce the probability of losing packets.

Packet collision: Even if the channel is reliable, the communication itself may still be unreliable. This is due to the broadcast nature of the WSN which causes

a high probability of packet collisions happening. Packet collision occurs when a sensor node makes an attempt to transmit a packet but at the same time is receiving a packet from another sensor node, which can cause the packet transfer to the destination node to fail. This condition becomes worse in a high density WSN and can thus be a major problem.

Latency: The WSN uses a multi-hop communication technique to overcome the short transmission range due to the low transmit power used by a sensor node to save its energy source. However, latency could become a major issue in this multi-hop communication to deliver a packet from a source node to a destination node. This communication technique may introduce delay within the network as packets traverse hop-by-hop in the network. Network congestion and node processing can add greater delay into the packet delivery.

2.5 Routing Challenges in WSN

The design of routing protocols in WSN is influenced by many challenging factors. These factors must be considered in the development of routing protocols before efficient data delivery can be achieved in a WSN. Routing challenges and design issues that affect the development of a routing protocol in a WSN are explained below [4]:

Node deployment: Two methods of node deployment which can be used are deterministic (manual) or randomised. The selection of node deployment in a WSN is application dependent. In deterministic deployment, sensors nodes are manually placed at specific, predetermined locations in a deployment area. However, in random node deployment, sensor nodes are scattered randomly. An example of an application that is suitable to be used in a deterministic approach is a temperature monitoring system for a greenhouse. Sensor nodes need to be located at specific places in the greenhouse in order to obtain accurate measurements. In the military, however, nodes are deployed randomly from an aeroplane to collect information about a battlefield.

Energy consumption: A sensor node has a limited amount of energy because it is just powered by batteries. In order for a WSN to have a longer lifetime, the energy source must be conserved. Two main energy consumptions are communication (transmitting/receiving) and node processing activity. Every sensor node must cooperate with the others in order to provide efficient communication. The malfunctioning of some sensor nodes due to power failure can cause significant topological changes, and might require the rerouting of packets and the reorganisation of the network. Normally, the sensor nodes in a WSN are in sleep mode most of the time to save energy due to idle listening.

Data reporting method: Data reporting can be categorised as either time-driven, event-driven, query-driven, or hybrid (a combination of methods). The time-driven delivery method is suitable for periodic data monitoring applications. All sensor nodes sense the environmental phenomenon and send the sensor reading periodically at constant time intervals. In the event-driven method, a sensor node sends data immediately if an event is detected due to changes in the value of a sensed attribute. This method is very suitable for time-critical applications for alarm/detection monitoring systems. In query-driven methods, sensor nodes react or respond to a query generated by a base station or other nodes in the network. Sensor nodes only send a report to the base station if an event meets with the query.

Network dynamic: In many research studies, sensor nodes are assumed to have a fixed location in the network. However, in certain cases both the base station and the sensor nodes are moving. Routing protocols must be able to support this kind of situation in which the network topology is changing. Here, stability becomes the important design factor in any routing protocol on top of energy, bandwidth, etc. An example of an application related to this mobility situation is a target/tracking application.

Scalability: In a WSN, the number of sensor nodes being deployed in a sensing area varies and may be in the order of tens, hundreds or thousands (small, medium and large-scaled network accordingly). Routing protocols must be scalable enough to cater for deployments with any of these different numbers of nodes. Most of the existing routing protocols are tested in small-scaled networks consisting of less than 100 sensor nodes. The performances of the routing protocols in a large-scaled network still need to be determined.

Fault tolerance: Some of the sensor nodes may fail due to energy depletion, broken circuitry, or becoming damaged by sensed environments. These failures of the sensor nodes should not affect the operation of the network. If any node fails, the routing protocols must accommodate the formation of alternative routes for packets to the base station through the available nodes. Therefore, multiple levels of redundancy are needed in a fault-tolerant sensor network. Fault-tolerant routing protocols are needed in a condition where the sensor nodes are being threatened with failure such as in disaster situations: forest fires, volcanic eruptions and flooding.

Quality of Services: a WSN has different quality of service requirements for different applications. In multimedia applications such as audio and video, data must be delivered within a certain period of time from the moment it is sensed, or it will be useless. In critical applications, routing protocols must be able to ensure that critical packets are delivered to the base station within a short period

of time successfully. In normal monitoring applications, data may be aggregated before being delivered to the base station and the delay in delivery is not really important as long as the data reaches the base station. However, conservation of energy, which is directly related to network lifetime, is considered relatively more important than quality of service in many applications. As sensor node energy is depleted, the network may require to reduce energy consumption in the network to extend the network lifetime. Hence, energy-aware routing protocols are required to meet this quality of service requirement.

2.6 Routing Protocols in WSN

A WSN is application-specific whereby the network being deployed has specific requirements, which is different for different applications. This means that routing protocols must be designed to meet the application requirements. A few years after the introduction of WSN, this technology is used in normal and simple monitoring applications. Examples of applications are temperature sensing in a room, the humidity sensing of soil and the sensing of light intensity in a greenhouse. Sensor nodes need to sense the environment and periodically send the reading to the base station. These systems are required to operate over a long duration of time but are just powered by batteries. This means that energy saving is one of the main requirements for a routing protocol to meet in order to prolong the lifetime of the networks. The usage of a WSN is extended to support more complex data in multimedia applications (video and audio data transfer). Routing protocols in a WSN should provide low delay data transfer in order to achieve high data quality, otherwise the data will be useless. A few research studies have managed to extend a WSN implementation for monitoring disaster situations such as forest fires, volcanic eruptions, flooding, and earthquakes, etc. In these critical applications, routing protocols must be able to provide reliable data transfer in such unpredictable and damaging environments during the time of the disaster. Routing protocols should be able to deliver every critical packet to its destination within an acceptable delay and using less energy. This clearly shows that the advancement of routing protocols follows the trends of application need. The next section will explain a few examples of routing protocols in each type of application.

2.6.1 Routing for Normal Applications

In normal monitoring applications, energy saving is a key requirement in the routing protocol's design. A WSN is required to operate over long durations of time. Routing protocols use multi-hop communication in order to avoid the energy

loss in transmitting the data directly to the base station. In direct communication, the sensor nodes that are located further away from the base station need to transmit with higher power and have their energy drained much faster than the sensor nodes located closer to the base station. Examples of popular energy saving routing protocols for a WSN are LEACH, PEGASIS, SPIN and GEAR. A brief description of each of the routing protocols is given below:

Low Energy Adaptive Clustering Hierarchy (LEACH) [38]: LEACH is a cluster-based protocol, which includes distributed cluster formation. LEACH randomly selects a few sensor nodes as cluster heads (CHs) and rotates this role to evenly distribute the energy load among the sensor nodes in the network. In LEACH, the CH nodes compress the data arriving from the sensor nodes that belong to the respective cluster, and send an aggregated packet to reduce the amount of information that must be transmitted to the base station. LEACH uses a TDMA/CDMA MAC to reduce inter-cluster and intra-cluster collisions.

Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [58]: PEGASIS is a near optimal chain-based protocol. The basic idea of PEGASIS is that sensor nodes only need to communicate with their closest neighbour and take turns in communicating with the base station in order to extend the network lifetime. When a round of turns for all nodes communicating with the base station ends, a new round starts, and so on. This reduces the power required to transmit data per round as the power draining is spread uniformly over all the nodes. Hence, PEGASIS has two main objectives: the first is to increase the lifetime of each node by using collaborative techniques; and the second is to allow only local coordination between nodes that are close together so that the bandwidth consumed in communication is reduced.

The Sensor Protocols for Information via Negotiation (SPIN) [37]: This routing protocol is a family of negotiation-based information dissemination protocols suitable for WSNs. The routing protocol focuses on the efficient dissemination of individual sensor node observations to all sensor nodes in the network, assuming that all sensor nodes are potential base stations. This enables a user to query any node to get the required information about the network. The routing protocol is designed to overcome the problems of implosion and overlap which cause wastage of the energy source in classic flooding. In order to overcome these deficiencies, SPIN uses a data negotiation and resource adaptation algorithm. Sensor nodes have to broadcast an advertisement to their neighbours and send the actual data only to those nodes that are interested. To reduce the energy expended in the broadcast of advertisements, the SPIN protocol family uses meta-data descriptors, which describe the actual sensor data in a more compact size. In addition, SPIN adapts the routing decision based on the remaining energy of each node in the

network.

Directed diffusion [42]: Directed diffusion is a data-centric (DC) and application-aware paradigm, in the sense that all the data generated by the sensor nodes is named by attribute value pairs. A sensing task is disseminated throughout the sensor network as an interest for the named data, which is originated by the sink node. The main idea of the DC paradigm is to combine the data coming from different source routes (in network aggregation) by eliminating redundancy and minimising the number of transmissions, thus saving network energy and prolonging network lifetime.

Geographic and Energy Aware Routing (GEAR) [103]: This routing protocol makes use of geographic information while disseminating queries to appropriate regions since data queries often include geographic attributes. GEAR uses energy-aware and geographically informed neighbour selection heuristics to route a packet toward the destination region. The key idea is to restrict the number of interests in directed diffusion by only considering a certain region rather than sending the interests to the whole network.

2.6.2 Routing for Multimedia Applications

WSN enhancement makes it possible for multimedia data, such as video stream, audio stream, and still images from the environment, to be transferred over the networks. This capability not only enhances existing sensor network applications, but it also enables several new applications such as multimedia surveillance systems, storage of potentially relevant activities, traffic avoidance, enforcement and control systems, and advanced health care delivery [3]. In order for routing protocols to support multimedia data, the routing protocols must meet the main applications requirement which is low end-to-end delay. Examples of routing protocols for multimedia applications are described below:

SPEED [36]: This routing protocol is a stateless protocol for real-time communication in a WSN. It bounds the end-to-end communication delay by maintaining a desired delivery speed across the network through a combination of feedback control and non-deterministic QoS aware geographic forwarding. The protocol provides three types of real-time communication services: real-time unicast, real-time area-multicast and real-time area-anycast. SPEED is a highly efficient and scalable protocol for sensor networks where the resources of each node are limited.

Multipath Multi-SPEED Protocol (MMSPEED) [31]: MMSPEED is an extension over the SPEED protocol. The routing protocol is based on a cross-layer approach between the network and the MAC layers. It was designed to differentiate efficiently between flows with different timeliness and reliability requirements.

Multiple QoS levels in the timeliness domain are provided by multiple packet delivery speed options. In the reliability domain, various reliability requirements are supported by probabilistic multipath forwarding. This means that packets can choose the most effective combination of service options depending on their timeliness and reliability requirements.

Secure Real-time with Load Distribution (SRTLD) [1]: This is a routing protocol that provides secure, real time data transfer and efficient distributed energy usage in a WSN. The SRTLD routing protocol ensures high packet throughput with minimised packet overhead and prolongs the lifetime of a WSN. The routing depends on an optimal forwarding decision which takes into account link quality, packet delay and the remaining energy of the sensor node. The SRTLD routing protocol possesses built-in and enhanced security for packet transfer. Encryption and decryption with authentication of the packet header further supplement secure packet transfer.

Adaptive Real-time Routing scheme (ARP) [76]: Real-time data transmission and dynamically adapts to the different real-time requirements of the applications. The routing scheme differentiates its service type by the deadline field of packets and their destinations. Based on this information, ARP calculates the required transmission velocity and assigns a priority to the data packets. Packets with a higher priority receive the services earlier than low priority packets. Although ARP provides priority in the data transmission at nodes, it cannot guarantee end-to-end real-time packet delivery to destinations.

2.6.3 Routing for Critical Applications

There are three fundamental requirements associated with routing protocols in a WSN; they need to be energy efficient, have reliable data delivery, and a low latency. In normal applications, energy efficiency becomes the important requirement in order for the network to operate over a long duration of time. In multimedia applications, low end-to-end delay becomes the main requirement to ensure that high quality data is being received at the base station. In critical applications, reliable data delivery and low delay become more important requirements instead of energy efficiency. However, it is not a good idea to totally ignore the energy consumption in the routing protocol's design because it can cause the network to die much faster, possibly before or during the disaster. This is not acceptable in critical applications which require the networks to be continuously available to provide information about the disaster situation.

Routing protocols in critical applications can be divided into three categories based on the time of deployment: before, during and after the critical event.

Deployment	Applications	Routing Requirement
Before	Disaster detection system	Energy is the main requirement because the time the critical event will start to occur is unknown. Overall remaining energy source at the time the critical event detected is uncertain. Energy efficient and load balancing are suitable routing techniques in this case.
During	Disaster monitoring system	Reliable, low delay transfer and energy is important to ensure that critical packets can be delivered to the sink. Routing protocols must adapt with dynamic changes in the disaster. In certain cases the network is being threatened by the environment.
After	Rescuing support system	Reliable and low delay transfer are very important parameters. Critical information (such as the location of an injured person) must be delivered to the sink as soon as possible. It is a very crucial time; a minute late could cause loss of human life.

Table 2.3: Routing protocol requirements for different types of WSN deployment in critical applications.

The selection time of deployment is dependent on the system that needs to be established. Deployment before a critical event is for detection monitoring systems, deployment during a critical event is for the monitoring of the event, and deployment after a critical event is for the rescuing support system. Each type of deployment has different requirements for the routing design, as described in Table 2.3. Examples of routing protocols in critical applications are explained below:

TEEN and APTEEN: the earliest routing protocols designed specifically for time-critical applications are two hierarchical routing protocols called Threshold-Sensitive Energy Efficient Sensor Network Protocol (TEEN) [60] and Adaptive Periodic TEEN (APTEEN) [61]. In TEEN, sensor nodes sense the environment continuously, but sensed data is transmitted less frequently. A CH sends its members a hard threshold and a soft threshold. The hard threshold is the threshold value of the sensed attribute, and the soft threshold is a small change in the value of the sensed attribute which triggers the sensor node to become active to send an alert to CH. Thus, the hard threshold reduces the number of transmissions by allowing the sensor node to send data only when a sensed attribute is in the range of interest. The soft threshold further reduces the number of transmissions that might otherwise occur when there is little or no change in the sensed attribute. This means that a more accurate information of the network can be achieved

with a smaller value of the soft threshold, at the expense of increased energy consumption. The main disadvantage of this scheme is that if the thresholds are not received, the nodes will not send any data to the CH. APTEEN, on the other hand, is a hybrid protocol that changes the threshold or periodicity values used in the TEEN protocol according to user requirements and the types of application.

Periodic, Event-Driven and Query-Based Protocol (PEQ) [10]: PEQ is a routing protocol that meets a WSN's requirements for critical surveillance applications. PEQ uses the publish/subscribe paradigm to disseminate requests across the network and an ack-based scheme to provide fault tolerance. In the publish/subscribe interaction paradigm, one or more sinks receive events notification from a sensor in the network. Initially, the sink expresses an interest in a sensor by subscribing to certain information that it requires from the sensor. When the sensor detects that information, it publishes it by firing an event, which is sent to the sinks through a notification message in an asynchronous way. PEQ can provide fast broken path reconfiguration, low latency for event notification, and high reliability and low energy dissipation in the delivery of the events.

Hierarchical Periodic, Event-driven and Query-based (HPEQ) [8]: HPEQ is a cluster-based routing protocol that groups sensor nodes to efficiently relay all sensed data to the sink. Sensor nodes with more residual energy are selected as aggregators that relay data to the sink in order to uniformly distributing energy dissipation among the nodes, and reducing latency and network data traffic. HPEQ uses the publish/subscribe paradigm to disseminate requests across the network based on PEQ.

Environmental Monitoring Aware Routing (EMA) [97]: A "context-aware" routing protocol that adapts its routing table based on the failure threat due to the sensed environments. EMA proactively avoids route breakages due to node failure caused by the environment and at the same time attempts to be power efficient. EMA can provide robust and reliable data transfer from sources to the destination. In order to select the best neighbour to forward data to, a metric is used as the routing decision criteria. Nodes health status, RSSI and hop count are parameters used to calculate the metric. Health status is defined to be a value between 0 and 100, with 0 being the worst and 100 the best health. Health status is linearly dependent on the temperature. If the temperature is below a detection threshold, the health status is 100. Otherwise, if the temperature is above the detection threshold, the health status is 0, indicating that the node is likely to fail at any time. EMA is able to provide a robust and reliable routing protocol in a forest fire scenario.

Delay-bounded and Robust Routing Protocol (DRR) [105]: This is a routing protocol which is suitable for use in emergency environments. DRR provides delay-

bounded delivery and robust routing to ensure safe and fast response in emergency situations. DRR is able to adapt in dynamic emergency situations where nodes are being threatened with failure by hazards, and works well with the holes problem in the network. This is achieved by considering the health status of the nodes, which depends on the environmental conditions, in the routing decision. DRR introduces four levels of node statuses: safe, low safe, dangerous, and fail depending on the environmental conditions. In order to provide robust data transfer, routes are established along safe nodes that have higher chances (compared to nodes with other statuses) to be selected as forwarding nodes. DRR solves the holes problem by increasing the node transmission power gradually to find another neighbour during new neighbour discovery. If a node fails to find any neighbours toward the base station, the node sends a notification message to the neighbour nodes to stop sending packets through it. Then a routing re-discovery is invoked by the node.

Pattern Based Routing (PBR) [93]: This is a pattern based routing protocol that activates nodes when patterns are detected in order to avoid unwanted message relay. If a node senses a pattern, the node starts sending data through the active nodes to the actor node (or the cluster header), which is located a few hops away. Sending essential information in the form of patterns reduces the number of collisions and improves latency. Besides this, PBR can reduce the message size and improve the packet delivery ratio.

2.7 Routing Techniques in WSN

The routing techniques implemented in the existing routing protocols in order to achieve energy efficiency, reliability and low delay data transfer are explained below:

1. *Mobile collector* [75, 91, 89, 6]: This involves one or more mobile sink nodes moving around and sending broadcast messages periodically to collect data from nearby sensor nodes in the network. Sensor nodes that receive the message configure their routing table to join the mobile sinks. After this, sensor nodes start sending sensed information directly to mobile sink nodes. However, in the situation whereby the mobile sink node is not available, the sensed information is sent to the static sink using multi-hop communication. In an emergency situation, a mobile sink offers a temporary connection to some cluster nodes which become disconnected from the network because some nodes are being destroyed.

2. *Packet priority* [87, 67, 49]: This technique introduces a priority field in the packet to provide priority in transmission. In [87], the authors make

the assumption that four levels of priority will be sufficient to support real-time traffic: forced, urgent, reserved and normal data. For example, if two packets are available in the buffer, the packet with the higher priority will be transmitted first compared to the lower priority packet. In [67], the authors proposed a mission-oriented selective routing scheme that runs on each node in the network and invokes different algorithms for data of different priority, so as to determine a route which meets the objectives of each priority level in terms of path delay, energy consumption and reliability.

3. *Multi-path routing* [24, 62]: Reliability can be provided by enabling redundancy in data transmission, which can be achieved by constructing several paths from the source to the destination and traversing the same data packets through each of these paths. Enhanced multipath routing limits the routing to neighbours located towards the direction of the destination and ignores the others. The disadvantages of this technique are that it requires a higher energy consumption and creates network congestion. This technique is very suitable for small network sizes.

4. *Prediction of neighbour's condition* [62]: The main advantage of this technique is to increase the probability of successful data transfer to the sink. The technique uses prediction over contexts of the sensor nodes (such as previously encountered neighbours, failure neighbours, battery level, current sensor reading and etc.) to foresee which of the neighbours is the best node to forward the data to the sink.

5. *Load-balancing* [84]: Imbalanced loading causes the energy of some sensor nodes to deplete very quickly resulting in the short lifetime of the network. In order to solve this problem, the load-balancing technique can be used to distribute the load in the network evenly. Packets are only forwarded to neighbours which have a higher energy level toward the destination. Lower energy sensor nodes do not become part of the communication but are still able to continue monitoring the environment. Indirectly, the lifetime of the network can be extended for a much longer period.

6. *Multiple sinks* [9, 97]: A single sink represents a bottleneck in the network which can cause a major problem, especially during critical situations. The introduction of multiple sinks in the network is able to solve this problem by providing an alternative for data delivery which avoids overusing the nodes located close to the sink. At the same time, this technique can increase the networks fault tolerance and provide higher reliability. The availability of multiple sinks might also result in lower transfer delay in a large scale WSN.

7. *Repositioning sink* [102]: This technique uses an approach to dynamically relocate the sink closer to the sensor nodes involved in heavy traffic, in order to increase the energy efficiency, packet delivery ratio and timeliness. However, this technique has the disadvantage that a relocation may jeopardize the sink since it often moves the base-station too close to the danger zones. An efficient, safety-oriented relocation algorithm for the sink is needed to overcome this problem. As an example, STEER [102] strives to protect the sink while maintaining good network performance.

8. *Back-up cluster header* [34, 28]: This technique is suitable for use in a cluster-based network topology. During cluster formation and cluster header (CH) selection, a back-up cluster header (BH) is selected based on the node's capability (energy, link quality, functionality, etc) which is usually slightly lower than the CH. The BH takes over the role of cluster header when the CH has failed, perhaps due to energy depletion or being damaged by the environment. The main advantage of this technique is the reduction of the time needed to select the new cluster header. The cluster connectivity can be recovered more quickly in order to continue providing information to the sink.

9. *Acknowledgement* [79]: This technique ensures that the sender node knows whether each transmitted packet has been received or not by the destination node. For each one hop communication, the destination node must send a reply or acknowledgement when it receives a packet from the sender node. It is important for the sender to know that the data it sends is received by the destination node. If no acknowledgement is received, the sender retransmits the data only if the number of retransmissions does not exceed the maximum value. The drawback is that this incurs a significant increment in delivery delay proportional to the number of hops.

10. *Maximise safe paths [97, 105]*: This routing technique provides a high reliability data transfer by routing packets through the safe paths avoiding the critical regions. It is because a node failure due to the hazard in the critical regions can caused a broken path. In addition, this technique can reduce the time and energy consumption in route recovery. This technique requires a routing protocol to have the ability to adapt with the dynamic network during critical situations. EMA and DRR are two examples of routing protocols using this technique.

The situation is different in PBR (Pattern Based Routing) [93], where a routing path is established through the active nodes in the critical regions. Each node de-

termines whether the critical event has occurred or not based on the pattern of the sensed environmental data. If a critical pattern is detected, the node is activated to send the pattern data to the actor node and becomes part of the communication link. Otherwise, the node remains inactive to save energy. However, PBR does not consider node failure caused by hazards in the routing design.

WEAR [84] argues that concentrating only on an optimal path could cause the energy of the nodes along the path to deplete sooner compared to the other nodes in the network. This can lead to holes creation in the network because of node failures due to the exhausted power supply. The hole will keep enlarging and shorten the lifetime of the network. This is not acceptable because it is necessary for the network to continuously monitor the environment during critical situations. This problem happens because of the imbalanced load distribution amongst each of the sensor nodes in the network. By having an even load distribution amongst the sensor nodes, the lifetime of the network can be extended for a much longer period.

In the present research, MUP is proposed to investigate the idea of exploiting the energy of nodes that are going to become damaged soon due to the environment. This can save the energy of other nodes in the network, which is expected to prolong network lifetime during the critical phase.

2.8 Summary

This chapter has given an overview of WSN. WSNs are designed based on the IEEE 802.15.4 standard, which defines the physical and MAC sub-layer specifications for LR-WPAN. In addition, a WSN uses a 6LoWPAN adaptation layer allowing an IPv6 packet to be transmitted within the minimum datagram size of IEEE802.15.4 that makes it possible to connect a thousand sensor nodes to the Internet. A WSN has a few limitations and constraints, such as limited energy, small memory size, and slow speed of data processing. These introduces several challenging factors in designing a routing protocol in WSN. The existing routing protocols in WSNs are classified into three categories based on the applications: normal, critical and real time applications. The routing techniques used by the routing protocols to achieve their application requirements, in terms of network lifetime, fault tolerance and reliability, especially during a disaster or a critical situation, are explained in this chapter. Based on this background study, we have developed the idea of exploiting the energy of the unsafe nodes that are going to be destroyed soon in order to save the energy of the other nodes during a disaster situation. In order to study the idea, we have proposed the Maximise Unsafe Path routing protocol, which will be elaborated in detail in the following chapter.

Chapter 3

MUP System Design

3.1 Introduction

MUP is designed for normal monitoring applications. As an example, a possible application is temperature monitoring in a forest. Each node senses the surrounding temperature of the forest and sends the sensor reading to the sink periodically. In normal situations, the surrounding temperature changes gradually throughout the day. In order to avoid the waste of energy involved in sending a large number of redundant packets, which contain the same temperature values, each node should send data in an infrequent manner such as every hour or twice a day. During a forest fire situation, nodes must increase their sensing rate and send data more frequently, such as a packet every second. The reason for this is that the environmental temperature during a forest fire is unpredictable and can change significantly [5]. It is very important that MUP is able to adapt with the environment to achieve the routing requirements in both the normal and the disaster phase.

The main objective of MUP is to provide a longer network lifetime during the disaster phase. This allows the WSN to operate for as long as possible to detect and monitor the environment, which reduces the possibility of a disaster becoming uncontrollable. In order to meet the objective, MUP is designed to exploit the energy of the unsafe nodes before they are totally destroyed by the sensed environment in order to save the energy of the other nodes by concentrating packets through the unsafe nodes. This approach has a trade-off of poor network performance in terms of the packet delivery ratio and end-to-end delay due to congestion and radio interference.

This chapter explains the design concept of MUP, which consist of three functional modules: the critical event detection module, the neighbourhood management module and the routing management module. The MUP design concept is

implemented as extension to RPL. Because of this, an overview of RPL design is included in this chapter. Each modification that has been made in implementing the MUP design concept in RPL is described clearly.

3.2 MUP Design Concept

In MUP, the routing design concept consists of three functional modules: routing management, neighbourhood management and critical event detection. These functional modules belong to the control plane of the routing. Routing management is the main module that plays an important role in updating forwarding choices and making forwarding decisions. The neighbourhood management module discovers a subset of forwarding candidate nodes and maintains the routing table. The critical event detection module detects if any critical event occurs and sends alert to the routing management module. In addition, this module is needed to determine health status of a node. Connections among these functional modules are shown clearly in Figure 3.1

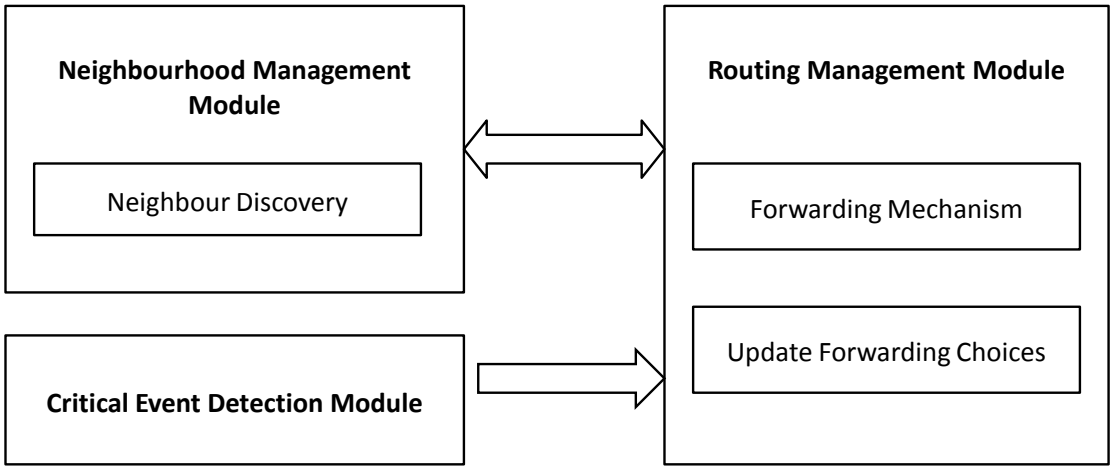


Figure 3.1: Functional Modules of MUP

The routing management module consists of two sub-modules: the forwarding mechanism and updating forwarding choices. The forwarding mechanism sub-module decides on a suitable neighbour for the efficient forwarding of packets, based on these routing parameters: the total path cost and the node health status. The selected neighbour will be used by the data plane of the routing as the forwarding nodes for a MUP node to forward any received packet that is not destined for itself. The update forwarding choices sub-module has two responsibilities: updating the neighbour metrics and triggering routing management messages to inform neighbours if any changes of health status are detected.

The neighbourhood management module plays an important role in discover-

ing neighbour nodes and maintaining the neighbour table. This module is very important during the initialisation stage. When a sensor node is started, the node invokes the neighbour discovery mechanism to start broadcasting routing management messages. Each neighbouring node that receives the message send a reply. Upon receiving a reply, the neighbourhood management module records the new neighbour in the neighbour table. Due to limited memory space, the neighbour table has a fixed length. If the neighbour table is full, the neighbourhood management module compares the metrics of the new neighbour with the metrics of the neighbours in the neighbour table. If the new neighbour has a lower path cost than any neighbour in the neighbour table, the neighbourhood management module will replace the highest path cost neighbour with the new neighbour information. Otherwise, the neighbourhood management module simply drops the new neighbour information.

The critical event detection module is very important in detecting disaster events and in determining node health statuses. If there is a change in the node health status, the module sends an update to the routing management module. After receiving the update, the routing management module triggers the sending of routing management messages to inform other neighbours about the node's current health status.

3.2.1 Node Health Status

MUP introduces a mechanism to determine the threat of the disaster event on every sensor in the network. Every sensor node faces different levels of threat, which can be measured based on the temperature sensor readings and the neighbour nodes' conditions. MUP defines four levels of node health status:

- SAFE: initial stage and while there is no disaster.
- LOWSAFE: one-hop away from a detected disaster event.
- UNSAFE: disaster event detected.
- ALMOST-FAILED: just about to be destroyed.

Two sensor reading thresholds are introduced to indicate different levels of environmental threat to a sensor node, which cause changes in the node health status: a starting sensor reading value when a node detects a disaster event (SAFE to UNSAFE) and a sensor reading value when a node is almost failed (UNSAFE to ALMOST-FAILED). If a node detects a sensor reading equal to or just exceeding one of the thresholds, the node changes its health status accordingly. In addition,

a node changes its health status from SAFE to LOWSAFE when the node has just received an update routing message from a neighbour with an ALMOST-FAILED health status.

3.2.2 Total Path Cost Calculation

The total path cost can be calculated using an additive metric, such as Expected Transmission Count (ETX), link quality level, latency and node energy. The sink node is set to have the minimum total path cost. As MUP only considers one-hop neighbours in the best parent selection, for a non-sink node, it needs to compute the total path cost to the sink through each neighbour node. If the link for the path through that neighbour is not available, the path cost through that neighbour should be set to maximum. This maximum total path cost value prevents this path from being considered for path selection.

Two methods of total path cost calculation are introduced in the routing design. In the first method, a MUP node considers the total path cost from a neighbour node to the sink. Otherwise, in the second method, a MUP node considers the total path cost from the node itself to the sink through a neighbour node. The difference between these methods is that either a link cost between a node and its neighbour node is included or not included in the total path cost calculation. Figure 3.2 shows an overview of how a MUP node calculates the total path using each method of total path cost calculation.

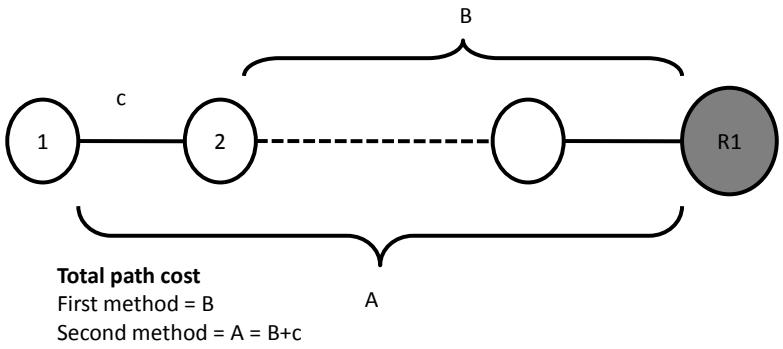


Figure 3.2: Methods of Total Path Cost Calculation.

Based on a simulation study, the findings indicate that the first method has the main advantage of providing stability to the routing protocol where a path is not influenced by fluctuations in the link quality between a node and its neighbour. This routing stability is very crucial during the disaster phase when packets in the network are focused through the unsafe nodes in order to exploit the energy of these unsafe nodes before they are totally damaged in the disaster. This situation can cause the link between a node and an unsafe node to become unstable due

to interference. The possibility of two or more nodes transmitting packets at the same time, when the unsafe node becomes active, is higher in this condition. If the link cost between the node and its neighbour is included in the total path cost calculation, which is the second method, the node will change its path away from the unsafe node to other safe nodes even with small changes in the link quality. In order to ensure that a node does not keep sending packets when the link quality to its unsafe parent is very poor, a threshold is set. If the link cost is over the threshold, nodes simply include the current link cost into their total path cost calculation and another route can be established. Let's consider that a MUP node, N, needs to calculate the total path cost through a neighbour node, n. $C(N \rightarrow n)$ denotes the total path cost through the neighbour, $link\_cost(N \rightarrow n)$ denotes the link cost between node N and the neighbour node, and $C(n)$ denotes the sum of the link costs from the neighbour node to the sink. The complete formula for both methods of total path cost calculation can be expressed as the following:

First method,

$$C(N \rightarrow n) = \begin{cases} C(n), & 0 < link\_cost_{(N \rightarrow n)} < threshold \\ C(n) + Link\_cost_{(N \rightarrow n)}, & link\_cost_{(N \rightarrow n)} \geq threshold \end{cases} \tag{3.1}$$

Second method,

$$C(N \rightarrow n) = C(n) + link\_cost_{(N \rightarrow n)} \tag{3.2}$$

Where C(n),

$$C(n) = \sum_{i=0}^{n} link\_cost_i \tag{3.3}$$

3.2.3 Best Parent Selection

The best parent selection mechanism selects a node that has the minimum path cost, to avoid energy being wasted in unnecessary transmissions. At the same time, the selected nodes should be in an unsafe condition in order to exploit the energy of these nodes before being burnt in the fire. This means that the mechanism gives high priority to nodes with the lowest path cost and in an unsafe condition, leading to them becoming the best parents. Otherwise, nodes with the higher path cost and nodes in the safe condition are given less priority.

The decision about best parent selection uses the total path cost to the sink and the node health status as parameters. Firstly, the mechanism filters nodes with the lowest path cost. If just one node has the lowest path cost, then that

node will be selected as the best parent. However, if there is more than one node with the same lowest path cost, the mechanism considers the node's health status by giving priority in the following order: UNSAFE, LOWSAFE and SAFE. If this is still tied, the mechanism simply remains with the current best parent to achieve network stability. The decision algorithm can be described as follows:

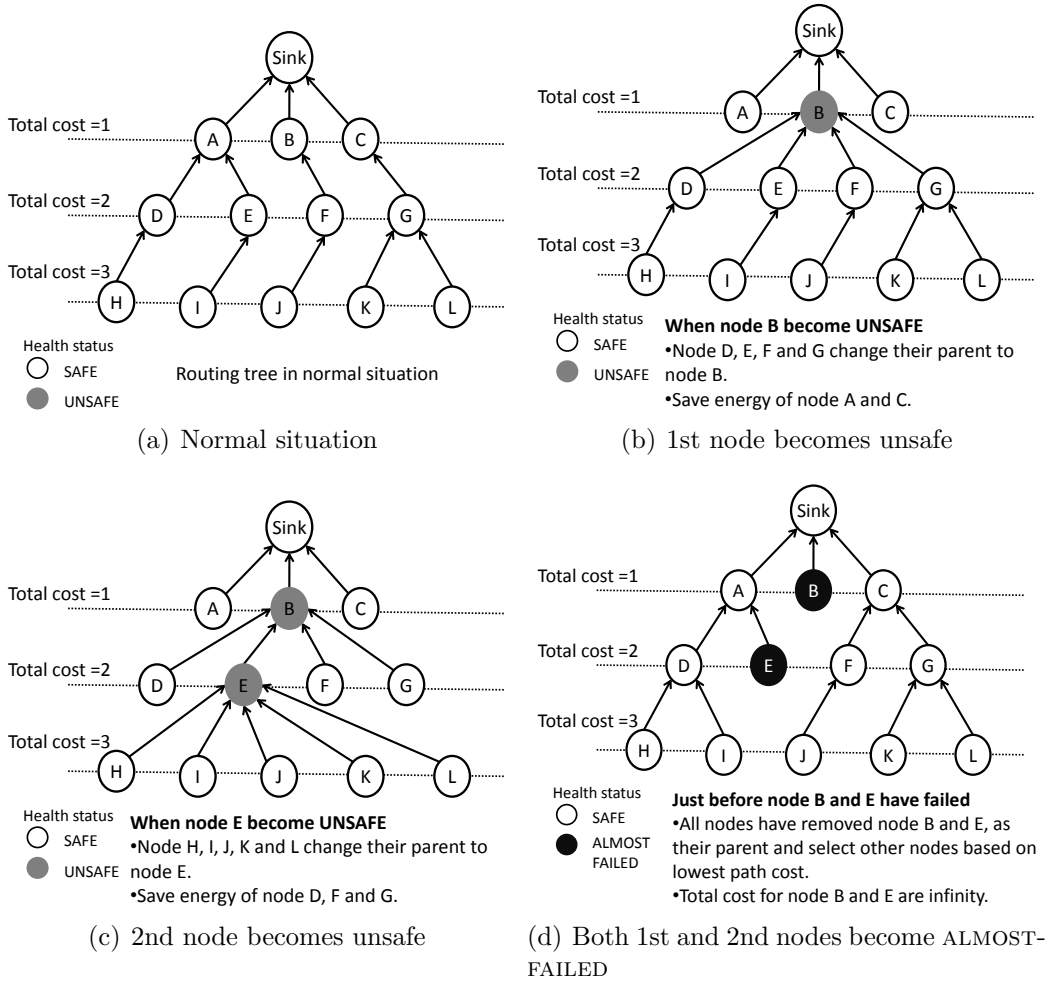
- **First:** Find the node with the lowest path cost.
- **Second:** If there is more than one, then select the node according to its health status in the following priority order: UNSAFE, LOWSAFE, and SAFE.
- **Third:** If this is still tied, remain with the current best parent.

ALMOST-FAILED nodes are not considered as next hops as the probability is high that they will fail soon and cause a broken path and packet loss. This is easily done by increasing the path cost of ALMOST-FAILED nodes to infinity. Other routes are then discovered that avoid these nodes.

3.3 Operation of MUP

Figure 3.3(a) shows a constructed routing topology for a WSN in the normal situation. All nodes have the SAFE health status because there is no disaster event occurring. In the initial step of constructing the routing topology, the sink advertises information about the network using update routing messages. Only the nodes located within the communication range of the sink are able to receive the messages, and these nodes are nodes A, B and C. Upon receiving the messages, each node makes a decision about whether to join the network and selects the sink as their best parent. Afterwards each of them computes its total path cost to the sink. Since all nodes in the network act as routers, the nodes send update routing messages containing the network information with updated total path cost to all the neighbouring nodes. Nodes D, E, F and G, located within the communication range of nodes A, B and C, receive the update routing messages. Those nodes repeat the same process of joining the network, selecting a parent (which is based on the best parent selection mechanism) and advertising update routing messages. This is repeated until all nodes have joined the network and have a route to the sink.

When node B detects that a disaster event has started to occur in the network, as shown in Figure 3.3(b), the node changes its health status from SAFE to UNSAFE. The node sends update routing messages quickly in order to inform its neighbours about its current health status. Upon receiving the update routing messages, the



(a) Normal situation

(b) 1st node becomes unsafe

(c) 2nd node becomes unsafe

(d) Both 1st and 2nd nodes become ALMOST-FAILED

Figure 3.3: Operation of MUP.

neighbour nodes process the messages and make routing decisions based on the best parent selection mechanism. In this case, nodes D, E, F and G change their parent to node B, which allows these nodes to forward packets through node B to utilise the node's energy before it becomes damaged in the disaster. Node B is selected as the parent because the best parent selection mechanism gives a higher priority to this node, which has an UNSAFE health status, compared to the previous parents which have a SAFE health status; all of them have the lowest total path cost to the sink. However, nodes A and C retain the sink as their parent, which has the lowest total path cost compared to node B. By concentrating packets through node B, there is no packet going through nodes A and C, which can save the energy of these nodes.

The disaster event spreads to node E. The node changes its status from SAFE to UNSAFE after the disaster event is detected, as shown in Figure 3.3(c). The same process as with node B is repeated. The node needs to send update routing messages to its neighbours informing them about its current health status. The neighbouring nodes that are just receiving these messages have to adjust their

route accordingly. In the case of nodes H, I, J, K and L, these nodes change their parent to node E. This decision is made because node E, with its UNSAFE health status, has been given a higher priority compared to their previous parents, nodes D, F and G, which means that all nodes have the lowest total path cost (ETX = 2). Node E needs to handle more packets than before it detected the disaster event, which drains the node's energy more quickly before it becomes damaged. The energy of nodes D, F and G can be saved since no packet is going through these nodes at this time.

When node B and E are almost damaged, these nodes need to change their health status again from UNSAFE to ALMOST-FAILED. The nodes send update routing messages once again to inform their neighbours about their current health status. Before these nodes send update routing messages, the total path cost is set to infinity to indicate that these nodes are not suitable as intermediate nodes because they will become damaged at some point soon. By having an infinite total path cost, the neighbouring nodes can avoid sending packets through nodes B and E by removing these nodes as their candidate parents and allowing other nodes to be selected as the new parent as shown in Figure 3.3(d). This is very important to avoid packet loss when nodes B and E suddenly become damaged in the disaster event.

To summarise, MUP always aims to exploit the energy of unsafe nodes by concentrating packets through these nodes during a disaster situation. These nodes become important intermediate nodes which need to handle most of the packets and forward them to the sink. At the same time, the safe nodes become less important and handle fewer packets in the network. As a result, the energy of the safe nodes can be saved which can prolong network lifetime during the disaster phase.

3.4 RPL Design Overview

RPL is a distance vector routing protocol. RPL uses a Destination Oriented Directed Acyclic Graph (DODAG) to ensure that the topology is constructed in such a way that no cycle is present. The DODAG is routed at a single destination which is called a DODAG root (a sink or a base station). The graph is constructed based on a routing metric that is computed by OF. The OF specifies how routing constraints and other functions are taken into account during topology construction.

RPL tries to avoid routing loops using a metric called a Rank that shows a node's position relative to the DODAG root. The Rank is calculated based on a routing metric or constraints. The Rank may be equal to a simple hop-count

distance, may be calculated as a function of the routing metric or it may be calculated with respect to other constraints. The Rank value increases if nodes move away from the root and decreases when nodes move closer to the root.

The WSN is designed specifically based on application scenarios. For different applications, it has a different network requirement. For example, a DODAG may be constructed whereby the ETX metric is considered to achieve energy-efficient data transfer and a high packet delivery ratio. Other than this, in some cases, a DODAG may be constructed by considering the current amount of battery power of a node to achieve energy balance for packet delivery. For this reason, one or more DODAG roots may belong to an RPL instance, which is built over an existing physical infrastructure.

Four types of control message are defined for topology maintenance and information exchange. These control messages are: DODAG Information Object (DIO), DODAG Information Solicitation (DIS), Destination Advertisement Object (DAO) and Destination Advertisement Object Acknowledgement (DAO-ACK). The DIO message is the main source of routing control information. It carries information that allows a node to discover an RPL Instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG. The DIS message allows for a node sending a request for DIO messages from a reachable node. The DAO message supports downward traffic and is used to propagate destination information upward along the DODAG. The last type of message is the DAO-ACK, which is sent by a DAO recipient in response to a DOA message. Typically, the DIO and DIS messages are sent as multicast packets. Otherwise, the DAO and DAO-ACK messages are sent as unicast packets. All four types of control message are defined as ICMPv6 information messages.

Another important feature that needs to be considered in the routing protocol design is the maintenance of the topology. The routing topology must be updated and maintained when inconsistencies are detected in the network, such as changes of route, sensor node failure and bad communication links, to ensure that packets can be delivered successfully to the sink. However, this feature introduces redundancy of messages, which consumes energy because this is not the actual data collected. Since Low Power and Lossy Network (LLN) are typically battery powered, it is crucial to limit the amount of control messages over the network. In RPL, the sending rate of DIO messages is controlled by a Trickle timer which is able to adapt with changes of network condition. When the condition of the network is stable, the control messages will be rare, but when the condition of the network changes frequently, RPL sends control messages more often. Most routing protocols broadcast control packets at a fixed time, which causes energy to be wasted when the network is in a stable condition.

RPL supports three basic traffic flows: multipoint-to-point (MP2P), point-to-multipoint (P2MP), and point-to-point (P2P).

1. Multipoint-to-point traffic

   - Multipoint-to-point (MP2P) is a dominant traffic flow in many LLN applications. The destinations of MP2P flows are designated nodes that have some application significance, such as providing connectivity to the larger Internet or core private IP network. RPL supports MP2P traffic by allowing MP2P destinations to be reached via DODAG roots.

2. Point-to-Multipoint Traffic

   - Point-to-multipoint (P2MP) is a traffic pattern required by several LLN applications. RPL supports P2MP traffic by using a destination advertisement mechanism that provisions Down routes toward destinations (prefixes, addresses, or multicast groups) and away from the root. Destination advertisements are used to update routing tables as the underlying DODAG topology changes.

3. Point-to-Point Traffic

   - RPL DODAGs provide a basic structure for point-to-point (P2P) traffic. For an RPL network to support P2P traffic, a root must be able to route packets to a destination. Nodes within the network may also have routing tables to destinations. A packet flows towards a root until it reaches an ancestor that has a known route to the destination. In the most constrained case (when nodes cannot store routes), that common ancestor may be the DODAG root. In other cases, it may be the node closest to both the source and the destination.

3.5 Upward Traffic

Upward routing enables all nodes in a network to send environment data (such as temperature and humidity) to a common data sink, sometimes called a gateways or root node. In a typical WSN, each node periodically generates data packets at a certain interval (such as every minute) which have to find their way through the network. RPL provisions routes upward towards DODAG roots, forming a DODAG optimised based on an OF. RPL nodes construct and maintain these DODAGs using DIO messages.

3.5.1 DIO Message Structure

A DIO message contains the main source of information which is needed during topology construction. A DIO message allows a node to discover an RPL Instance, learn its configuration parameters, select a DODAG parent and maintain the DODAG. Figure 3.4 shows the DIO message structure.

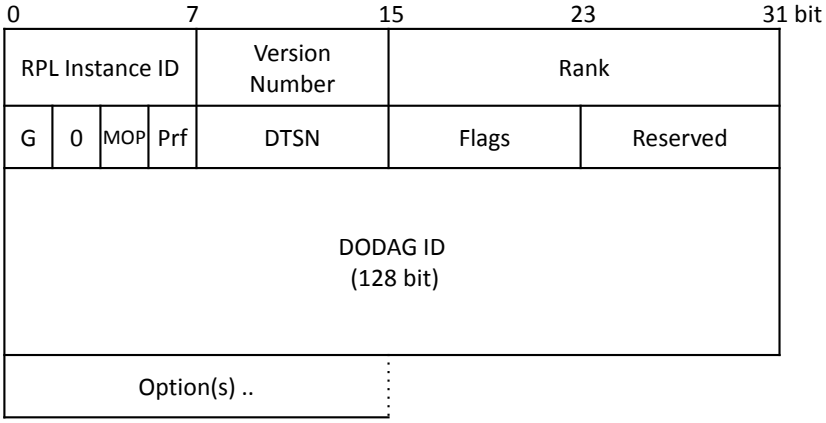


Figure 3.4: DIO message structure [100].

The first field indicates the RPL Instance ID. The second and the third field include the DODAG Version number and the Rank of the sender. After this, the 'G' flag indicates whether the DODAG can satisfy an application-defined goal. If the DODAG can satisfy the application-defined goal, it is considered Grounded. Otherwise, the DODAG is said to be Floating. This may happen when a DODAG is disconnected from the rest of the network and just supports connectivity to its node. The MOP field defines the mode of operation of the RPL instance for downward routing and is set by the DODAG root. All nodes that join the DODAG must be able to accept the MOP in order to fully participate as a router, or else they must only join as a leaf node. RPL supports four types of MOP, which are: no downward routes of operation, non-storing mode of operation, storing mode of operation with no multicast support and storing mode of operation with multicast support. Each MOP has its own encoded value. The Prf field is called the DODAG preference, which is a 3-bit unsigned integer that defines how preferable the DODAG root is compared to other DODAG roots within the instance. This field ranges from 0x00 to 0x07, where 0x00 is for the least preferred and 0x07 is for the most preferred. The next field is the Destination Advertisement Trigger Sequence Number (DTSN) field. The DTSN field is needed for saving a sequence number that is maintained by a node issuing the DIO messages to guarantee the freshness of the message. The Flag and Reserved fields are unused. This field must be initialised to zero by the sender and must be ignored by the receiver. The last used field is for the DODAGID which is a 128-bit IPv6 address set by a

DODAG root that uniquely identifies a DODAG.

The DIO message can be extended by the use of options. An important option that plays a crucial role in parameters exchange is the DODAG Configuration Option. It is used to distribute the configuration information of the DODAG operation for each node in the DODAG. Since this information is generally static and unchanging within the DODAG, it is not necessary to include it in every DIO message. This information is configured at the DODAG root and other nodes must not modify it when propagating the DODAG Configuration option. Figure 3.5 represents the structure of the option.

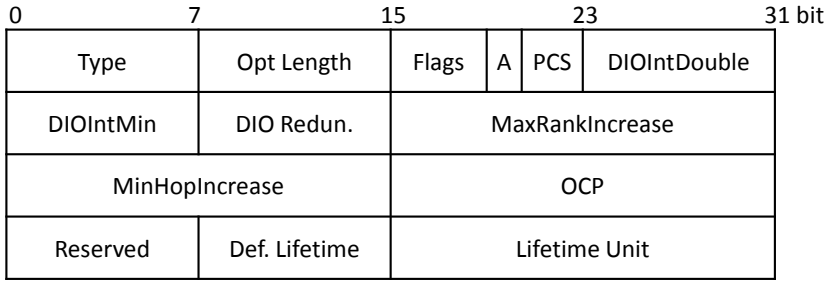


Figure 3.5: DIO Configuration Option [100].

The first byte always includes the type of the DODAG configuration option, which is set to 0x04. The following byte is for the Option Length which always takes the value of 0x14 (indicating 14 bytes options length). After this, the 4-bits that are unused in the Flags field are reserved for flags. The remaining used bits in the Flags field are for the Authentication Enabled (A) and Path Control Size (PCS) flags. The 'A' flag is a 1-bit flag describing the security modes of the network, i.e. whether a node must be authenticated with a key authority before joining the network as a router. The PCS flag is a 3-bit unsigned integer used to configure the number of bits that may be allocated to the Path Control field. These two flags will not be discussed further here, as they are not important for this research. The next two bytes define the maximum interval (I_{max}) and the minimum interval (I_{min}) needed for the trickle timer. The DIO redundancy field is an 8-bit unsigned integer used to define a constant k for suppressing the DIO messages. If a node sends more than k DIOs, it may suppress them. Disabling suppression (which can be done by setting the constant k value to 0x00) means that every node will always send on every interval. The maxRankIncrease field defines an upper limit for the Rank. The MinHopIncrease field stores the minimum increase of the Rank between a node and any of its parent nodes. The next field is OCP field, identifies the OF. The DODAG Configuration Option concludes with the Default Lifetime for all routes and the Lifetime Unit (defined in seconds).

3.5.2 Constructing Topologies

WSNs do not have a predefined topology compared to those imposed by point-to-point wired technology. Thus, RPL has to discover the links and then select its peers sparingly. A network consists of one or more RPL instances which may be connected to the same backbone infrastructure. The RPL instance consists of a set of independent DODAGs. An RPL Instance may provide routes to certain destination prefixes, reachable via the DODAG roots or alternate paths within the DODAG. A RPL Instance may comprise [100]:

1. *A single DODAG with a single root.*
   For example, a DODAG optimised to minimise latency rooted at a single centralised lighting controller in a Home Automation application.

2. *Multiple uncoordinated DODAGs with independent roots (differing DODAGIDs).*
   For example, multiple data collection points in an urban data collection application that do not have suitable connectivity to coordinate with each other, or that use the formation of multiple DODAGs as a means to dynamically and autonomously partition the network.

3. *A single DODAG with a virtual root that coordinates LLN sinks (with the same DODAGID) over a backbone network.*
   For example, multiple border routers operating with a reliable transit link, (e.g. in support of an IPv6 Low-Power Wireless Personal Area Network (6LoWPAN) application) that are capable of acting as logically equivalent interfaces to the sink of the same DODAG.

4. *A combination of the above as suited to some particular application scenario.*

In general, there are three types of nodes in an RPL network; roots, routers and leaf nodes. The root nodes act as gateway nodes which provide connectivity to another network. The router nodes could become intermediate nodes and may advertise topology information to their neighbour. The leaf nodes do not advertise DIO messages and just have the ability to join an existing DODAG.

The DODAG building process starts at a root node which is configured by the system administrator. The root begins to send information about the graph using DIO messages. Each node that receives the messages makes decisions based on certain rules concerning whether to join the DODAG or not. Once the node has joined a DODAG, the node chooses an appropriate parent. The choice is based on the OF, which uses information about metric and constraints in the network. Afterwards each of them computes its Rank within the DODAG, which indicates the coordinates of each node in the graph hierarchy relative to the root. In case a

node acts as a router, it sends the graph information with updated Rank to all the neighbouring nodes. Those nodes repeat the same process of joining the DODAG, selecting a parent and advertising DIO messages until all nodes have joined the DODAG and have a route to the root. In the case of a node as a leaf, it just joins the graph and does not send any DIO message.

Four values have to be considered in order to uniquely identify and maintain the topology. The first is the RPLinstanceID which identifies a set of one or more DODAGs and is called the RPL Instance. Each DODAG may be optimised for a given scenario of application. The second is the DODAGID which is unique for each DODAG. The combination of RPLInstanceID and DODAG ID identifies a single DODAG in the network. The third is the DODAG version number which is incremented each time a DODAG reconstruction is needed. The combination of RPLInstanceID, DODAGID, and DODAG version number uniquely identifies a DODAG version. The fourth is Rank which establishes a partial order over a DODAG version, defining individual node positions with respect to the DODAG root. Figure 3.6 depicts an example of an RPL instance comprising three DODAGs with DODAG roots R1, R2 and R3.

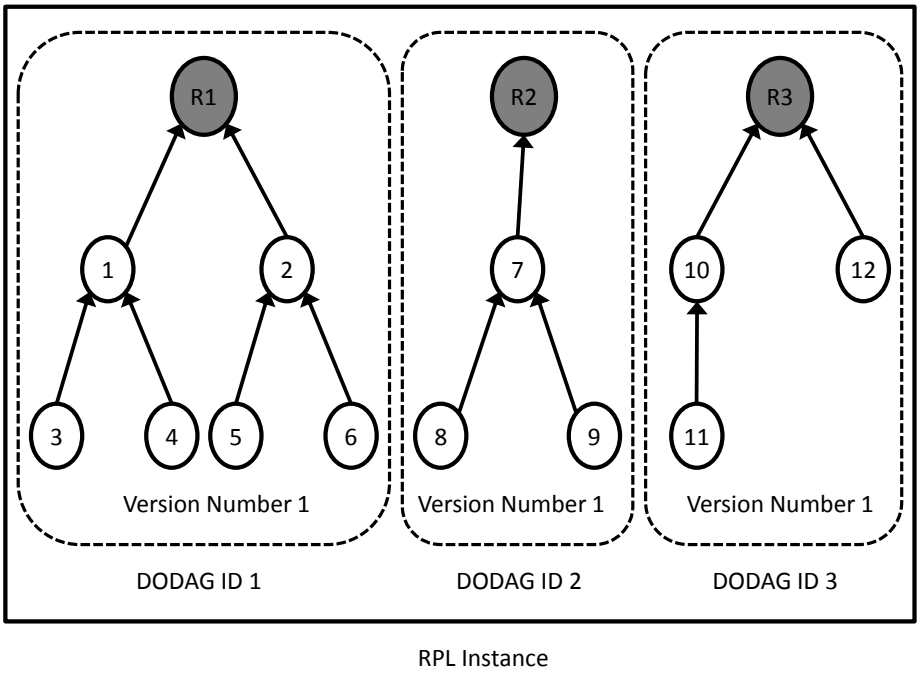


Figure 3.6: RPL topology [92].

RPL introduces three logical sets when building upward routes. Firstly, the candidate neighbour set which includes all reachable nodes. They may belong to different DODAG versions. Secondly, the parent set which is a subset of the candidate neighbour set. It includes only those nodes that belong to the same DODAG version. Thirdly, the preferred next hop parent selected from the parent

set, based on OF. For example, based on Figure 3.6, if node 6 only has three reachable nodes, which are node 2, 5 and 8, these nodes are included as its candidate neighbour. However, only node 2 and 5 will be stored in the candidate parent set because they have the same DODAG ID. Node 8 belongs to a different DODAG, which is not included in the set of candidate parent. Node 6 selects node 2 as the preferred parent since it has the lowest path cost, because the location of node 2 is closer to the root than node 5.

As mentioned, RPL dynamically adapts the sending rate of its control DIO messages which is controlled by the Trickle Timer. The range of sending rate is defined by two values, the minimum sending time interval and the maximum sending time interval. During the initial stage of topology construction, the sending rate is set to the minimum interval. After the timer has expired, RPL doubles the interval. When the network is stable, RPL keeps doubling the interval until it reaches to the maximum interval and maintains this value. Whenever RPL detects an event which indicates that the topology needs maintenance, it resets the timer back to the minimum interval.

3.6 Objective Function

The OF function is not an algorithm and is separated from the core protocol specification. This separation of OFs allows RPL to be adapted to meet the different optimisation criteria required by the wide range of deployments, applications, and network designs [100]. The OF is used by an RPL node to select and optimise routes within an RPL instance, especially during DODAG construction when the network is just established. In an RPL instance, each DODAG has a specialised OF. In addition, the OF function is used to compute the Rank, which is a value representing an abstract distance to the root of the DODAG within the DODAG Version. The Rank is exchanged among nodes in the network for the loop detection and avoidance mechanism. RPL defines two OFs, which are the Objective Function Zero (OF0) and the Minimum Rank Objective Function with Hysteresis (MRHOF). In this research, the MRHOF is selected as the RPL OF because it can provide better Rank and route stability than OF0, which is a very important criterion for critical situations such as in forest fire scenarios.

The MRHOF is an OF that minimises the node rank in terms of a given metric, while using hysteresis to prevent excessive rank churn [33]. The MRHOF is also designed to find the paths with the smallest path cost. If the path cost of the current path is larger than the path cost of the minimum path cost by a given threshold, the MRHOF switches the route to the path with the minimum cost. The MRHOF may be used with any additive metric as long the routing objective

is to minimise the given routing metric. This means that the MRHOF cannot be used if the routing objective to maximise the metric. Here, RPL uses ETX [20] as the default metric in the MRHOF to compute the Rank and total path cost. The ETX is a dynamic metric that dynamically changes depending on the condition of the network, making it suitable to be used in critical situations rather than a static metric. The MRHOF can also support any additive metric such as hop count, link quality level, latency and node energy. The use of hysteresis delays the effect of the changes in the link metric on parent selection, which makes the topology stable.

The RPL node computes the path cost for each candidate neighbour that is reachable within its communication range. The path cost represents the total link cost of the path, in terms of the ETX metric, from a node to the root of the DODAG through the neighbour. In MRHOF, the root node is set to have the minimum total path cost. For a non-root node, it needs to compute the path cost to the root through each candidate neighbour by adding these two components: the link cost to a candidate neighbour and the path cost in the metric container in the DIO sent by that neighbour. If the link cost for the path through that neighbour is not available, the path cost through that neighbour should be set to maximum. This cost value prevents this path from being considered for path selection. In order to ensure that the path cost value is up-to-date, the path cost must be recomputed each time the link cost to the candidate neighbour is updated and when a node receives a new metric advertisement from the candidate neighbour.

In order to calculate Rank, the MRHOF converts the total path cost value as its Rank. The DODAG root set their rank to the minimum path cost for the selected metric. A non-root node converts its Rank value based on the total path cost through its preferred parent. During the conversion, the node needs to calculate the Rank increase, which is the link cost between the node and the preferred parent multiplied by the unit in MinHopRankIncrease, which by default is set to a constant value of 256. The resulting Rank increase is added to the Rank of the preferred parent $R(P)$ to obtain the Rank of this node, $R(N)$. The formula to calculate the Rank is shown below:

$$R(N) = R(P) + rank\_increase \tag{3.4}$$

$$rank\_increase = Link\_metric * MinHopRankIncrease \tag{3.5}$$

Parent selection is a process for a node to select the best parent among all reachable candidate neighbours within its communication range which have the

same DODAGID as the node's preferred parent. During the process of parent selection, the node needs to compute the total path cost for all the candidate neighbours. After this, the node selects its preferred parent. In the MRHOF, the node must select a candidate neighbour as its preferred parent if the path corresponding to that neighbour is smaller than the path cost corresponding to the rest of the neighbours. However, if the smallest path cost through the candidate neighbours is smaller than the current minimum path cost by less than the PARENT_SWITCH_THRESHOLD (which is set to 0.5), the node just continues to use the current preferred parent.

3.7 Expected Transmission Count (ETX) metric

The ETX metric indicates the quality of a link between two nodes. The ETX of a link is defined as the predicted number of packet transmissions required to send a packet over that link which includes retransmissions. The ETX value varies between one and infinity, with an ETX of one representing a perfect transmission medium and an ETX of infinity indicating a completely non-functional link. The ETX of a link value is calculated using the Exponentially Weighted Moving Average (EWMA) of $\alpha = 0.9$ as in (3.6).

$$ETX_p = \alpha ETX_{p-1} + (1 - \alpha)T_p \tag{3.6}$$

Where ETX_p is the ETX value after sending packet p. ETX_{p-1} is the ETX value before sending packet p with initial value of 5.0. T_p is the number of transmission attempts for packet p. This ETX value is updated after each unicast transmission. Information on how many transmission attempts were made to send a unicast packet is provided by the lower layers stack.

3.8 Trickle Timer

The trickle timer allows nodes in a lossy shared medium to exchange information in a highly robust, energy efficient, simple and scalable manner [56]. The mechanism can reduce the number of communications rate logarithmically with network density to save more energy. RPL uses a trickle timer to control the sending rate of DIO messages. When a node transmits the messages, it is based on the underlying consistency model of the network graph. If it hears inconsistencies, the node tries to solve this problem. Instead of flooding a network with packets, the

trickle algorithm controls the sending rate so that each node hears a small trickle of packets, which is just enough to stay consistent.

There are certain events that are treated as inconsistencies in the network. These inconsistency events are when a node detects packet looping in the network, significant changes in the routing cost and hears a newer version number of the DIO message. As inconsistencies are detected, the nodes reset the trickle timer that allows the node to send DIO messages more often. As the network becomes stabilised, the trickle timer increases which results in fewer DIO messages being sent into the network. This means that the frequency of the DIO messages depends on the stability of the network. In other words, as the network becomes stable, the number of DIO messages decreases exponentially. However, the number of DIO messages increases dramatically in order to solve and fix related issues quickly when an inconsistency is detected.

A trickle timer runs for a defined interval and has three configuration parameters: the minimum interval size called I_{min}, the maximum interval size called I_{max}, and a redundancy constant k. The minimum interval size, I_{min}, is defined in units of time (such as in milliseconds and seconds). In RPL, I_{min} is defined as 4096 ms. The maximum interval size, I_{max}, is defined as the number of doublings of the minimum interval size (the base-2 log(max/min)). RPL defines I_{max} as 8. The amount of time specified by I_{max} is 4096 ms x 256, equal to 1048 s or approximately 17 min. The redundancy constant, k, is a natural number (an integer greater than zero) which is set as 10 in RPL. In addition to these three parameters, the trickle timer maintains three variables: I (the current interval size), t (a time within the current interval), and c (a counter).

When a node starts, the trickle timer algorithm sets I to a value in the range of $[I_{min}, I_{max}]$, i.e. greater than or equal to I_{min} and less than or equal to I_{max}. The algorithm then begins the first interval. At this time, the trickle timer resets c to 0 and t to a random point in the interval taken from the range $[I/2, I]$, i.e. values greater than or equal to $I/2$ and less than I. Whenever the node hears a transmission that is consistent, the trickle timer increments the counter c. At time t, the node transmits if, and only if, the counter c is less than the redundancy constant k. When interval I expires, the trickle timer doubles the interval length. If this new interval length would be longer than the time specified by I_{max} the trickle timer sets the interval length I to be the specified by I_{max}. If the node hears a transmission that is inconsistent, it resets the trickle timer only when I is greater than I_{min}. In order to reset the timer, the trickle timer sets the I value to I_{min} and starts a new interval again. However, the trickle timer does nothing if I is already equal to I_{min} when the node hears an inconsistent transmission. The trickle timer can also reset its timer in response to external events.

Based on the above algorithm, the only time the trickle algorithm transmits DIO message is at time t. This means that there is an inherent delay between a detecting an inconsistency (resetting I to I_{min}) and responding to that inconsistency (transmitting at time t in the new interval). This delay can avoid an immediate responses to the detected inconsistency which can cause a broadcast storm, whereby many nodes respond at once in a synchronised fashion. By making responses follow the trickle algorithm (with the minimal interval size), a protocol can benefit from trickle's suppression mechanism and scale across a huge range of node densities.

3.9 Routing Loops

The formation of routing loops is a common problem in various types of networks, and LLN is not an exception. This problem occurs due to topology changes caused by node failure, broken communication links, and the mobility of the node. A node may select a new route to a given destination when changes in the network topology are detected. If the node selects a neighbour node located at a further distance than its own location, loops may occur. These routing loops lead to network congestion, packet drops, energy waste and delays. However, a fast and reliable detection of such topology inconsistencies is a not sufficient solution for LLNs. Furthermore, a topology repair mechanism consumes extra energy when it is triggered. Therefore, a routing protocol for LLNs has to define a loop avoidance strategy to be considered during topology construction.

In practical situations, RPL does not guarantee a loop free-path even though it has specifies two mechanisms to deal with a routing loop: the loop avoidance and the loop detection mechanism. The loop avoidance mechanism tries to avoid creating a loop during the construction of the network topology. The loop detection mechanism tries to detect a loop caused by topology changes in the LLN. RPL uses this loop detection to ensure that packets make forward progress within the DODAG version and trigger repairs when necessary. RPL does not guarantee tight delay convergence times but it can detect and repair a loop as soon as possible.

3.9.1 Loop Avoidance Mechanism

In the loop avoidance mechanism, RPL specifies two rules that rely on the Rank properties of nodes. Firstly, RPL nodes are not allowed to process DIO messages from nodes deeper than themselves because such nodes are possibly in their own sub-DODAGs. Secondly, a node is not allowed to select as a parent, a neighbouring

node that is deeper, such that the node will end up advertising a value $Rank_{lowest}$ + $Rank_{maxinc}$, where $Rank_{lowest}$ is the lowest Rank the node has advertised in a DODAG version and $Rank_{maxinc}$ is a configurable value specified at the root. This is very important to avoid the formation of looping and to prevent instabilities in the DODAG version.

Consider the example shown in Figure 3.7. The topology is constructed by taking the ETX metric into account and the Rank value equal to this metric. The initial network topology is shown in Figure 3.7(a). Here, each node is allowed to process a DIO control message, even from a lower rank node. If the link between the root node A and node B fails, node B will simply pick its child node as the next hop since the ETX cost to the root is 3. The other possible way to the root is through node C with an ETX cost equal to 4. This situation will create a feedback loop between node B and node D as shown in Figure 3.7(c). Therefore, the first rule plays an important role by not allowing an RPL node processing DIO messages to from nodes at a deeper or higher Rank than itself because these nodes may belong to its own sub-DODAG. Even if node D is reachable for node B, it should not be considered as a next hop node. Instead, node B should initiate local repair by poisoning its routes to leave the network and then rejoin the DODAG version.

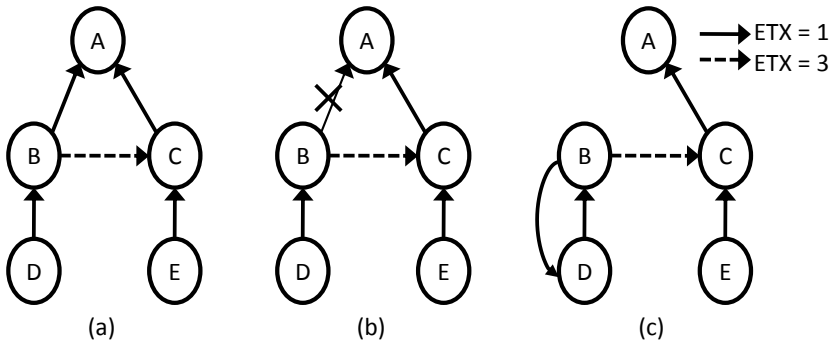


Figure 3.7: Loop Creation [92].

In order to avoid an RPL node becoming greedy when rejoining the DODAG, by moving deeper to increase the number of parents, RPL also limits the movement of a node within a DODAG Version. Moving up in a DODAG does not present the risk of creating loops, but moving down might cause it. Figure 3.8 illustrates nodes in the DODAG when operating under the greedy condition, which will optimise their routes between two parents. Let Figure 3.8(a) be the initial condition in which node A is a DODAG parent for node B and C. Suppose node C is able to leave the DODAG and rejoin at a lower Rank, by taking node A and B as the DODAG parent as depicted in Figure 3.8(b). Now node C is deeper than node A and B and is satisfied to have two parents. Suppose node B in its greediness is

willing to receive and process a DIO message from node C (against the first rule), and then node B leaves the DODAG and rejoins at a lower rank, taking both node A and C as the DODAG parents. Node B becomes deeper than node A and C, and node B is satisfied with two DODAG parents as shown in Figure 3.8(c). Then, node C, because it is also greedy will leave and rejoin at a deeper level to obtain two parents again by having a lower Rank than both of the other two nodes. Next, Node B, because of its greediness, leaves and rejoins deeper in the DODAG. The process will repeat and the DODAG will oscillate between Figure 3.8(b) and Figure 3.8(c), which will cause instability. In this situation, the second rule plays a very important role to prevent RPL nodes becoming too greedy, and leaving the DODAG and rejoining at a deeper or lower Rank than its own Rank. This rule ensures a node cannot be influenced beyond some limit into instability by the action of nodes selecting parents that may be in their own sub-DODAG.

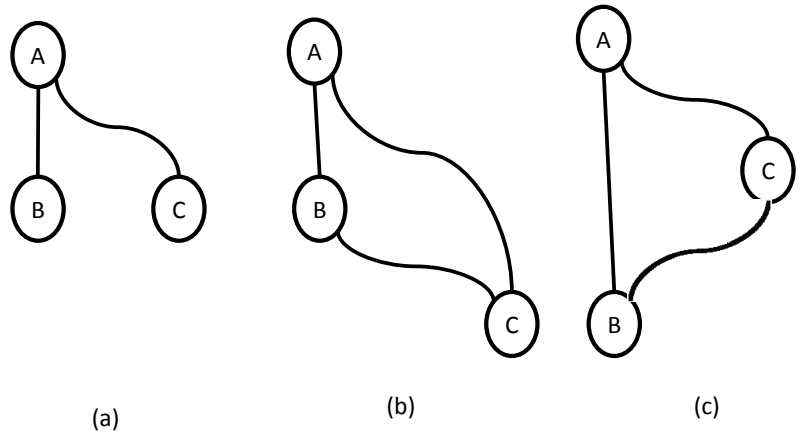


Figure 3.8: Greedy DODAG Parent Selection [100].

3.9.2 Loop Detection Mechanism

Normally, LLN nodes rarely generate data packet, which means that keeping the topology consistent all of the time can waste energy. Since nodes in the LLN have a limited energy supply, it would be very costly to track variations of physical connectivity in the LLN. For example, if the topology is kept consistent, it may happen that nodes experience a lack of energy after a certain period of time, which affects their radio communication capability. That is the reason why transient and infrequent changes in connectivity in the LLN need not be addressed until data packets are present.

RPL uses additional information that is transported in the data packets. The RPL Packet Information is placed in the IPv6 option field, which is updated and examined on each hop. There are five control fields within the RPL Packet Information, but only two of them are used in loop detection. The first one

indicates whether the packet has been sent in a upward or downward direction. The second is the Rank of the sender. A possible loop is detected based on the relationship between direction of traffic and comparison of Rank values (between the sender node and the receiving node). For example, if a node receives a packet flagged as moving in an upward direction, the packet records that the sender Rank is lower than the receiving node, then the receiving node is able to conclude that the packet has not progressed in the upward direction and a loop is detected. Otherwise, for the downward direction, a loop is detected if the sender Rank is higher than the receiving node.

3.10 Local and Global Repair

When failures occur, it is very important for any routing protocol to be able to repair the routing topology. Similarly, RPL supports graph repair mechanisms in the case of link and nodes failures. RPL specifies two techniques for the graph repair mechanisms, which are local repair and global repair. Firstly, local repair only takes place locally within the DODAG, without any implication on the entire graph. A local repair is triggered by a node when a link or neighbour node failure causing the node has no other path in the 'up' direction, in order to find an alternate path quickly, without giving much consideration to the optimal path. During the local repair process, the node removes all its existing parents and waits for the next periodic sending of a DIO by its neighbours. After receiving a DIO message, the node will select the sender as its new best parent and rejoins the network. In order to prevent loops in the routing tree, it is very important for the node to poison the routing tree after a local repair by announcing its own rank to be infinity. By poisoning the routing tree, this process triggers a change of parent or a local repair in the children. This means that a condition where the node attaches itself to its own child, thus creating a loop, can be avoided. As local repairs take place, the graph may start to diverge from its optimum shape, until at a certain point it might be necessary to rebuild the graph using the second mechanism mentioned, which is global repair.

Global repair is a repair mechanism that rebuilds the graph entirely from scratch. The global repair can be triggered only from the DODAG root by incrementing the DODAG version number which initiates a new DODAG version. Each node in the new DODAG version can choose a new position in the graph by having a new Rank value based on the OF which is not constrained by the node Rank within the old DODAG version. When a node receives a DIO message with a new DODAG version which is greater than its current DODAG version, the node participates in the global repair. The node updates its DAG information

based on the new DODAG version, and adds the sender to its parent candidates. In addition, the node calculates its new Rank in the DODAG graph. The global repair mechanism can be considered as an optimisation technique, but has a cost of additional control traffic in the network. This repair mechanism resets the trickle timer to the minimum packet interval, causing nodes to send DIO messages rapidly and therefore cause possible energy waste and congestion of the network.

3.11 Downward Routing

Another important key feature of RPL is the support of downward routing. RPL uses Destination Advertisement Object (DAO) messages to construct and maintain downward routes. This feature supports P2MP traffic flows from the DODAG root toward the leaves, which allows a network administrator to control the nodes that are not even in range. Besides this, downward routes also required to support P2P flows, which is achieved by sending messages toward a DODAG root (or a common ancestor) through an upward route, then away from the DODAG root to a destination through a downward route.

RPL supports two modes of operation for downward traffic: the non-storing mode and the storing mode. The non-storing mode makes uses of source routing. In this mode, each node has to provide information about its parent list up to the root. After receiving such topology information, the root computes the path to the destination. The storing mode is fully stateful. Thus, each non-root and non-leaf node has to maintain a routing table for possible destination. Any given RPL instance is either storing or non-storing.

3.11.1 DAO Message Structure

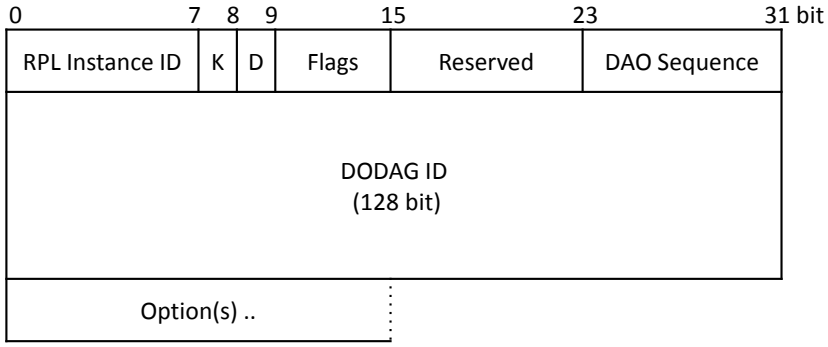


Figure 3.9: DAO Message Structure [100].

Figure 3.9 represents the structure of a DAO message that is used to propagate routing information in order to enable downward traffic. The first field is the

RPL Instance ID which is similar to a DIO message. This value refers to the identification of an instance that the nodes have learned from the received DIO during the construction of the topology. The next field is the Flag field where only the first two bits are used, but the remaining 6 bits which are unused become reserved for future flags. The first bit is the 'K' flag which indicates whether the sender expects the recipient to send a DAO-ACK back. The second bit is the 'D' flag which indicates if a DODAGID field is present. This flag must be set when a local RPLInstanceID is used. The DOA Sequence field contains a sequence number that is incremented at each unique DOA message sent by the sender and is echoed in the DAO-ACK message. After this, the DODAGID is a 128-bit unsigned integer, set by a DODAG root that uniquely identifies a DODAG.

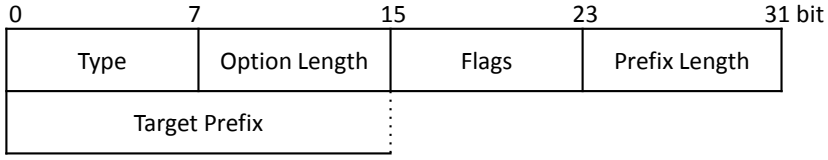


Figure 3.10: DAO Target Option [100].

The DAO message can be further extended using the options field. Here, only two options will be discussed, which are the Target option and Transit Information option. First, the Target option is used to indicate the reachability information of a target IPv6 address, prefix, or multicast group. This option may be repeated as necessary to indicate multiple targets. RPL defines the structure of the option as shown in Figure 3.10. The first field stores the type of the option which is set to 0x05. After this, the Option Length field shows the length of the option in octets, excluding the Type and Option Length fields. The Flags field is unused, it must be initialised to zero by the sender and must be ignored by the receiver. The next field is the prefix length, containing the number of valid leading bits in the IPv6 prefix. The last field is the target prefix, which may identify, for example a single node or a whole group of nodes that can be reached via a common prefix.

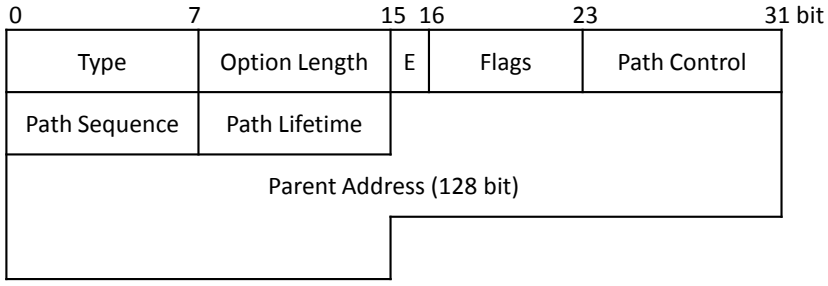


Figure 3.11: DAO Transit Information Option [100].

Secondly, the Transit Information option is used to indicate the attributes for a path with one or more destinations. The destinations are indicated by one or more

Target Options that precede the Transit Information option(s). Figure 3.11 shows the structure of the DOA Transit Information option. The first field indicates the type of the option, which is set to 0x06. The second field is the Option Length field, which indicates if the DODAG parent Address sub-field is present. In the case of the storing mode, the parent address may be excluded. The next field is the Flags field. Only the first field of the Flags is used. This field, called the 'E' flag, indicates whether the parent router redistributes external targets into the RPL network. The fourth field is the Path Control field which limits the number of DOA parents to which DOA messages may be sent. The fifth, the Path Sequence field, indicates if a Target Option with updated information has been issued. The last field is the Path Lifetime, which defines how long a prefix for a destination should be kept valid. The time is measured in Lifetime Units and is implementation specific. A value of all one bits (0xFF) represents infinity. A value of all zero bits (0x00) indicates a loss of reachability.

3.11.2 Non-storing Mode

In the non-storing mode, each node generates DAO messages to report its DAO parents to the DODAG root. Each node has to extend the DAO message by using the Transit Information Option to store DOA parent addresses. This field must contain one or more addresses. A typical non-storing node may use multiple Transit Information Options in order to report its complete parent set to the root. The DAO message is sent to the DODAG root along the default route established during construction of the DODAG graph. Figure 3.12 illustrates this process.

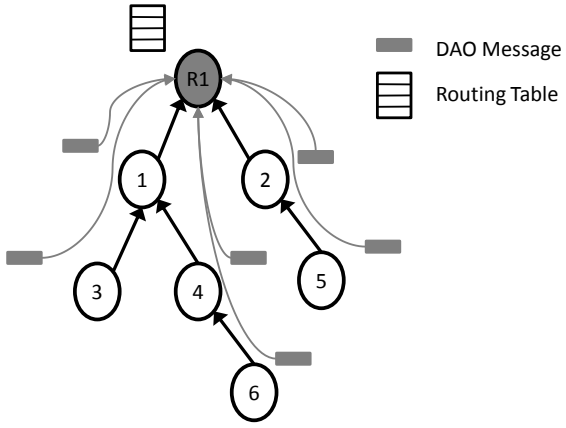


Figure 3.12: Non-storing Mode [92].

Intermediate nodes inspect DOA messages and compare the stored path sequence number with the one last seen. In this way, a node can distinguish between stale and up-to-date routing information. This per-hop calculation is very important to minimise the number of messages when the DOA parent changes. Other-

wise, if the nodes reported complete source routes, when the DAO parent changes, the entire sub-DODAG would have to send new DAOs to the DODAG root. Besides this, the nodes pack received DAOs by sending a single DAO message with multiple RPL Target options. Each RPL Target option has its own, immediately following the Transit Information Options.

After collecting all the information from the nodes, the DODAG root calculates the down route for each node in the network. If it needs to send a data packet to a destination, the IPv6 source routing header is used. Thus, nodes can easily forward data packet until it reaches the given destination or the IPv6 Hop Limit reaches 0.

3.11.3 Storing Mode

In the storing mode, messages are routed downward by the IPv6 destination address. Similar to the non-storing mode, a node in the storing mode is required to generate DAO messages. However, this time a DAO message is no longer propagated to the DODAG root. It is sent as an unicast message to each of its DAO parents, which maintain the additional downward routing table. Figure 3.13 gives an overview of this process.

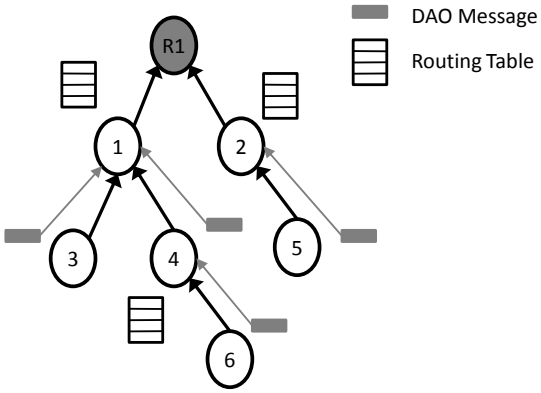


Figure 3.13: Storing Mode [92].

On receiving a unicast DAO, a node must compute if the DAO would change the set prefixes that the node itself advertises. This computation should include consultation of the path sequence information in the Transit Information options associated with the DAO, to determine if the DAO message contains newer information that supersedes the information already stored at the node. An example of the information is receiving a No-path DAO, which is being sent by a node when it removes a node from its DAO parent set to invalidate the existing route.

Since a node's responsibility is not to advertise its parent set but to announce prefixes that are reachable through it, the node has to keep the parent address field

in the Transit Information Option empty. If the device is a router it has to use the Target Option in order to advertise a prefix. Here, multiple Target Options must be used to extend the DAO in case the router has multiple prefixes. If the DODAG root wants to send a data packet, it must be sent only to all one-hop neighbours. Afterwards, the packet is routed downward according to a look-up table until it reaches its destination, or the hop limit in the IPv6 header reaches 0.

3.12 Implementation of the MUP Design Concept in RPL

It is a sensible approach to develop the MUP design concept by reusing an existing routing protocol. In this research, we have implemented MUP as an extension to RPL. This means that MUP is not a totally new routing protocol. The routing management module is implemented by introducing a new Objective Function (OF) module, which is a modification of the existing OF module in RPL. The neighbourhood management module uses the existing Neighbour Discovery mechanism which is available in RPL. The critical event detection module is developed as a new module.

Firstly, the critical event detection module is an important module in detecting disaster events and in determining health status. This module is implemented in the application layer. The module reads the sensor periodically for every second. For each sensor reading, this module determines the environmental threat on a sensor node according to the transition thresholds and represents it as health status as explained in Section 3.2.1. The threshold values must be defined in this module, depending on the applications, as explained in Section 4.6. If the sensor reading is equal to or just exceeding the threshold, MUP has to change its health status accordingly by informing the OF module in order to set the current node health status and to trigger update routing messages.

Secondly, the new OF module for MUP consists of four components: the Rank calculation, the current health status assignment, the total path cost calculation and best parent selection. The Rank calculation component plays an important role in calculating the Rank value of a node. Rank indicates a relative distance between a node to the sink, which is based on the total path cost. Since this Rank value is just for the loop detection mechanism, the method to calculate the Rank remains the same as in RPL, as described in Section 3.6.

The current health status assignment component is a new component introduced in the existing OF module. This component is important for MUP to set

the current health status of a node based on the information from the critical event detection module and the neighbour node's condition. Information from the critical event detection module is used to determine whether the current node's health status should be set either as SAFE, UNSAFE or ALMOST-FAILED. The assignment of a LOWSAFE health status is dependent on the neighbour node's condition. After receiving a DIO message from a neighbour with an ALMOST-FAILED health status, the component sets the current health status to LOWSAFE only if the node is in the SAFE health status. MUP considers any changes of a node's health status as an inconsistency in the network. If this condition happens, the component triggers the DIO messages to inform its neighbours about the node's current health status. One byte of the current health status field is introduced in the existing metric container, which is included in the DIO messages.

In terms of the total path cost calculation, MUP uses the ETX metric as a parameter to calculate total path cost, which is the same metric used by RPL. MUP has introduced two methods of total path cost calculation as explained in Section 3.2.2. The first method of total path cost calculation is a new part introduced in the existing OF. The second method of total path cost calculation is the same method as used in RPL. The formulae to calculate the total path cost for MUP, which are explained in section 3.2.2, can be expressed in terms of the ETX metric as in Equation 3.7 for the first method of total path cost calculation and Equation 3.8 for the second method of total path cost calculation. For the first method of total path cost calculation, the threshold of the link cost between a node and its neighbour is set to 10. If the neighbour has an ALMOST-FAILED health status, the total path cost is set to a maximum. MUP just allows one method of total path cost calculation to be used during the calculation of the total path cost of each neighbour at one time.

First method,

$$C(N \rightarrow n) = \begin{cases} C(n), & 1 \leq ETX_{(N \rightarrow n)} < 10 \\ C(n) + ETX_{(N \rightarrow n)}, & ETX_{(N \rightarrow n)} \geq 10 \end{cases} \tag{3.7}$$

Second method,

$$C(N \rightarrow n) = C(n) + ETX_{(N \rightarrow n)} \tag{3.8}$$

Where C(n),

$$C(n) = \sum_{i=0}^{n} ETX_i \tag{3.9}$$

In the best parent selection component, MUP uses information about the total path cost and node health status in determining the best neighbour to act as a parent to forward packets to the sink. Instead of the existing best parent selection component, it only uses information of the total path cost in the routing decision and selects a neighbour with the lowest path cost as the best parent. MUP uses the same approach as RPL of choosing a route to minimise the path cost but if more than one node has the lowest path cost, the health status becomes the decision criteria by giving the highest priority to the unsafe neighbours rather than the other neighbours. The consideration of health status in the routing decision allows MUP to adapt with the environmental conditions. In MUP, the decision algorithm used in the process of selecting the best parent is mentioned in Section 3.2.3. Firstly, the best parent selection component filters nodes with the lowest path cost. If just one node has the lowest path cost, then that node will be selected as the best parent. However, if there is more than one node with the lowest path cost (within the tolerance of 0.5), the component considers the node's health status by giving priority in the following order: UNSAFE, LOWSAFE and SAFE. If this is still tied, the component simply remains with the current best parent to achieve network stability.

There are two implementations of MUP: MUP-single and MUP-adapt. Each implementation has its own OF module. These two OF modules just differ in terms of the way they calculate the total path cost. Other than this, each OF module has the same Rank calculation, the current health status assignment and the best parent selection components. MUP-single uses the first method to calculate the total path cost for each neighbour. Based on the simulation study, it is found that MUP-single suffers a slightly lower packet delivery ratio and end-to-end delay in the normal situation for certain network condition as shown in Figure 5.2 and Figure 5.3 accordingly. These poor performances are caused by the unstable link between a node and its neighbour node that MUP-single does not consider in the total path cost calculation because it uses the first method. MUP-adapt is introduced to overcome the drawback of MUP-single in a normal situation. MUP-adapt adaptively selects a method between the two methods of total path cost calculation based on their neighbours' health status. If both neighbours are in the SAFE health status (always true in normal situations), the second method is used as the total path calculation. On the other hand, if one or both of them is not in the SAFE health status, the node selects the first method to calculate the total path cost.

There are other routing protocols in WSNs which route packets away from the unsafe nodes. Because of this, we have introduced the SAFEST routing protocol

for comparison purposes. SAFEST has the opposite routing concept to MUP. Instead of routing packets through the unsafe nodes, SAFEST aims to route packets through safe nodes. SAFEST is designed based on the MUP-single implementation, with just a modification on the priority order of health status in the best parent selection component. The new priority of the health status for SAFEST is set as the following order: SAFE, LOWSAFE and UNSAFE.

3.13 Summary

This chapter has described the overall system design of MUP. MUP is expected to prolong network lifetime during a disaster situation by exploiting the energy of unsafe nodes that are going to be destroyed soon in order to save the energy of the other nodes in the network. This is achieved by concentrating packets through the unsafe nodes. However, this approach has a trade-off of poor network performance in terms of the packet delivery ratio and end-to-end delay due to congestion and radio interferences. The MUP design concept consists of three functional modules: neighbourhood management, routing management and critical event detection modules. The critical detection module determines node health status based on sensor reading information and informs the neighbourhood management module if there is any changes in the health status. The neighbourhood management module plays an important role in discovering a subset of neighbour nodes as forwarding nodes and maintaining the neighbour table. The routing management module computes the total path cost through each of the neighbour nodes based on the ETX metric. This module sets the health status of the nodes based on information from the critical event detection module and the neighbour node's condition, and triggers update routing management messages if there is a change in the node health status. In addition, this module plays an important role in selecting a neighbour node as the best parent to forward packets to the sink. The MUP design concept is implemented as an extension to RPL and each modification made has been described clearly in this chapter. The source code is available in the GitHub repository via the following link: https://github.com/ansarjamil82/Contiki-sensinode-ansar. In the following chapter, the simulation and the experimental configuration used to evaluate the performance of MUP are discussed.

Chapter 4

Simulation and Experiment Configuration

4.1 Introduction

MUP has been developed and studied through simulation and experiment. This chapter explains the common measurement configuration used in both processes. Contiki OS is selected as an operating system for the sensor nodes. Cooja, a simulator for the Contiki OS, is selected as a simulator and a debugging tool. As the basic network configuration, a network is established consisting of 25 nodes arranged in a grid topology. A forest fire scenario is implemented in the simulation and the experiment. The main parameters measured are the the network lifetime, packet delivery ratio and end-to-end delay.

4.2 Contiki OS

Contiki OS has been selected as the operating system for the sensor devices used in the present research. Contiki is an open source, lightweight operating system developed for the constraints environment in a WSN. Contiki is implemented in C language. It was written by Adam Dunkels from the Networked Embedded Systems group at the Swedish Institute of Computer Science. Typically, a Contiki system consists of a kernel, libraries, a program loader, and a set of processes [25]. A process may be either an application program or a service. The service implements functionality which is used by more than one application process. Communication between the processes always goes through the kernel, which does not provide a hardware abstraction layer, but allows the device drivers and applications to communicate directly with the hardware [25].

Contiki supports dynamic loading and the replacement of individual programs

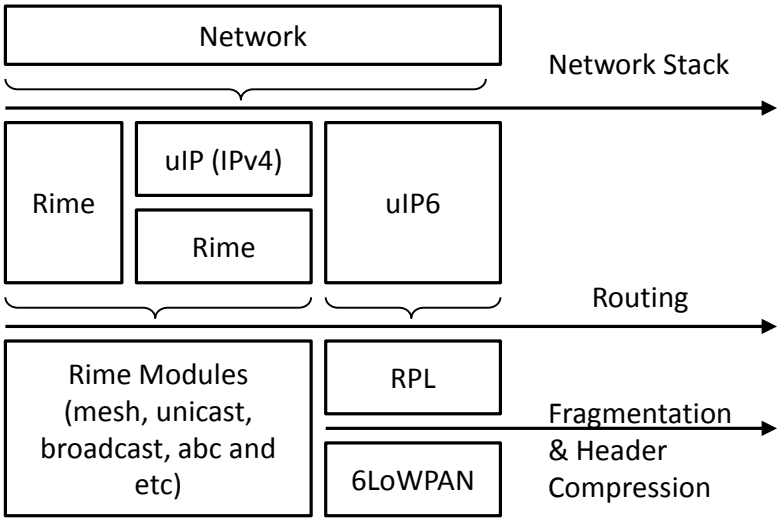


Figure 4.1: The Contiki communication overview [64].

and services. Moreover, optional pre-emptive multi-threading is implemented as a library that is linked on-demand with the programs that explicitly require this feature. The kernel is event-driven, and inter-process communication is carried out using message passing signals. The other distinct features of this OS include the native TCP/IP support using the uIP or lwIP TCP/IP stack, and the graphical subsystem that can be used with either direct graphic support for directly connected terminals, or a networked virtual display with VNC or Telnet [88]. Contiki uses a programming abstraction called protothreads, which frees programmers from the constraint of using state machines whenever working with event driven kernels.

Contiki uses two communication stacks which are uIP and Rime. uIP is a minimal RFC compliant implementation for IP that has a 5kB footprint. It is important to remember that uIP has several limitations: e.g. no IP options, no sliding window, can handle only one network interface, uses a single buffer for both incoming and outgoing packets and does not buffer sent packets. Rime is a new lightweight communication stack designed for low-power radios. Rime provides a wide range of communication primitives, from best-effort local area broadcast, to reliable multi-hop bulk data flooding [26]. Contiki can be configured to use Rime, uIP or combination of both as shown in Figure 4.1.

4.3 Application examples in Contiki

We use existing application examples available in the Contiki OS that are ready to be used during the development of a protocol in a WSN. Three are used in this research: *rpl-collect*, *rpl-udp* and *border-router*. Each application supports

6LoWPAN functionality.

The *rpl-collect* example is used in the simulation process. This application example allows the viewing of routing information for each node in the network during a simulation run. The *rpl-collect* example consists of two main application codes which are *udp-sender* and *udp-sink*. The *udp-sender* allows a sensor node to send data using a UDP packet to the sink periodically. The data is mostly related to the routing information, such as the current parent, the current total path cost, the current trickle timer interval and the current number of neighbours in the routing table. Other information about the number of packets sent (packet ID) and the total duration of each component in active mode (MCU, transmitter and receiver) are also included in the data. The *udp-sink* allows a sensor node to act as the sink. A plugin in the simulator can be used to analyse all data collected by the sink and display the results in the form of graphs. The *rpl-collect* provides adequate information about the characteristics and performance of the developed routing protocol. Any adjustment to the developed routing protocol can be made if necessary to achieve the routing requirement.

The *rpl-udp* is an application example that is ready to be used for the experiment and can be programmed into the actual sensor devices (N740 Nano-Sensor). This application example consists of two main application codes: *udp-client* and *udp-server*. The *udp-client* allows the programmed sensor node to collect the required data and send it using a UDP packet to the sink. The *udp-server* is an application enabling a sensor node to act as the sink, which is the destination node for the collection of data collected by the sensor nodes. The *udp-server* operates on a local network directly connected to the WSN. If required by the deployment, the *udp-server* can operate in any network connected to the WSN through the internet via a border-router.

The *border-router* is an application that allows a sensor node to route IPv6 packets between the WSN and the internet. Connectivity to the internet may be provided through any arbitrary IP link, including Ethernet, WiFi, GPRS or satellite. The IPv6 packet is forwarded by the border router to the tunslip6 interface when it receives a packet from a client node. Tunslip6 is established between the linux machine and the border-router through serial communication using a USB cable. During the initialisation of the tunslip6, the linux machine is assigned with an IPv6 address (one end of the tunnelling) and the address must have the same prefix as the border-router (another end of the tunnelling). The pinging process to either the border router or any node in the network can be done using the ping6 command via a terminal. Because the border-router forwards datagrams at the network layer, it does not maintain any application-layer state.

4.4 Cooja

A plugin in Cooja is used to interact with the simulation. Often it provides the user with some graphical interface and observes the network during the simulation. One of the available plugins in Cooja is the *collect-view* plugin. This plugin can be used with the *rpl-collect* application example. By using this plugin, the data collected by the sink from each node in the network can be analysed and the results are displayed in the form of graphs. Many results can be produced by the *collect-view* plugin, such as the number of received packets over time, the number of packets lost over time, the routing metric, the number of neighbours, the number of hops away from the sink and the trickle timer interval for each node in the network. In addition, the plugin is able to provide information about power utilisation for each node, such as the average power consumption, the average power consumption over time and the average Radio Duty Cycle (RDC). The most important thing is that this plugin is able to display the current routing tree topology of the network. Any changes in the routing tree can be observed clearly, making it easier to develop and understand how the routing algorithm routes packets in the network. Any appropriate improvements in the routing algorithm can be done effectively if needed.

Another plugin used in this research is the *WiseML* plugin [94]. The plugin is developed by the CONET (Cooperating Object Networks of Excellence) research group. This plugin allows an environmental scenario to be included within a normal network simulation, such as a forest fire scenario. The *WiseML* plugin accepts a WiseML file. In this research, the WiseML file contains the mote-time specific health status condition, which is based on the node's temperature model in a normal and a forest fire situation. This WiseML file is parsed by Cooja at simulation setup and the resulting node's health status condition is scheduled in the Cooja simulation loop. The advantage of this approach is the flexibility it gives in creating the scenario, it enables a researcher to set the point of fire breakout anywhere in the network and to create different fire spreading patterns. One drawback of this approach, however, is simulation scalability; it is not suitable in large scale simulations consisting of a hundred or a thousand sensor nodes.

The *timeline* plugin visualises radio traffic and radio usage of WSNs [72]. The plugin shows a timeline for each node in the simulation. The power state of the radio transceiver, radio interference, radio transmission and reception are shown in a colour code on the time line: white if the transceiver is off, grey if the transceiver is on, blue for transmission, green for receptions and red for radio interference. In addition, the plugin shows each of the node LEDs and provides watchpoints.

Other plugins available in Cooja are the *network visualiser*, *log listener* and

DGRM configurator plugin. The *network visualiser* plugin shows information about the network, such as each node position, power state (on or off), node ID, LEDs and radio traffic and radio environment. The *log listener* plugin shows printouts of all simulated nodes which can be filtered using node ID. The *DGRM configurator* plugin [18] is used to create a link between nodes for the DGRM radio model.

4.5 Energest

Contiki's energy consumption estimation module (Energest) [27] is included in the simulation and experiment, to estimate the energy consumption for each node in the network. This module measures the time duration for each of the following components in active mode: i) MCU, ii) Receiver, iii) Transmitter. When a component is switched on, the estimation mechanism stores a time stamp. As the component is switched off, a time difference is produced and added to the total time that the component has been turned on [27]. Then, the time value is converted to estimate the energy consumption using the current draw of the component from the sensor node's datasheet. The total energy consumption, E is defined as:

$$\frac{E}{V} = I_m t_m + I_l t_l + I_t I_t + I_r t_r + \sum_i I_{ci} t_{ci} \tag{4.1}$$

Where V is the supply voltage, I_m is the current draw of the microprocessor when running, t_m is the time during which the microprocessor has been running, I_l and t_l are the current draw and the time of the microprocessor in low power mode, I_t and t_t are the current draw and the time of the communication device in transmit mode, I_r and t_r are the current draw and time of the communication device in receive mode, and I_{ci} and t_{ci} are the current draw and time of other components such as sensors and LEDs.

4.6 Basic Network Configuration

As the basic network configuration, a WSN is deployed in an area consisting of 25 sensor nodes. All sensor nodes are identical and organised into 5×5 grid topology. The network uses mesh communication, in which each node only has the ability to communicate with its adjacent neighbours. A node that is located at one corner of the network is selected as the sink. Other sensor nodes send data to the sink periodically. Figure 4.2 shows the configured network in the simulation and the experiment.

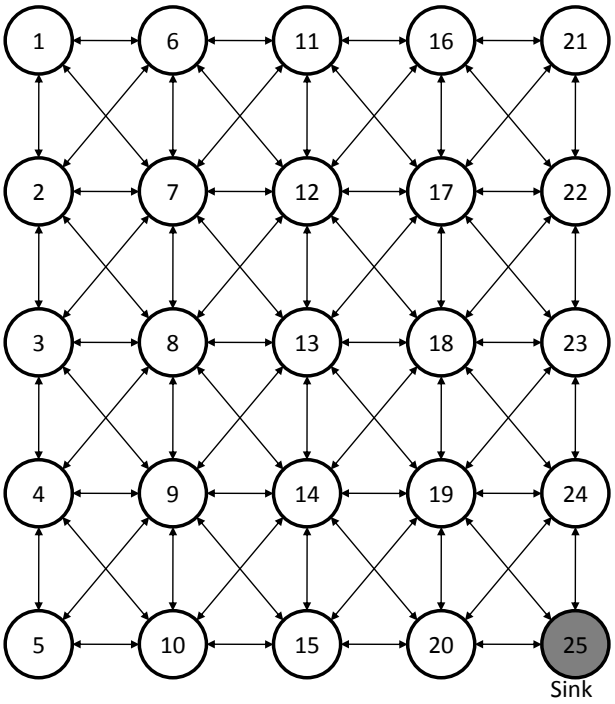


Figure 4.2: Network configuration.

For the simulation, the radio medium used for the network is called the Directed Graph Radio Medium (DGRM). This radio medium transforms the broadcast communication into several unicast communications, each with its own success ratio and propagation delay. The radio links are set to be perfect because we want to determine the performance of MUP under the perfect condition first before proceeding with the real lossy radio links in the WSN test-beds. Since the distance between two nodes is short, the propagation delay is ignored by setting the delay value as 0 s. To allow two nodes to exchange packets, an edge needs to be created between those nodes using the simulation plugin DGRM configurator. In addition, DGRM can be used with the WiseML plugin, which is needed in order to include a forest fire scenario in the simulation. In order to verify the experiment results, we did some simulation runs over lossy link as described in Appendix A.3.

Table 4.1: Software Configuration

Items	Setting
Radio Layer	N740 Nano-Sensor / Sky (IEEE 802.15.4)
MAC Layer	CSMA
RDC Layer	Contikimac
Channel Check rate	16 Hz
6LoWPAN	Yes
IPv6	Yes
Network Layer	MUP-single, MUP-adapt or RPL
Application Layer	rpl-collect, rpl-udp and border-router

On top of the routing protocols, the software component for each node supports uIPv6 with a 6LoWPAN adaptation layer. In order to save energy, Contikimac (which is a default duty cycling mechanism in Contiki) is included with the channel check rate set to 16 Hz. Table 4.1 shows the software configuration used in the simulation and the experiment.

A node detects fire when the temperature increases above a detection threshold that is set at 60 °C. The detection threshold is set slightly higher than the world's highest environment temperature, which is 56.7 °C [70]. However, this detection threshold may vary and must be set slightly higher than the highest recorded temperature according to the place where the network is deployed to avoid a faulty alarm. Just after a fire is detected, the node changes its node health from SAFE to UNSAFE and sends messages to inform all its neighbours about its current status. When the temperature reaches 110 °C, the node changes its health status to ALMOST-FAILED. At this stage, the node will die any time soon. Neighbours located one-hop away from the node are considered to be in a LOWSAFE condition. The node is considered totally burnt in the fire when the temperature has reached 130 °C, which is the maximum operating temperature for the node to function properly. In normal situations without fire, the node is always in the SAFE status. The determination of the health status can be simplified as illustrated in Figure 4.3. In order to visualise the current health status of a node during the simulation and the experiment, LEDs (red and green colours) are used as indicators with the combination as shown in Table 4.2.

Table 4.2: Combination of LEDs to indicate node health status

Health status	LED (Red)	LED (Green)
SAFE	off	on
UNSAFE	on	off
ALMOST_FAILED	on	on
DAMAGED	blink	blink

A forest fire breaks out 100 s after the simulation or the experiment is started, which means that during the first 100 s, all the nodes are in the safe condition and no nodes are failed. This allows the network to fully establish itself before the fire is started. In the simulation, the fire starts at the centre of the network [97] which is at node 13. However, the fire starts at the centre point between node 13, 14, 18 and 19 in the experiment. The fire grows and burns part of the network. The simplest circular shape of the forest fire growth model for a flat terrain without wind and slope [95] is used in both the simulation and the experiment. Five different speeds of fire growth are tested: 1, 2, 3, 4 and 5 $\mathrm{m\,min^{-1}}$. When the expanding fire reaches a node, the temperature is increased rapidly. The applied temperature

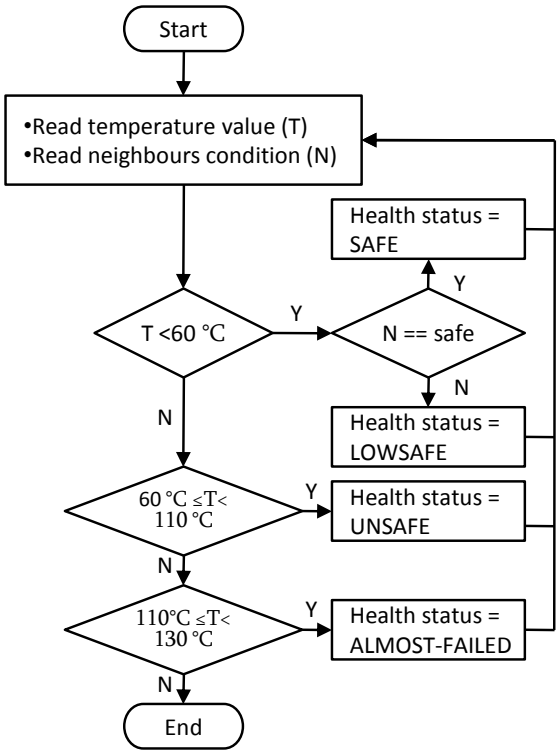


Figure 4.3: Determination of health status.

model is simple; as long as a node is not exposed to fire, its temperature value is always 20 °C. When the node becomes exposed to fire, a linearly growing offset is added to the node's temperature value [97]. Figure 4.4 shows the temperature increment model when fire reaches a node at time 100 s.

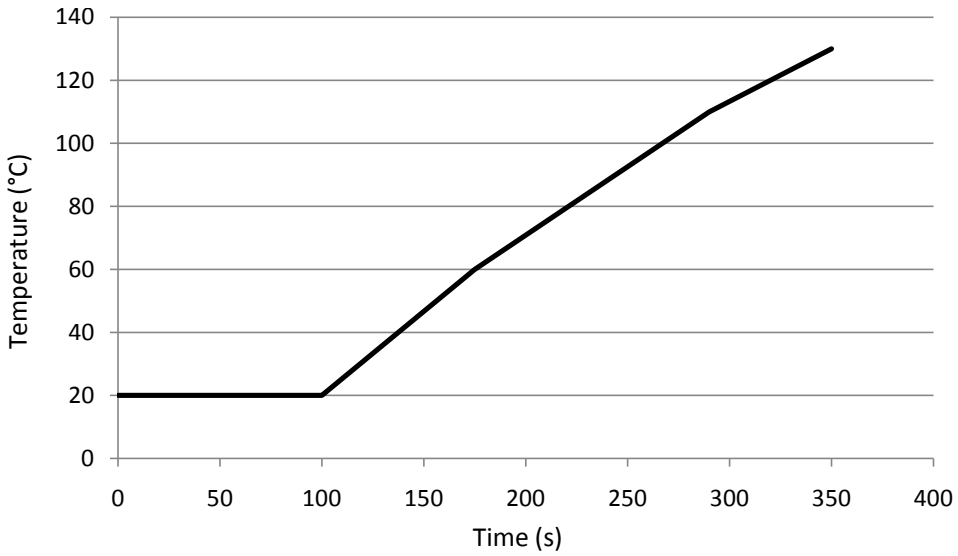


Figure 4.4: Temperature increment model.

4.7 Measured Performance Metrics

The main performance metrics measured in the simulation and the experiment are described as follows:

Network lifetime

There are multiple definitions of network lifetime in WSNs because the point when a network should be considered non-functional is application-specific [15]. It can be, for example, the instant when the first sensor dies [47, 46, 82, 66], a percentage of the sensors die, the network partitions [14], or the loss of coverage occurs [7]. Since MUP only takes place during the critical phase, a suitable definition of network lifetime in this research is the time interval from the start of the fire until a point when at least 50% of the nodes in the network are still functioning and at least have a path to the sink.

Packet delivery ratio

Packet delivery ratio is defined as the ratio of the number of successfully delivered packets to the number of packets sent for each node in the network. It is measured from the start of the simulation or the experiment until the network becomes non-functional. This metric illustrates the level of delivered data to the destination. It is expected that the network should be able to deliver all the packets sent by the node to the sink. A greater packet delivery ratio value indicates the better performance of a protocol.

End-to-end delay

The end-to-end delay is defined as the duration of the time taken to deliver a packet from the source to the sink successfully. It is also includes the delay caused by the route discovery process and the queue in data packet transmission. During the calculation of end-to-end delay, only the data packets that are successfully delivered to their destinations are counted. A lower value of end to end delay indicates the better performance of a protocol.

Total Number of Packets Collected at the Sink

Total number of Packets Collected at the Sink (TPCS) defines the total number of received packets by the sink starting from the time of network deployment until it becomes non-functional. The main objective of a WSN deployment is to collect information about the environmental conditions using sensors, and to send the sensor readings to the sink. It is very important for a network to be able to collect as much data as possible. This means that the more data a network can collect, the more it is considered

to provide a better performance. This metric can be used in selecting the best protocol for a network. For example, if there are two routing protocols: protocol A ($TPCS = 100$ packets and $lifetime = 2$ months) and protocol B ($TPCS = 50$ packets and $lifetime = 2$ months). By referring to the TPCS value, protocol A provides better performance compared to protocol B. Protocol A is able to provide 50 packets more than protocol B within the same network lifetime duration. If we have another routing protocol C ($TPCS = 150$ packets and $lifetime = 1$ month), protocol C becomes the best protocol because it has the highest TPCS value compared to the other protocols, even though protocol C has a lower network lifetime. In the case where two protocols have the same TPCS value, other network performances such as network lifetime can be used as the deciding metrics, which depends on the application requirement.

4.8 Summary

This chapter has presented the common simulation and experimental configuration used during the development and evaluation of MUP. Contiki OS is selected as the operating system for the sensor nodes. Cooja, a simulator for the Contiki OS, is selected as a simulator and debugging tool. Additional functions such as the Energest module and simulator plugins are included in the simulation and experiment as required. For the simulation, we have used the perfect and lossy radio model. The software is configured to have ContikiMAC, 6LoWPAN, CSMA and 'Sky' mote or 'N740 nano-sensor' (for radio layer platforms). We have used existing application examples available in the Contiki OS that are ready to be used for the simulation and experiment: *rpl-collect*, *rpl-udp* and *border-router*. As the basic network configuration, a network is established consisting of 25 nodes arranged in a grid topology. A forest fire scenario is taken as an example of a disaster situation and implemented in the simulation and experiment. The simplest circular shape of the forest fire growth model for a flat terrain without wind and slope is used in both the simulation and the experiment. A linear node temperature increment is used when a node becomes exposed to fire. Temperature thresholds for the detection of a fire event, when a node becomes almost failed and when a node is destroyed in fire, are described clearly in this chapter. The main parameters measured are the network lifetime, the packet delivery ratio and end-to-end delay. In the following chapter, the simulation study of MUP is carried out and of the performance the routing protocol is evaluated.

Chapter 5

Simulation of MUP

5.1 Introduction

In this chapter, the performance of MUP is discussed in detail based on the simulation results using the Cooja simulator. The routing protocol is simulated in a normal situation and in a forest fire situation. The performance of MUP is tested in different conditions such as with variation in the packet rates and forest fire growth speeds. For each simulation setting, we have performed about 30 simulation runs. The main parameters measured in the simulation are the network lifetime, the packet delivery ratio and end-to-end delay. The performance of MUP is compared with RPL and SAFEST.

5.2 Simulation Analysis of MUP in a Normal Situation

In a normal situation, MUP should provides a similar network performance as RPL. Since all nodes in the network are always in the safe condition, MUP selects the route with the lowest total path cost which is the same approach to that used by RPL. Figure 5.1 shows the network lifetime for MUP and RPL in a normal situation. As can be seen in the graph, it is shown that MUP provides a similar network lifetime to RPL. Figure 5.2 shows that MUP achieves a similar packet delivery ratio performance to RPL. The same trend is observed for end-to-end delay, where MUP provides similar performance with RPL as shown in Figure 5.3. However, there is a slightly lower packet delivery and end-to-end delay performance recorded by MUP-single at 0.5 packet/second when compared to RPL. This is because the MUP-single node just considers the link cost from its parent node to the sink in the total path cost calculation. There is a possibility for a packet to be dropped or buffered much longer if the link between the nodes

and its parent becomes unstable. Otherwise, the RPL node considers the link cost from the node to the sink through its parent. As can be seen in the graphs, MUP-adapt has solved the problem of MUP-single by implementing adaptive total path cost calculation that is based on condition of the neighbour nodes.

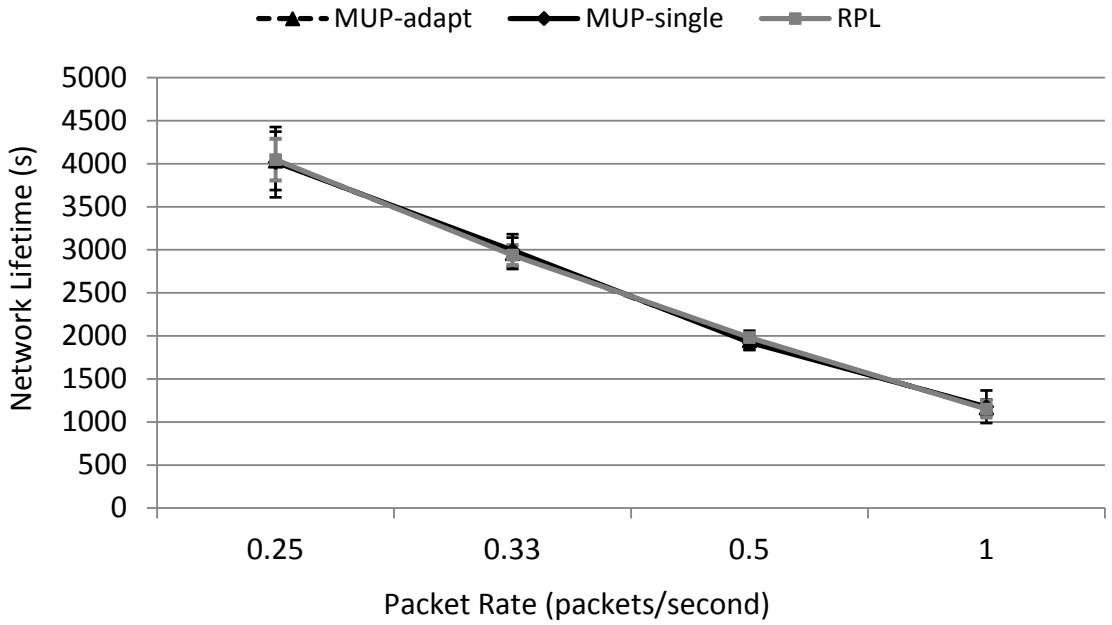


Figure 5.1: Network lifetime for MUP and RPL in a normal situation.

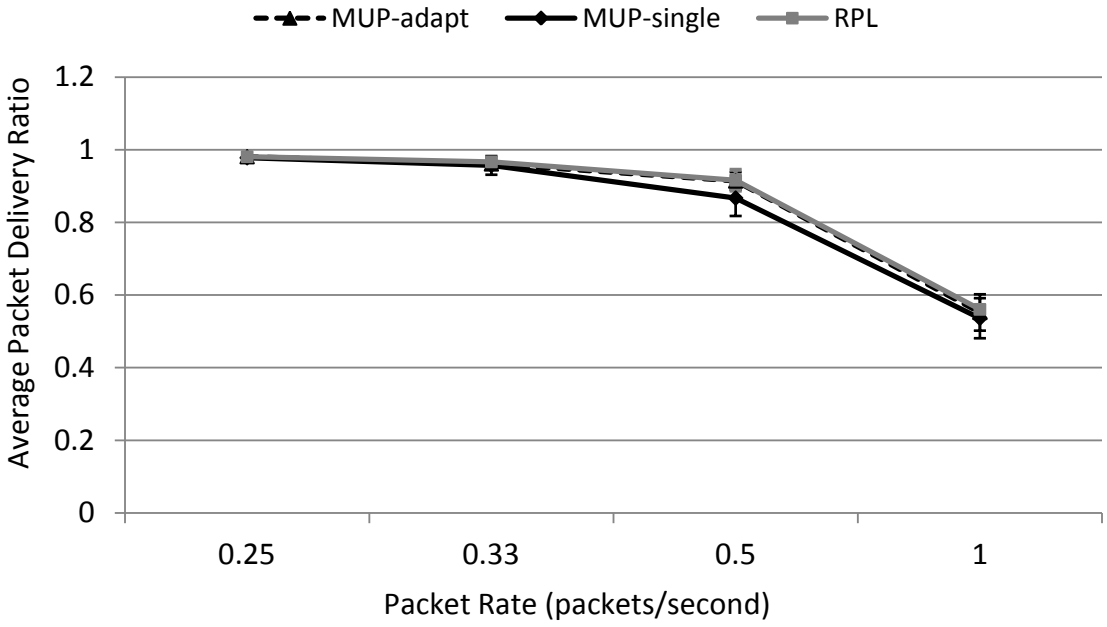


Figure 5.2: Average packet delivery ratio for MUP and RPL in a normal situation.

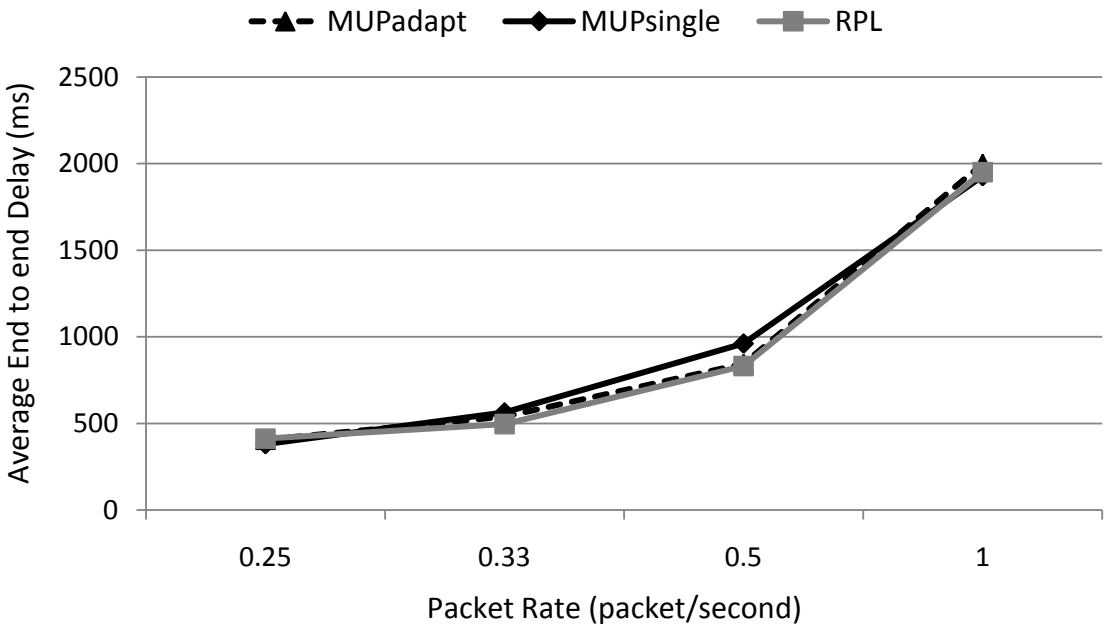


Figure 5.3: Average end-to-end delay for MUP and RPL in a normal situation.

5.3 Simulation Analysis of MUP in a Forest Fire Scenario

5.3.1 Network Lifetime

Figure 5.4 shows the network lifetime over different packet rate values in a forest fire situation. The graph shows that MUP provides a better network lifetime than RPL regardless of packet rates. This is because MUP has the ability to adapt with the environment by considering the node's condition in the routing decision in order to forward packets through the unsafe nodes avoiding the safe nodes. By contrast, RPL does not take any consideration of the node's condition in the routing decision and may use the safe nodes to forward packets to the sink, which shortens the lifetime of the network. The highest network lifetime improvement is achieved by MUP at packet rate 1 packet/second, with the network lifetime being about 30% longer than with RPL under the same circumstances. The higher the packet rate, the greater network lifetime improvement that can be achieved by MUP. In addition, MUP achieves a longer network lifetime compared to SAFEST, regardless of packet rates.

Figure 5.5 to Figure 5.8 show the number of nodes disconnected from the sink over time at different packet rates for MUP, SAFEST and RPL in a forest fire situation. In these graphs, a horizontal line is drawn to indicate the network lifetime threshold, where half of the nodes in the network are disconnected from the sink. The region below the threshold line indicates that the network is still func-

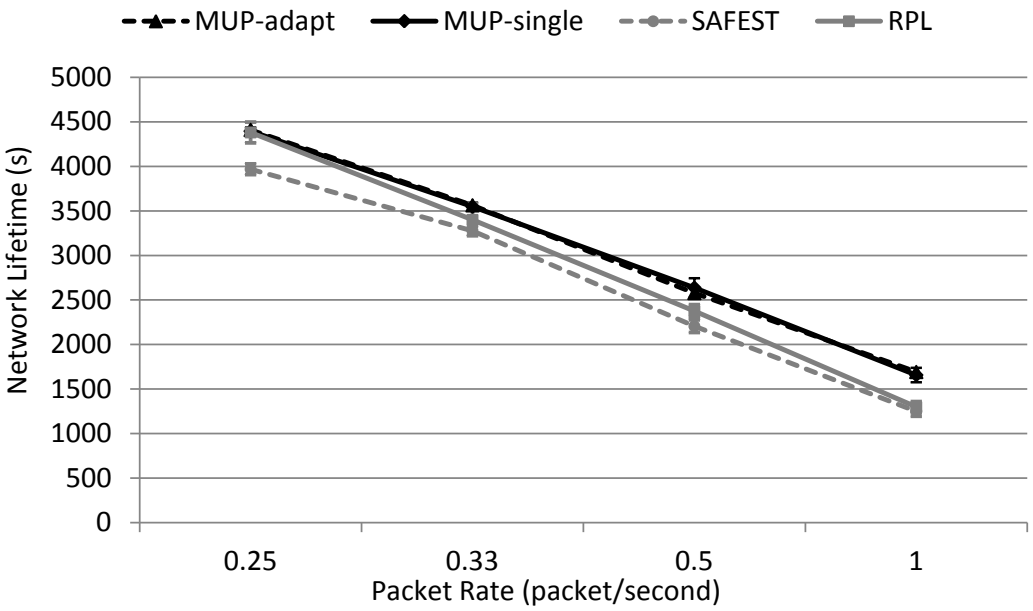


Figure 5.4: Network lifetime over different packet rates for MUP, SAFEST and RPL in a forest fire situation.

tional. Otherwise, the region above the threshold line indicates that the network is non-functional. This means that the network lifetime of the network can be easily observed by finding the crossing point between the response of each routing protocol and the threshold line. The crossing points are drawn with a dotted straight line and numbered with 1, 2 and 3 referring to SAFEST, RPL and MUP accordingly. The results show that MUP provides a longer network lifetime when compared to RPL and SAFEST, regardless of packet rates as expected. SAFEST has the shortest network lifetime when compared to the other routing protocols.

As illustrated in these graphs, each routing protocol has recorded a significant increase in the number of nodes disconnected from the sink as labelled in the graphs. This happens because the nodes located closer to the sink are depleted of energy sooner when compared to other nodes located further away from the sink. This is due to the nodes closer to the sink having to support packet delivery for other nodes located further away. If these nodes become non-functional, other nodes located further away will be disconnected from the sink. This is a well-known issue in WSNs and is called the bottleneck problem.

Figure 5.9 to Figure 5.12 show the time of the nodes to become non-functional due to being burnt in the fire or depleted of their energy at different packet rates for MUP, SAFEST and RPL. Normally, the unsafe nodes will be destroyed due to being burnt in the fire after detecting the disaster event. However, there is an occasion where an unsafe node can become non-functional due to depleted energy. As illustrated in Figure 5.9 and Figure 5.10, in the case of MUP, node 19 has depleted its energy first before it is burnt in the fire. This is happens due to the

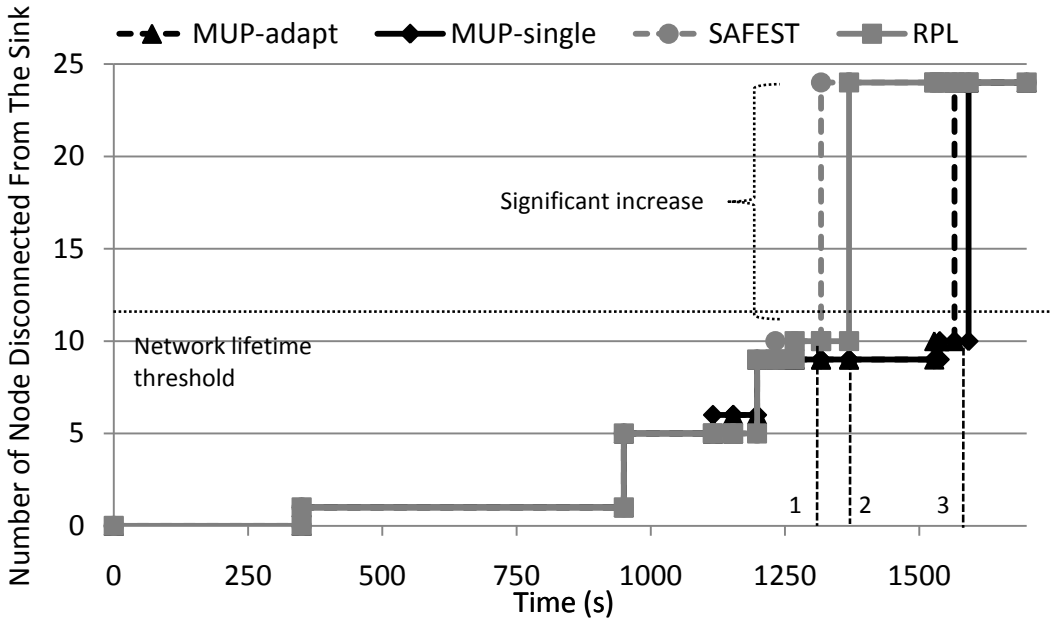


Figure 5.5: Number of nodes disconnected from the sink over time for MUP, SAFEST and RPL at 1 packet/second in a forest fire situation.

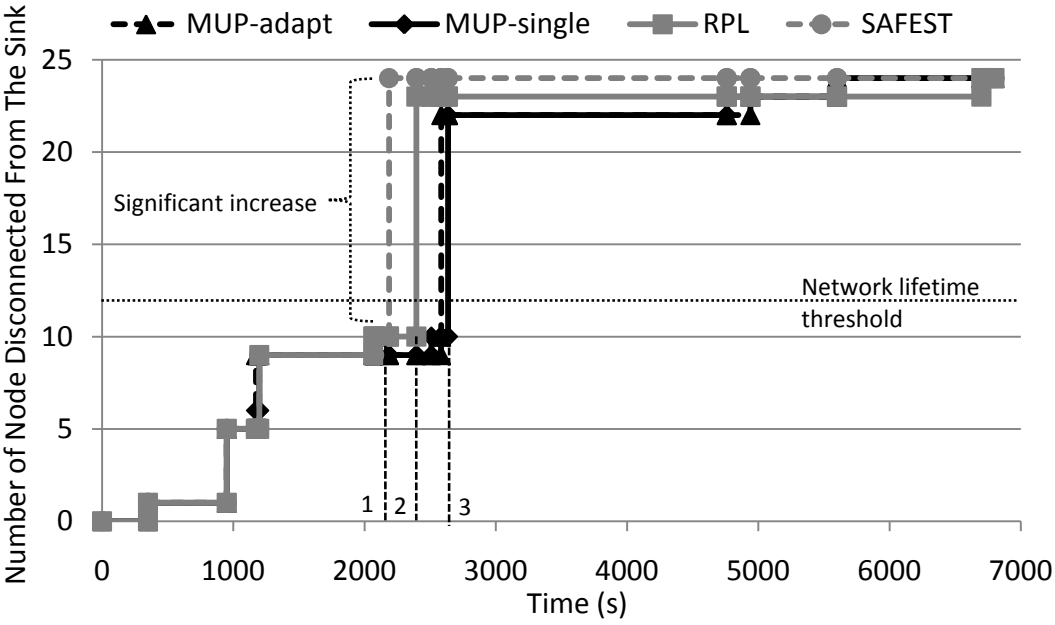


Figure 5.6: Number of nodes disconnected from the sink over time for MUP, SAFEST and RPL at 0.5 packet/second in a forest fire situation.

node needing to handle most of the packets in the network because MUP focuses packets through the node, when the node is in the unsafe condition. Node 19 drains its energy fast until it is totally out of energy before it is burnt in the fire.

After the unsafe nodes become non-functional, the network is divided into two parts, where the remaining safe nodes are separated into two lines of the routing topology. On one side of the routing topology, the line path is going through node 20 and on the other side of the routing topology, the line path is going through

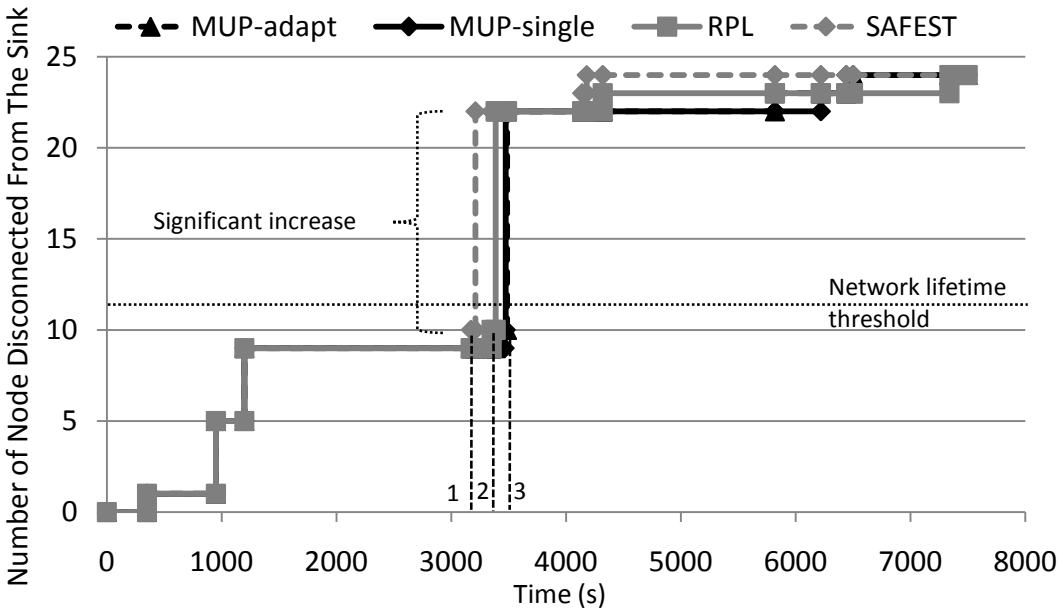


Figure 5.7: Number of nodes disconnected from the sink over time for MUP, SAFEST and RPL at 0.33 packet/second in a forest fire situation.

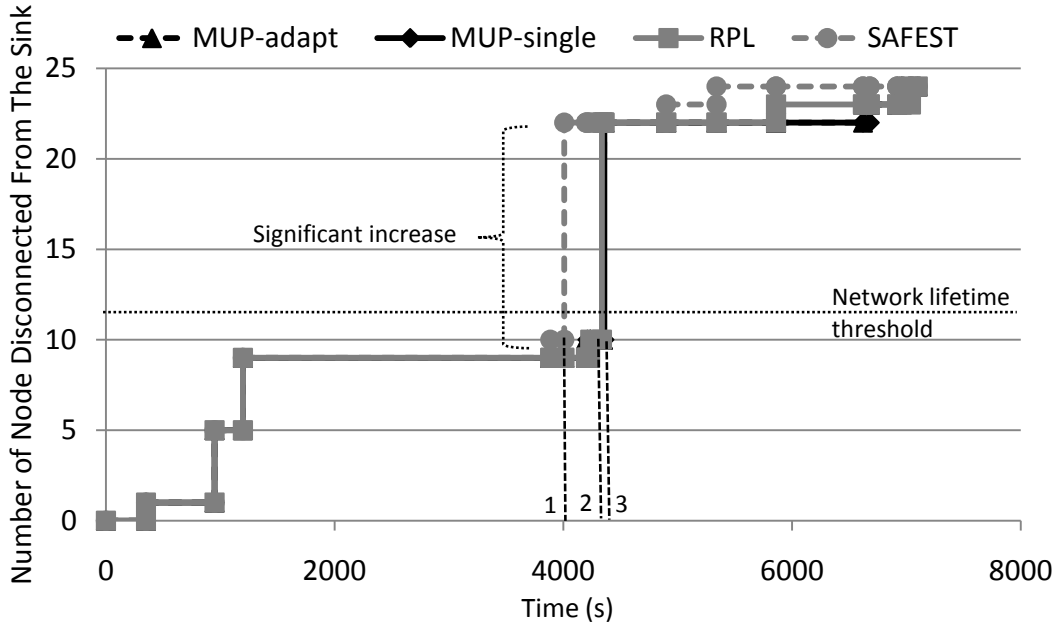
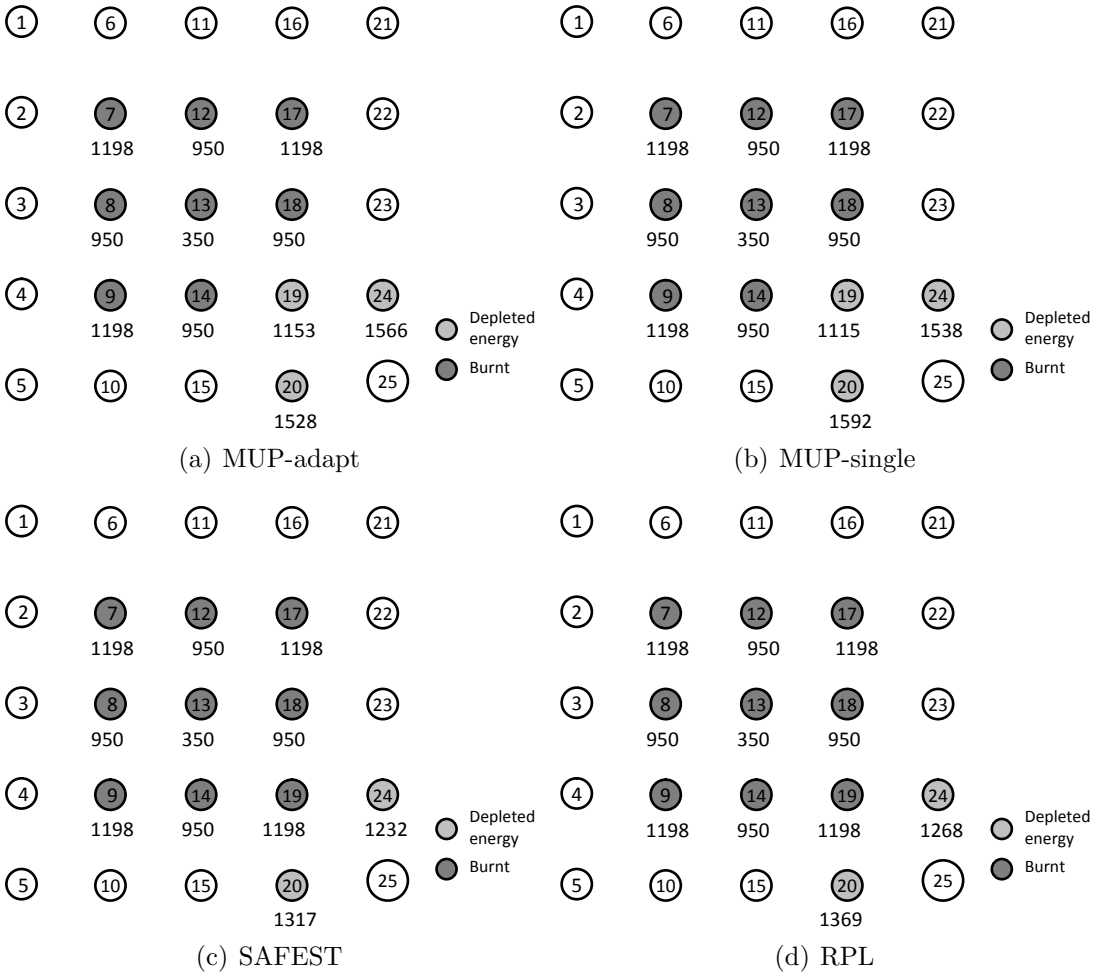


Figure 5.8: Number of nodes disconnected from the sink over time for MUP, SAFEST and RPL at 0.25 packet/second in a forest fire situation.

node 24. Because of that, it is expected that these two nodes will be depleted of their energy much faster compared to the other safe nodes located further away from the sink, as shown in Figure 5.9. However, there are occasions where the nodes located further away may be depleted of energy much faster when compared to nodes located closer to the sink, as shown in Figure 5.10 to 5.12. The results show that the safe nodes located two-hops away from the sink, which are node 15 and node 23 are depleted of their energy much faster when compared to the

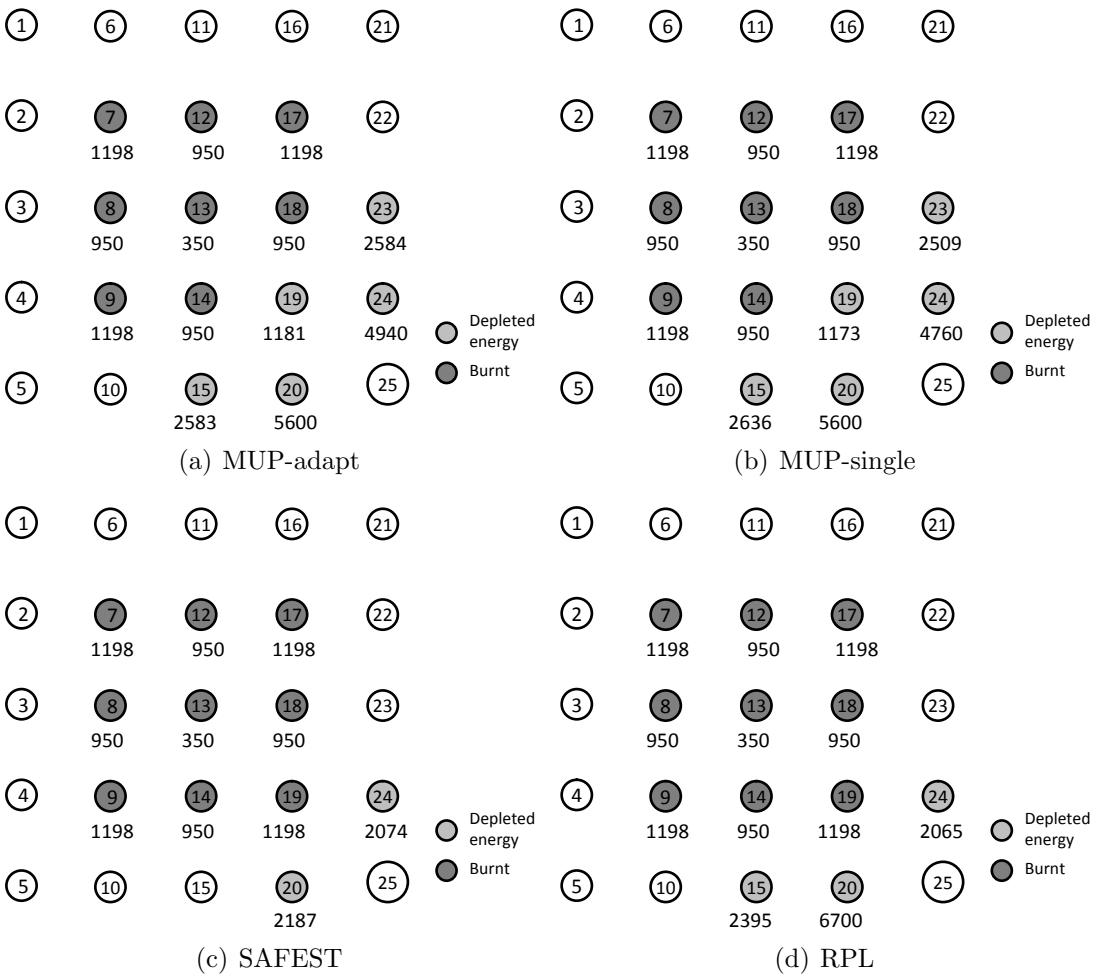
safe nodes located one-hop away from the sink, which are node 20 and 24. When node 15 and 23 have failed, only node 20 and 24 has a connection to the sink and stay alive for longer period of time because the nodes does not support packet delivery for any node. In order to investigate this matter, we have measured the energy remaining of these nodes over time for different packet rate. The complete simulation results can be found in the Appendix A.4.



(a) MUP-adapt (b) MUP-single

(c) SAFEST (d) RPL

Figure 5.9: Time node died (in seconds) for MUP, SAFEST and RPL at 1 packet/second in a forest fire situation.



(a) MUP-adapt (b) MUP-single

(c) SAFEST (d) RPL

Figure 5.10: Time node died (in seconds) for MUP, SAFEST and RPL at 0.5 packet/second in a forest fire situation.

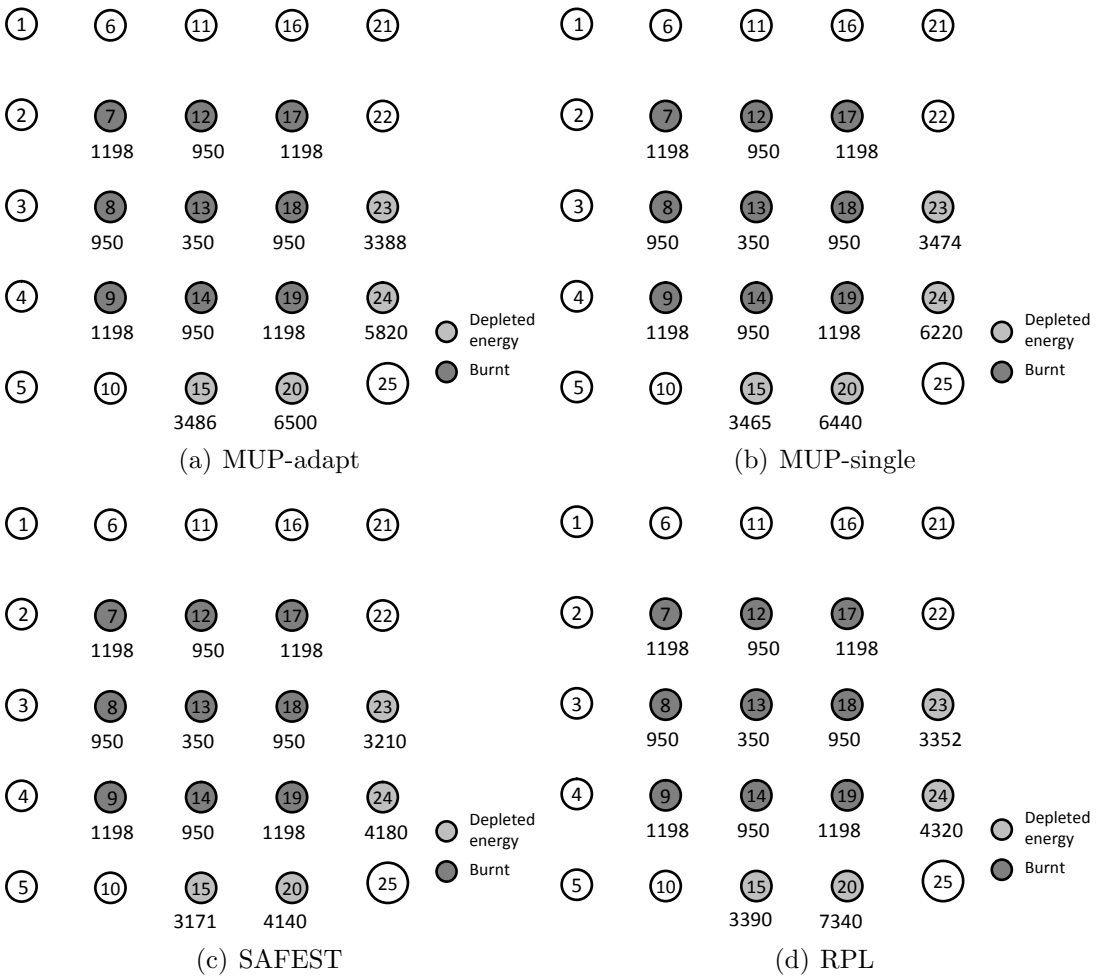
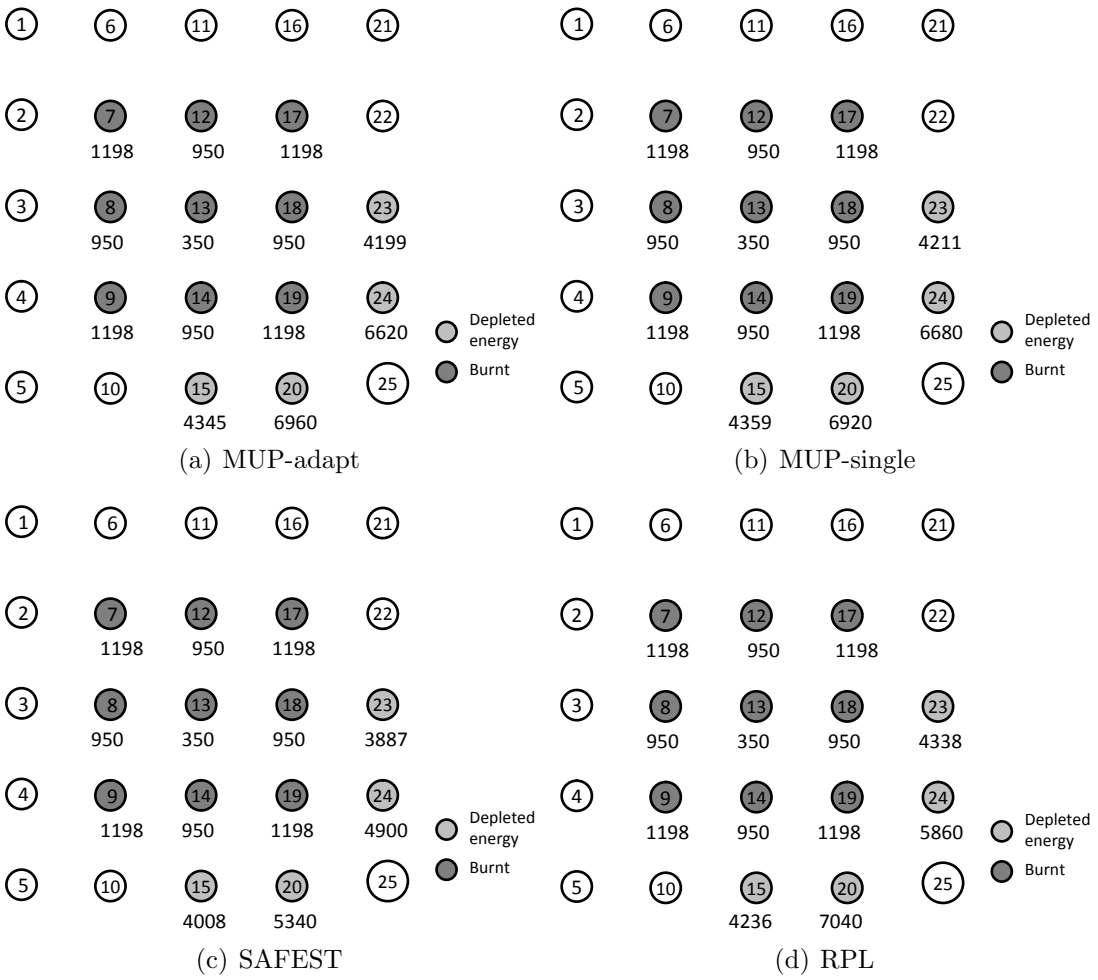


Figure 5.11: Time node died (in seconds) for MUP, SAFEST and RPL at 0.33 packet/second in a forest fire situation.



(a) MUP-adapt (b) MUP-single

(c) SAFEST (d) RPL

Figure 5.12: Time node died (in seconds) for MUP, SAFEST and RPL at 0.25 packet/second in a forest fire situation.

The findings indicate that three factors are causing the nodes which are two-hops away from the sink to be depleted of their energy faster than the node located one-hop away from the sink. The first factor is that the routing protocols do not provide load balancing among the nodes. This means that the remaining energy for each node within the same distance to the sink will not be similar. In a certain case, a node located closer to the sink may have a higher remaining energy than other nodes located further away because the node does not become a part of the intermediate node (does not have a child node) or supporting nodes. Figure 5.13 shows the unbalanced energy between nodes 20 and 24, whereby both nodes are located at the same distance and are always in a safe condition; here node 20 has a much higher remaining energy when compared to the remaining energy of node 24. In addition, the remaining energy of these nodes is much higher when compared to nodes 15 and 23 which are located further away from the sink.

The second factor is that only a path exists to the sink through a certain node in the network after a significant number of nodes are destroyed in the fire. This can cause the energy of that particular nodes, which provides the only path to the sink, to be drained much faster than the other nodes. As can be seen in Figure 5.14, the remaining energy of nodes 15 and 23 drops dramatically just after 950 s, which is labelled with symbol 'I' in the graph. This happens because these nodes are the only available paths to the sink at the time, after the other nodes within the same distance from the sink are destroyed in the fire.

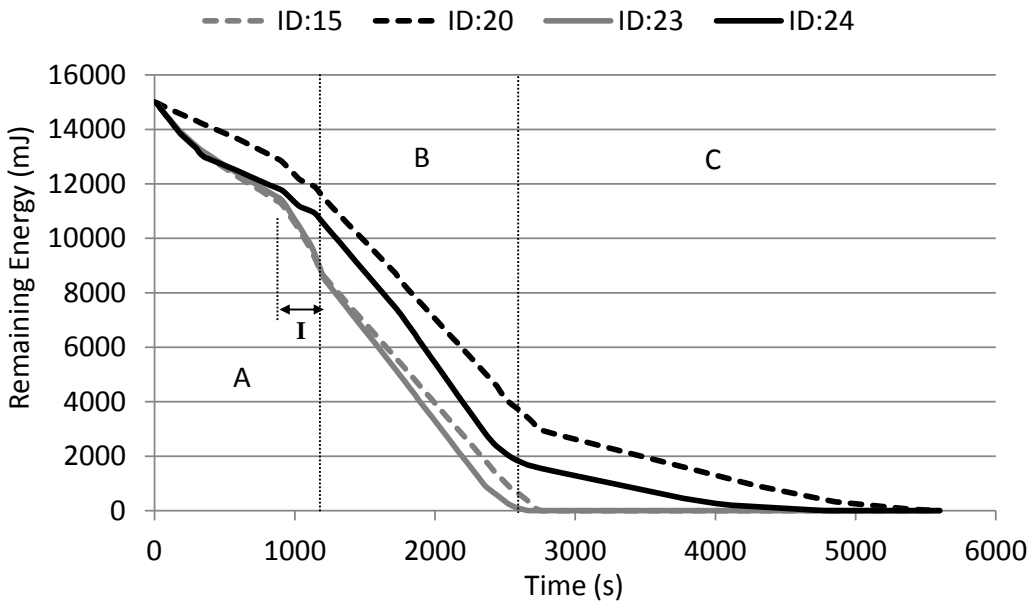


Figure 5.13: Remaining energy of node 15, 20, 23 and 24 over time for MUP (MUP-single) at 0.5 packet/second in a forest fire situation.

The third factor is that the sink is set to always in active mode. Figure 5.14 shows that nodes located closest to the sink, which are nodes 20 and 24, drain

their energy much slower when compared to the nodes located further away, which are nodes 15 and 23, referring to the region B in the graph. This happens because the sink is always in active mode (without a duty cycle) with the assumption that it uses the main power supply, but other nodes in the network are using a duty cycle. Since ContikiMAC is used as the duty cycle mechanism, which is a transmitter initiative approach duty cycle, each node needs to make a continuous transmission of the same packet until it receives an acknowledgement from the destination node or the time interval, which is a single Clear Channel Assessment (CCA), is expired. Upon receiving an acknowledgement, the sender node will transmit all other packets in the buffer. This means that the destination node must stay active for a short period to receive if there are other packets from the sender node. This is the reason why the nodes located closest to the sink require less transmission attempts to deliver a packet successfully; the sink is always in active mode to receive it as shown in Figure 5.15. Solid boxes refer to transmission attempt activities for the nodes closest to the sink, which are nodes 20 and 24. The dotted boxes refer to the transmission attempts for nodes 15 and 23, which are located two hops away from the sink. This situation is becoming significant at a lower packet rate, since the nodes which are two-hops away from the sink need to make more than one transmission attempt to send each packet to the nodes closest to the sink. However at a higher packet rate, the nodes located closest to the sink must stay active in listening mode for a longer period of time to receive all data in the buffers of the nodes which are two-hops away from the sink; this can cause the nodes to drain their energy much faster than at a lower packet rate.

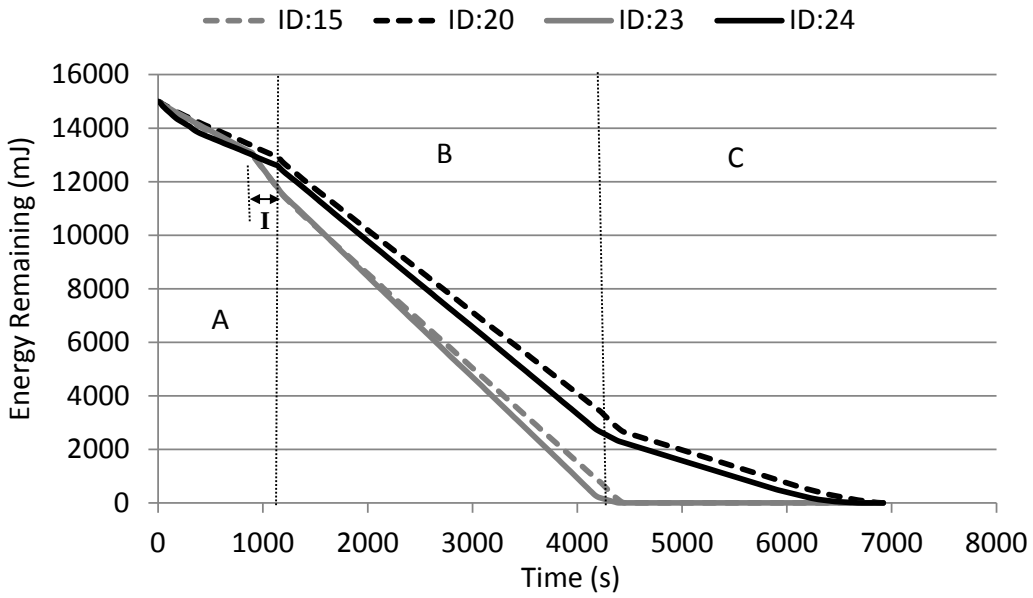


Figure 5.14: Remaining energy of node 15, 20, 23 and 24 to the sink over time for MUP (MUP-single) at 0.25 packet/second in a forest fire situation.

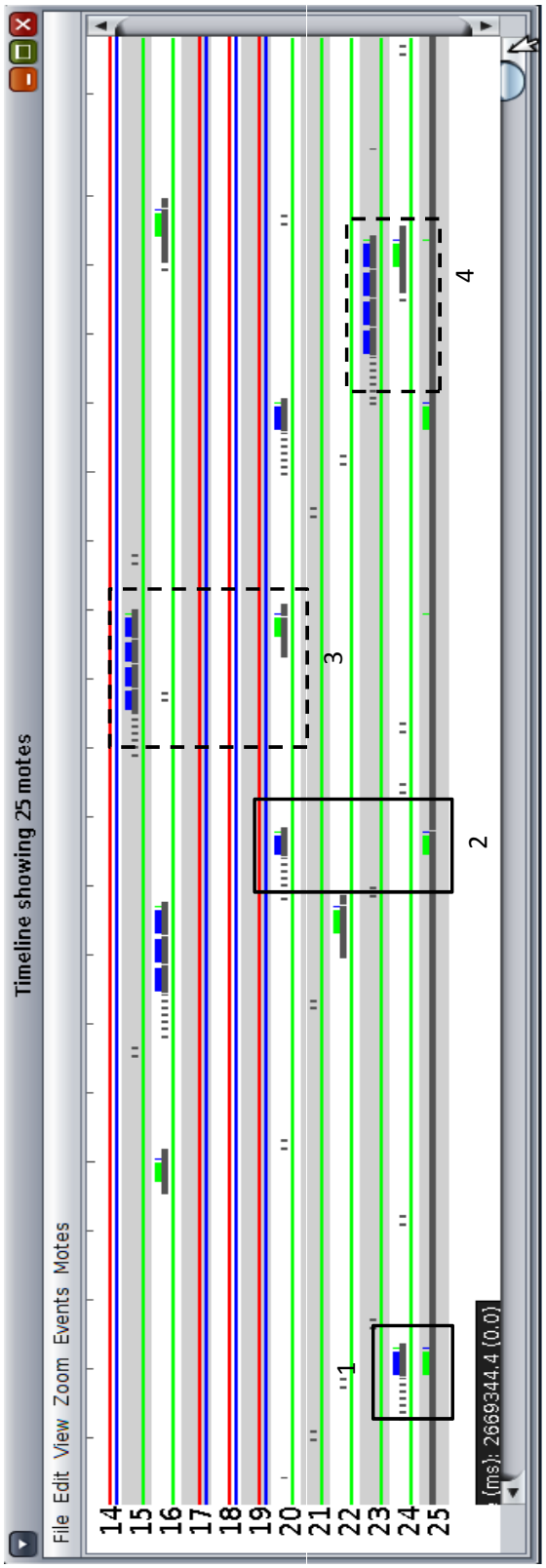


Figure 5.15: Captured timeline of radio activities for four safe nodes located closest to the sink.

5.3.2 Analysis of the Energy Consumption

Analysis of the energy consumption is divided into two parts according to node conditions: safe nodes and unsafe nodes. The first to be discussed is the energy consumption of the safe nodes for MUP, SAFEST and RPL over different packet rates in a forest fire situation. As illustrated in Figure 5.16, MUP provides a lower energy consumption of the safe nodes when compared to RPL. The difference in the energy consumption between both routing protocols is increased with the increase in the packet rate values. This is due to MUP being able to route more packets through the unsafe nodes, avoiding the safe nodes, at a higher packet rate than at a lower packet rate. At 1 packet/second, both implementations of MUP have achieved about 15% more energy saving of the safe nodes than RPL. In addition, SAFEST has recorded the highest average energy consumption of the safe nodes because the routing protocol routes packets through the safe nodes.

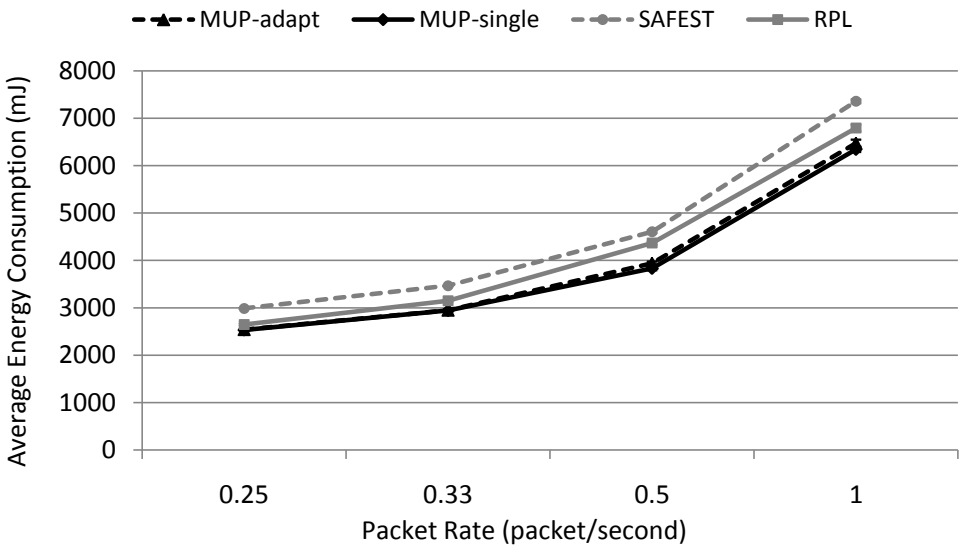


Figure 5.16: Average energy consumption for safe nodes over different packet rate values for MUP, SAFEST and RPL in a forest fire situation.

The second part of the energy consumption analysis is for the unsafe nodes. Figure 5.17 shows the energy consumption of the unsafe nodes for MUP, SAFEST and RPL over different packet rate values in a forest fire situation. As can be seen in the graph, MUP has recorded a higher energy consumption of the unsafe nodes than SAFEST and RPL for each packet rate. This is because MUP exploits the energy of the nodes before they get damaged in the fire. SAFEST has recorded the lowest average energy consumption for the unsafe nodes because the routing protocol routes packets away from the unsafe nodes.

We further the study by investigating the energy consumption for each of the unsafe and safe nodes. Figure 5.18 and Figure 5.19 show the average energy consumption for each of the unsafe nodes at different packet rates for MUP, SAFEST

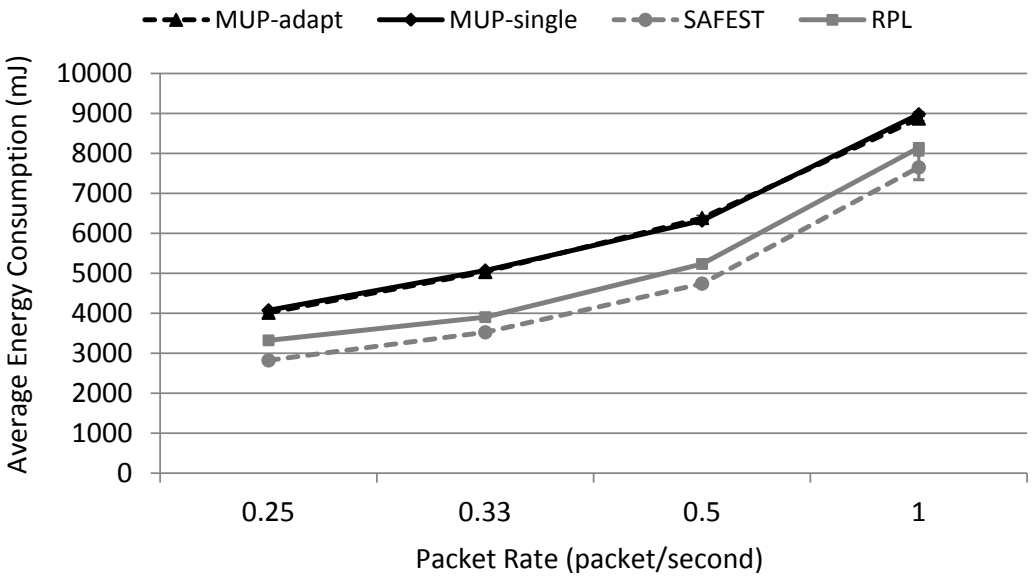
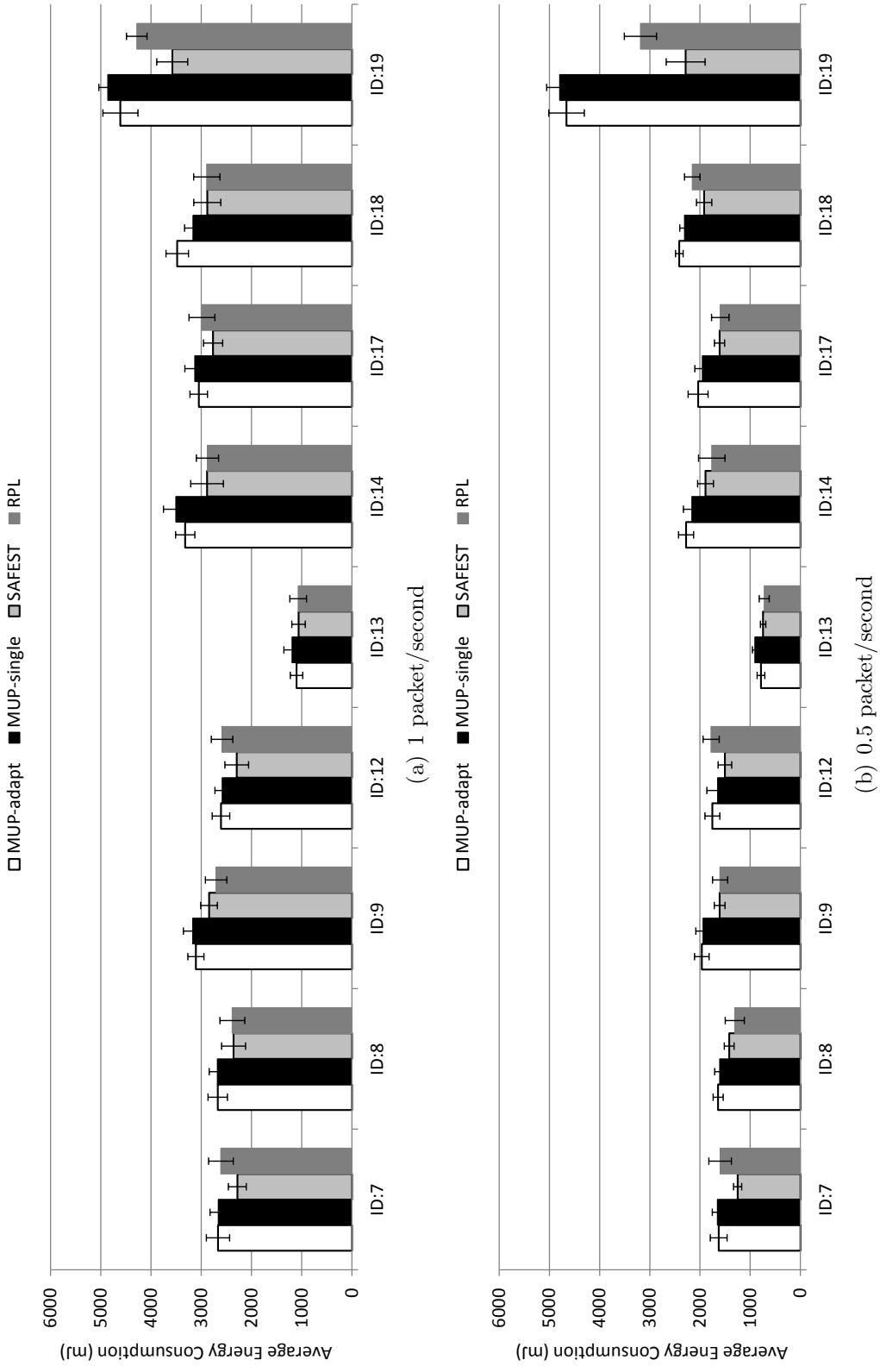


Figure 5.17: Average energy consumption for unsafe nodes over different packet rate values for MUP, SAFEST and RPL in a forest fire situation.

and RPL in a forest fire situation. As expected, MUP provides a higher energy consumption for each of the unsafe nodes when compared to SAFEST. This is because MUP aims to route packets through the unsafe nodes in order to exploit the energy of these nodes. By contrast, SAFEST aims to route packets avoiding the unsafe nodes causing the nodes to consume less of their energy. In the case of RPL, it is expected that the routing protocol should provide an energy consumption of an unsafe node between MUP and SAFEST. However, a certain node in the network may not meet the expectation. This is because RPL uses only the total path cost as a parameter in the routing decision without giving consideration to the node health status. Since a forest fire grows across the network, starting from the ignition point, the unsafe nodes do not detect the fire at the same time. For example, Figure 5.9(d) shows that node 8 becomes unsafe earlier than node 9. Consider an initial condition where a MUP node or an RPL node forwards packets through node 9. When node 8 becomes unsafe, the MUP node changes its routing path by selecting node 8 as the best parent, but an RPL node just remains with node 9, which is still in the safe condition. Before node 8 is burnt in the fire, the MUP node changes its best parent back to node 9, which will become unsafe when the fire reaches the node and it is then destroyed. In this condition, MUP has recorded a higher energy consumption of node 8 than RPL. Otherwise, MUP has recorded a lower energy consumption of node 9 than RPL, which does not meet the expectation.

Figure 5.20 and Figure 5.21 show the average energy consumption of the safe nodes for MUP, SAFEST and RPL. Overall, the results show that for the safe nodes located furthest away from the sink (located at the border of the network):

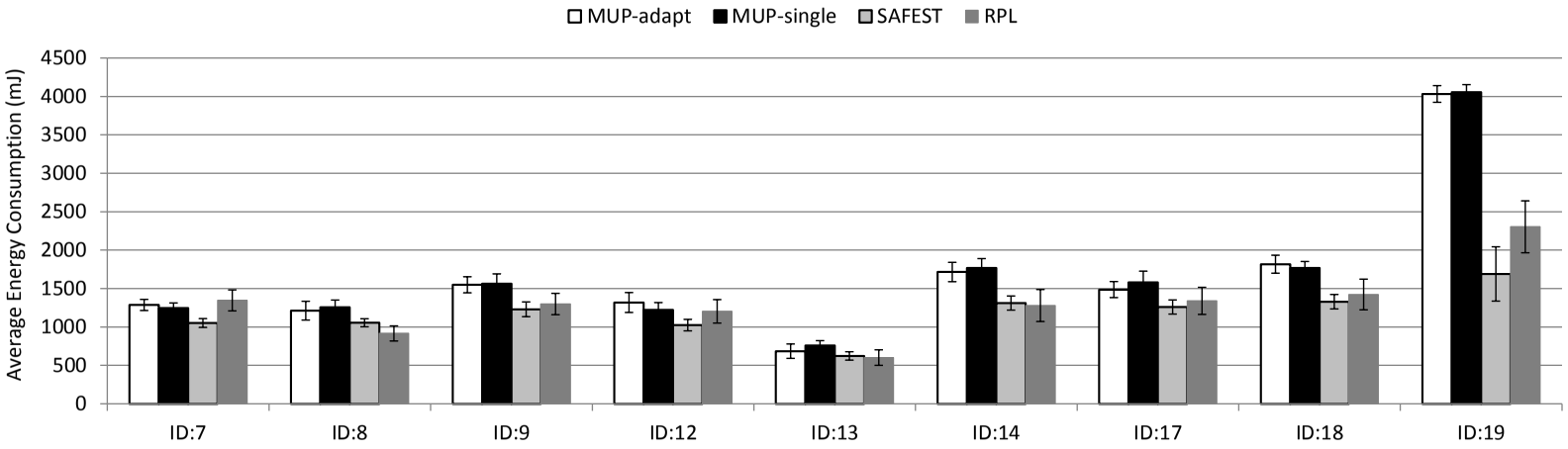
node 1, 2, 3, 4, 5, 6, 11, 16 and 21 are considered to have a similar energy consumption for all the routing protocols. This is because, most of the time, these nodes do not become intermediate nodes to support packet delivery to the sink. This means that the energy of these nodes is being used to send their individual data to the sink. As can be seen in the graphs, MUP provides a lower energy consumption for a safe node compared to SAFEST. This is because MUP avoids to sending packets through the safe nodes. Otherwise, SAFEST aims to route packets through the safe nodes causing these nodes to consume more energy. In the case of RPL, it is expected that the routing protocol should provide an energy consumption of a safe node between MUP and SAFEST. However, there are a few cases where RPL nodes do not meet this expectation. One of the factors is where the network only has a path to the sink through a safe node after a significant number of nodes have been burnt in the fire. MUP reconfigures the path through the safe node just before an unsafe node is destroyed in the fire. Instead of RPL, the routing protocol reconfigures the path only after several failures in the attempt to deliver packets to the destroyed nodes. This means that MUP routes packets through the safe node earlier than RPL, which can cause the energy consumption of the critical node for MUP to be much higher when compared to RPL. Other than that, RPL does not consider the node health status in the routing decision, which may also cause a safe node not to meet the expectation.



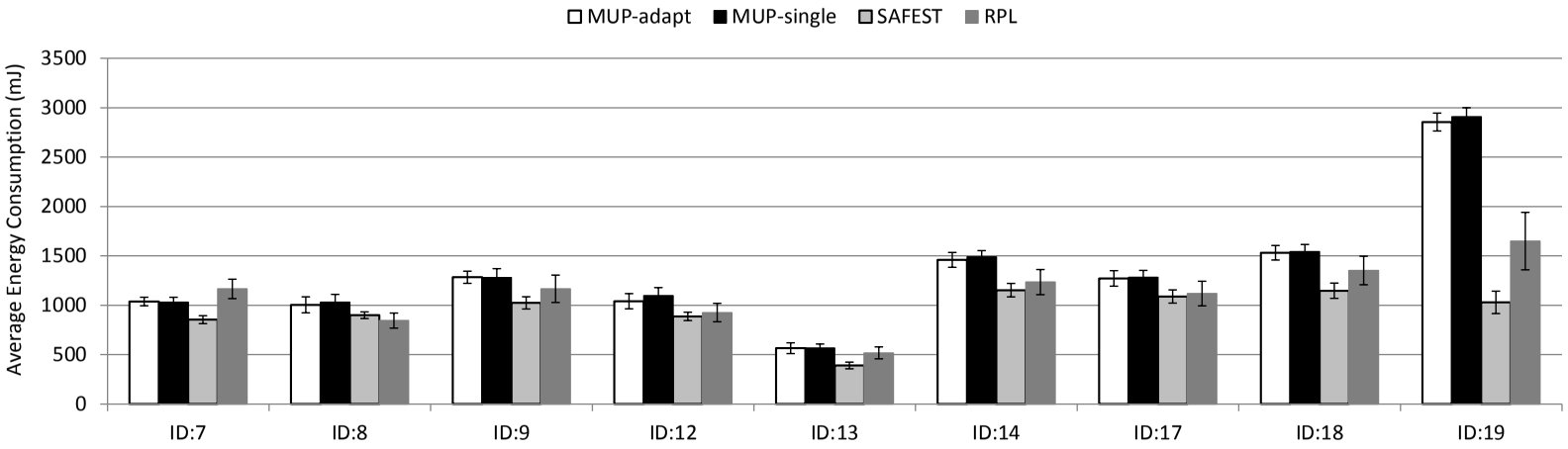
(a) 1 packet/second

(b) 0.5 packet/second

Figure 5.18: Average energy consumption for each unsafe node at 1 and 0.5 packet/second for MUP, SAFEST and RPL in a forest fire situation.



(a) 0.33 packet/second



(b) 0.25 packet/second

Figure 5.19: Average energy consumption for each unsafe node at 0.33 and 0.25 packet/second for MUP, SAFEST and RPL in a forest fire situation.

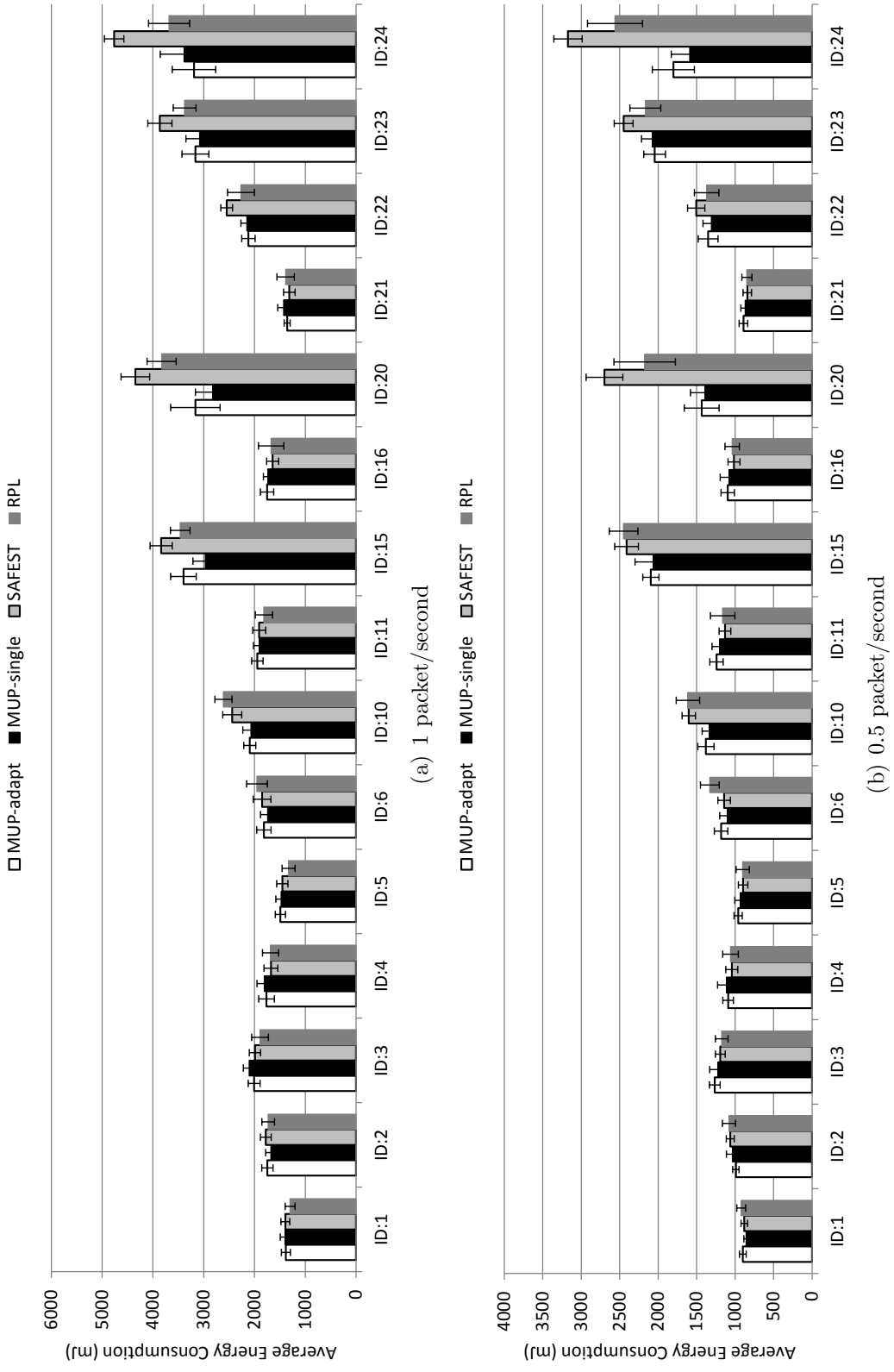
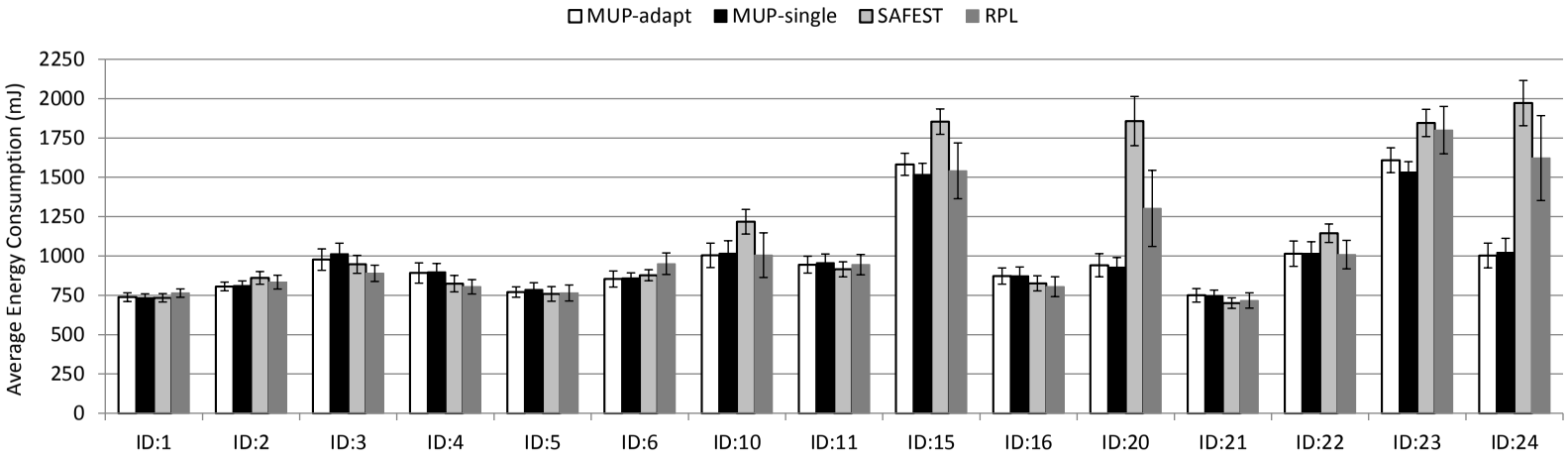
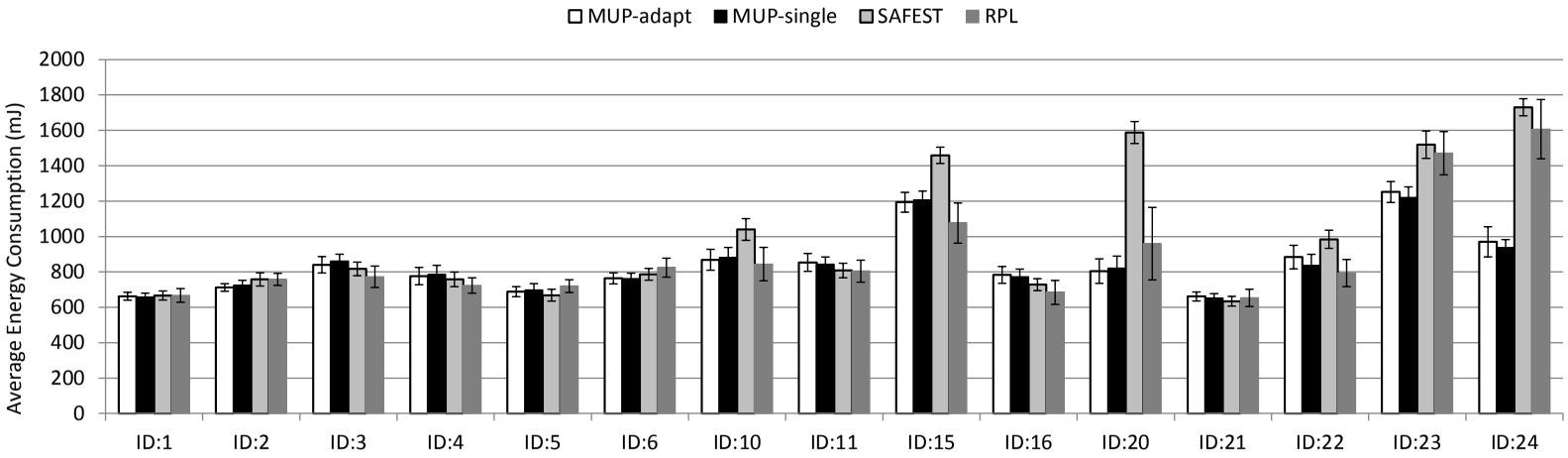


Figure 5.20: Average energy consumption for each safe node at 1 and 0.5 packet/second for MUP, SAFEST and RPL in a forest fire situation.



(a) 0.33 packet/second



(b) 0.25 packet/second

Figure 5.21: Average energy consumption for each safe node at 0.33 and 0.25 packet/second for MUP, SAFEST and RPL in a forest fire situation.

5.3.3 Packet Delivery Ratio

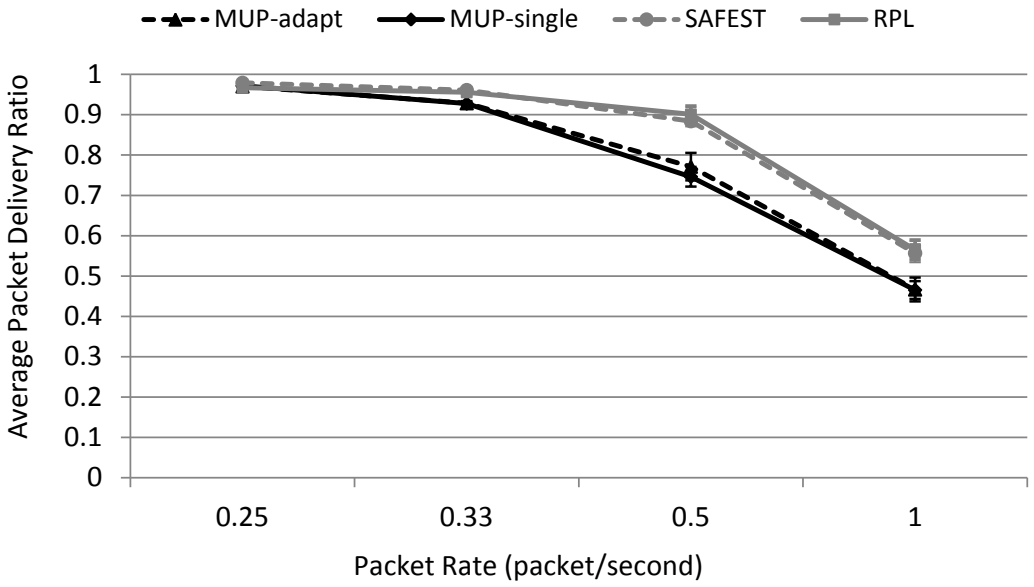


Figure 5.22: Average packet delivery ratio over different packet rate values for MUP, SAFEST and RPL in a forest fire situation.

Figure 5.22 shows that MUP provides a lower packet delivery ratio than RPL, regardless of the packet rate value in a forest fire situation. This is because MUP suffers from more packets being dropped than RPL due to the concentration of packets through the unsafe nodes during the critical phase. Both implementations of MUP have recorded the average packet delivery ratio over the different packet rates to be about 6% lower than SAFEST and RPL.

Figures 5.23 to 5.26 show the cumulative density function of the number of collected packets at the sink for different packet rates for MUP, SAFEST and RPL in a forest fire situation. As can be seen in the graphs, MUP provides a higher number of collected packets at the sink when compared to SAFEST and RPL regardless of packet rate, even though MUP suffers a poor packet delivery ratio performance during the fire. This is achieved because MUP prolongs the network lifetime which allows the routing protocol to collect more packets from the network for a longer period of time. Three horizontal lines are drawn in each graph to indicate the network lifetime for each routing protocol. These lines are drawn at the point where the response is just becoming flat, because the sink does not receive any more packets from the network, and it can be considered that the network has become non-functional. The horizontal lines are numbered with 1, 2 and 3, which belong to SAFEST, RPL and MUP respectively. The results show that MUP achieves a longer network lifetime when compared to MUP and RPL for all packet rates. SAFEST has recorded the shortest network lifetime when compared to the other routing protocols.

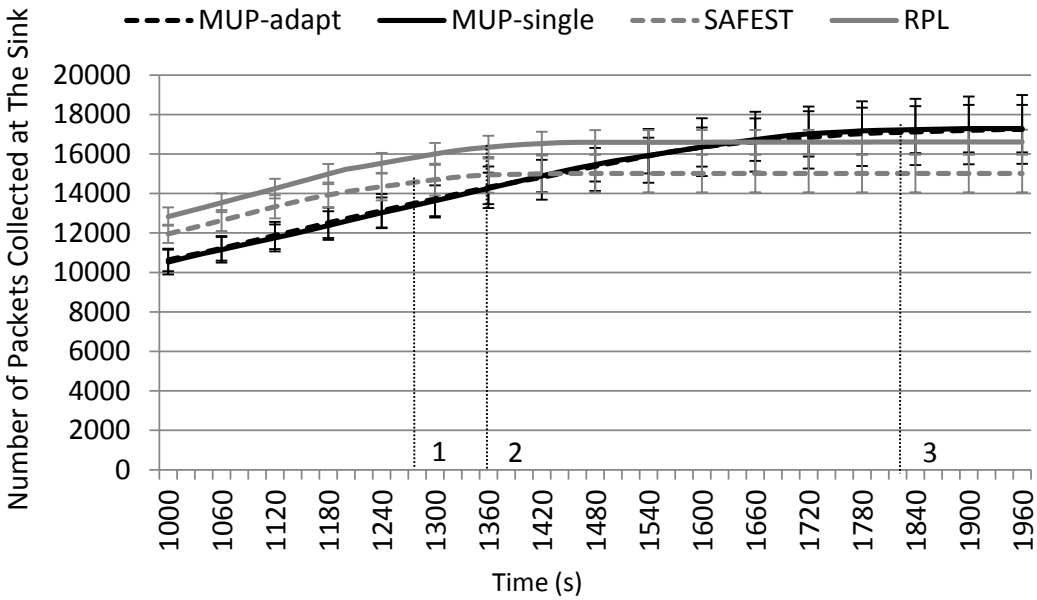


Figure 5.23: Total number of packets collected at the sink at 1 packet/second for MUP, SAFEST and RPL in a forest fire situation.

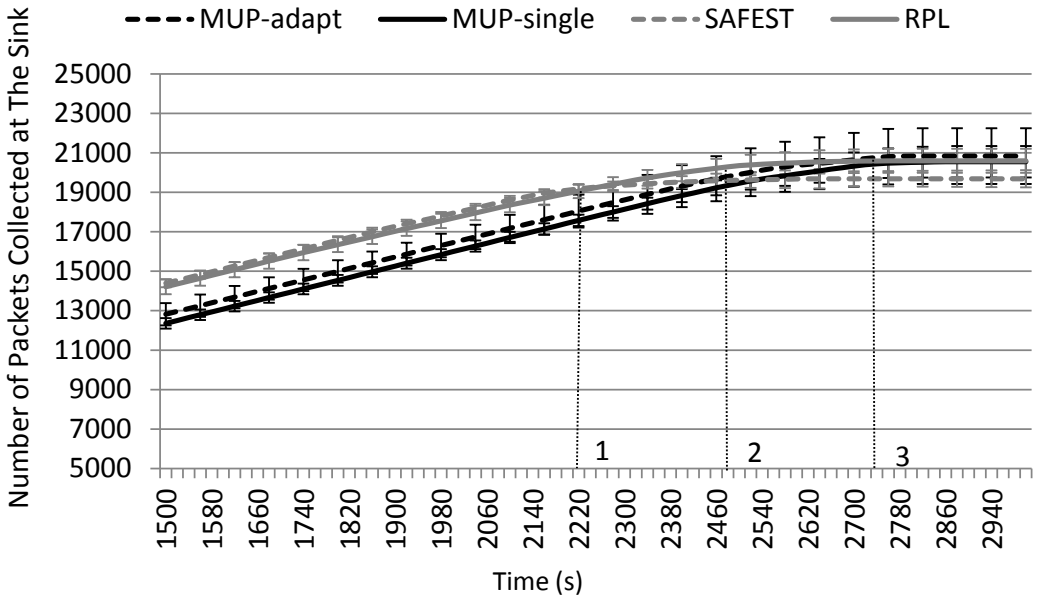


Figure 5.24: Total number of packets collected at the sink at 0.5 packet/second for MUP, SAFEST and RPL in a forest fire situation.

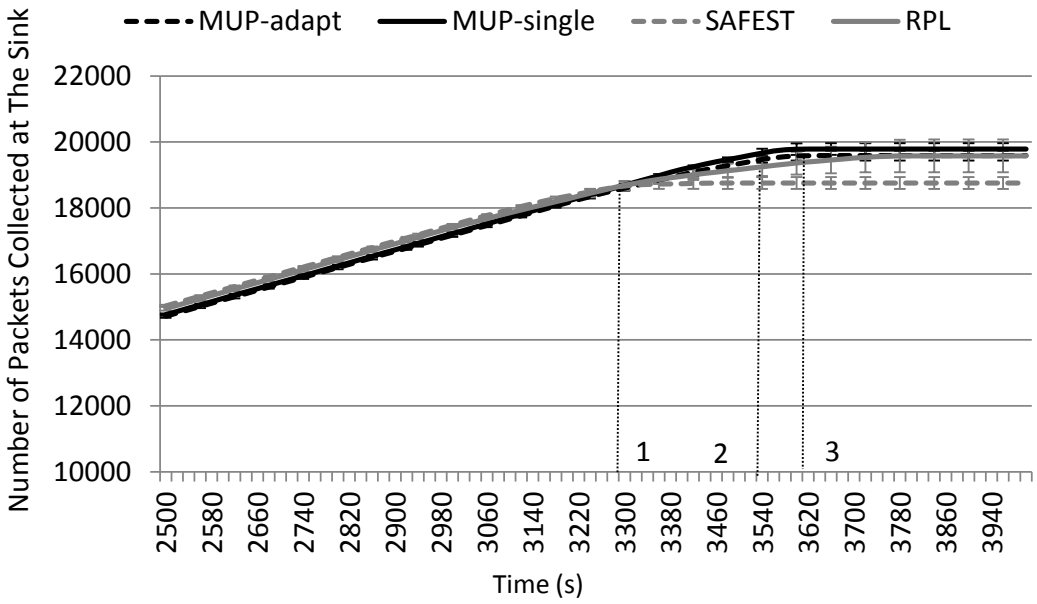


Figure 5.25: Total number of packets collected at the sink at 0.33 packet/second for MUP, SAFEST and RPL in a forest fire situation.

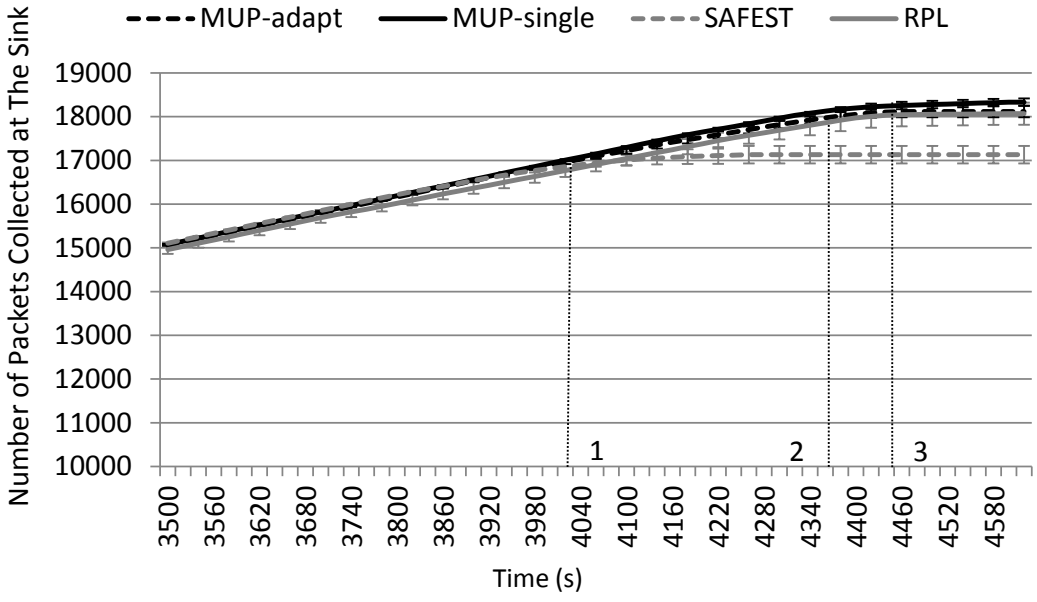


Figure 5.26: Total number of packets collected at the sink at 0.25 packet/second for MUP, SAFEST and RPL in a forest fire situation.

5.3.4 End-to-End Delay

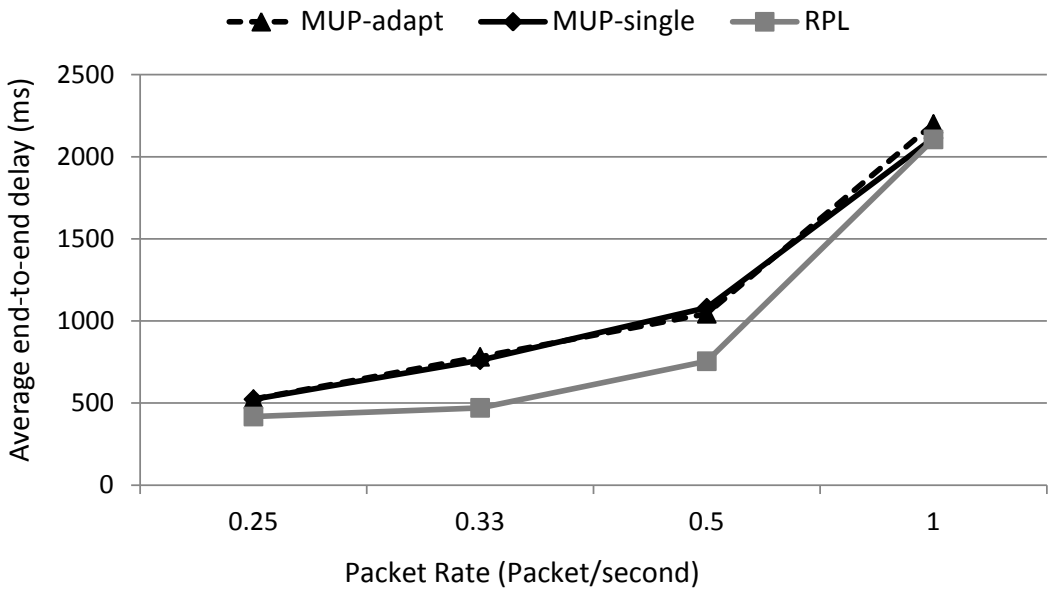


Figure 5.27: Average end-to-end delay over different packet rate values for MUP and RPL in a forest fire situation.

Figure 5.27 shows end-to-end delay performance over different packet rates in a forest fire situation. The graph shows that MUP suffers a higher end-to-end delay when compared to RPL. By taking the average value over the packet rates, both implementations of MUP have experienced about 20% higher end-to-end delay than RPL.

5.3.5 Analysis of the Average Number of Packet Transmission and Reception

Figure 5.28 shows the average number of packet transmission activity for unsafe nodes over different packet rates in a forest fire situation. As can be seen in the figure, MUP has recorded a higher number of packet transmission activity for unsafe nodes when compared to RPL. In MUP, the unsafe nodes need to handle more packets than other nodes in the network, which has caused the increment in the unsafe node's activities of transmitting and receiving packets. Figure 5.29 shows that MUP has recorded the average number of packet reception activity for the unsafe nodes higher than RPL. Otherwise, MUP provides a lower number of packet transmission and reception for the safe nodes when compared to RPL, as shown in Figure 5.30 and Figure 5.31 respectively. This is because, MUP avoids routing packets through the safe nodes in order to save the node energy. This means that the safe nodes handle less packets compared to other nodes in the network, which reduces the safe node's activity in transmitting and receiving

packets.

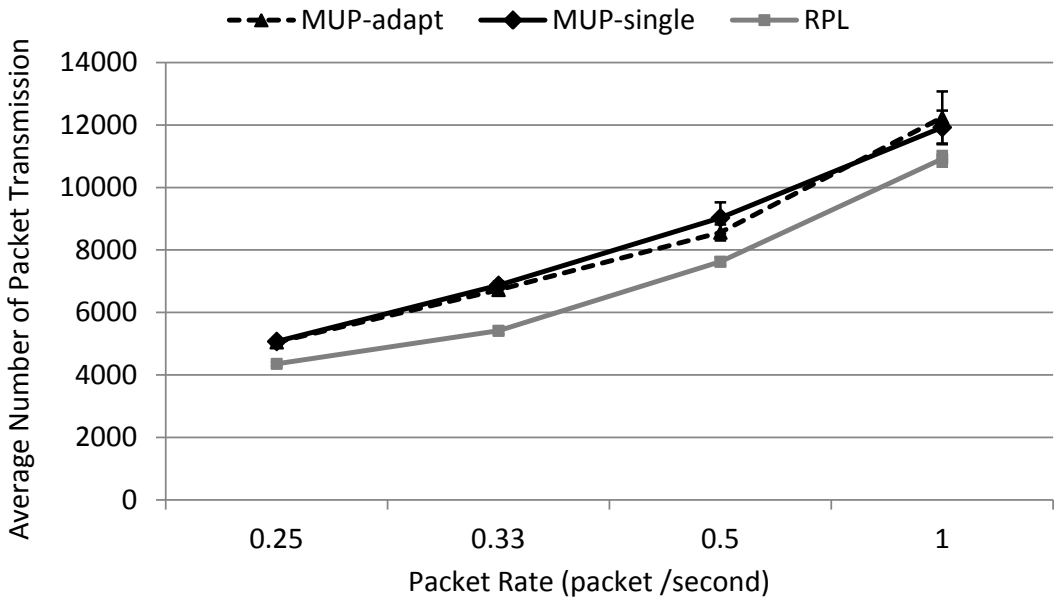


Figure 5.28: The average number of packet transmission for unsafe nodes for MUP and RPL in a forest fire situation.

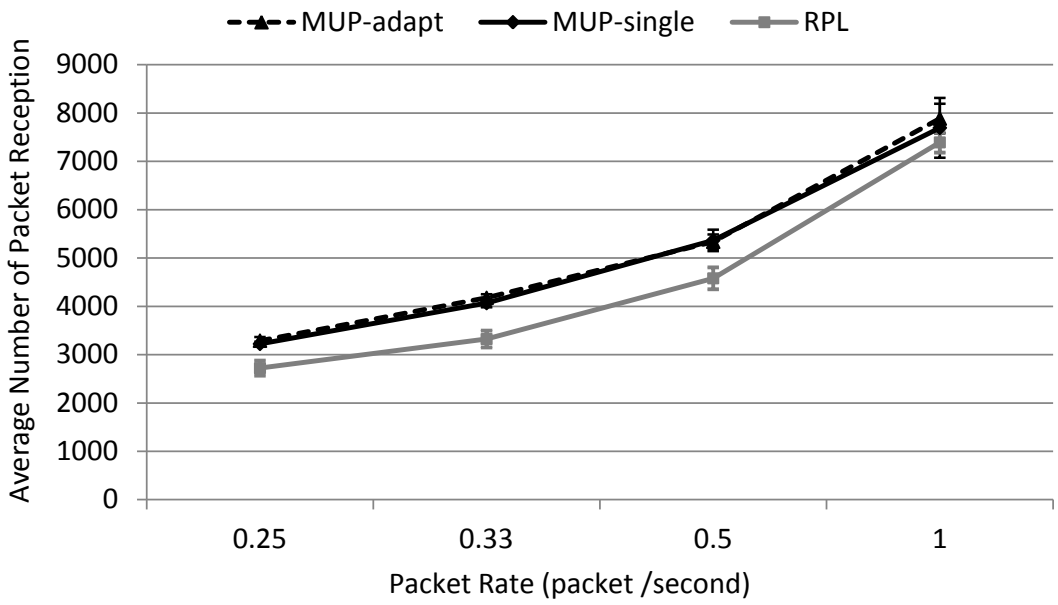


Figure 5.29: The average number of packet reception for unsafe nodes for MUP and RPL in a forest fire situation.

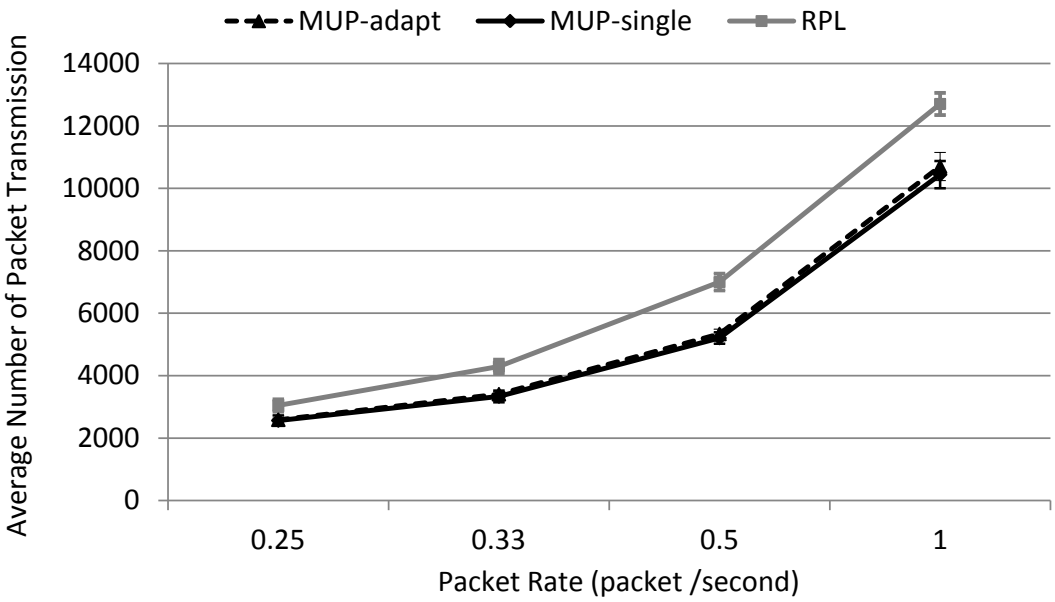


Figure 5.30: The average number of packet transmission for safe nodes for MUP and RPL in a forest fire situation.

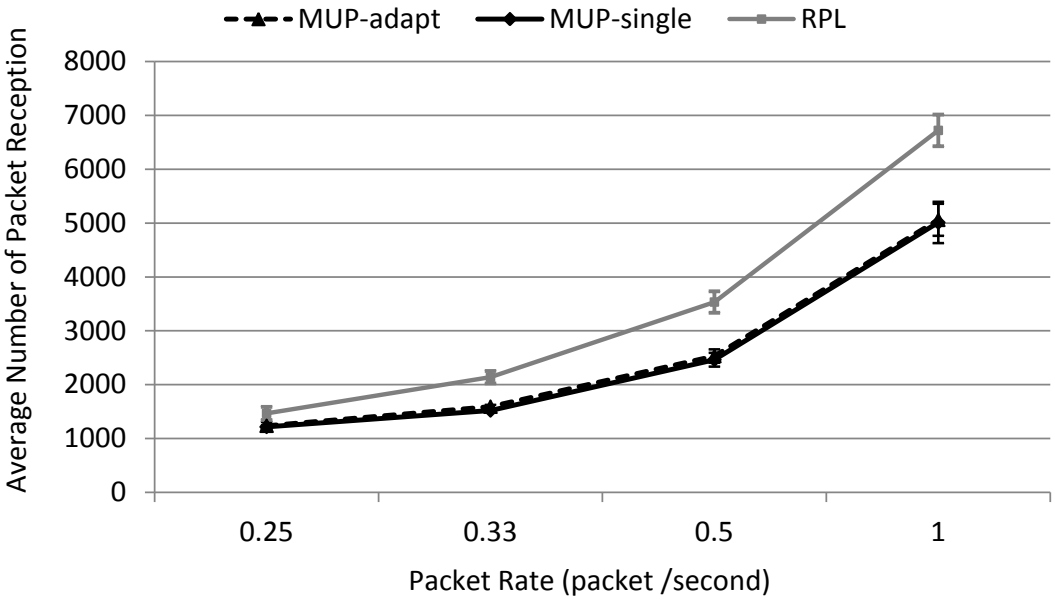


Figure 5.31: The average number of packet reception for safe nodes for MUP and RPL in a forest fire situation.

5.3.6 Analysis of Number of Packets Dropped due to Congestion

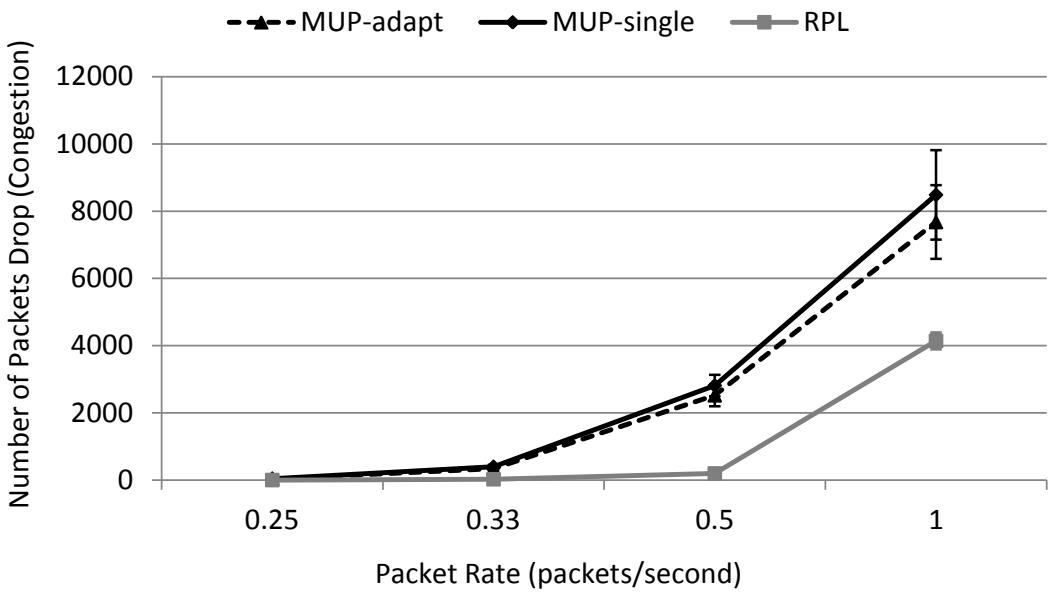


Figure 5.32: Total number of packets dropped due to congestion at unsafe nodes over different packet rates for MUP and RPL in a forest fire situation.

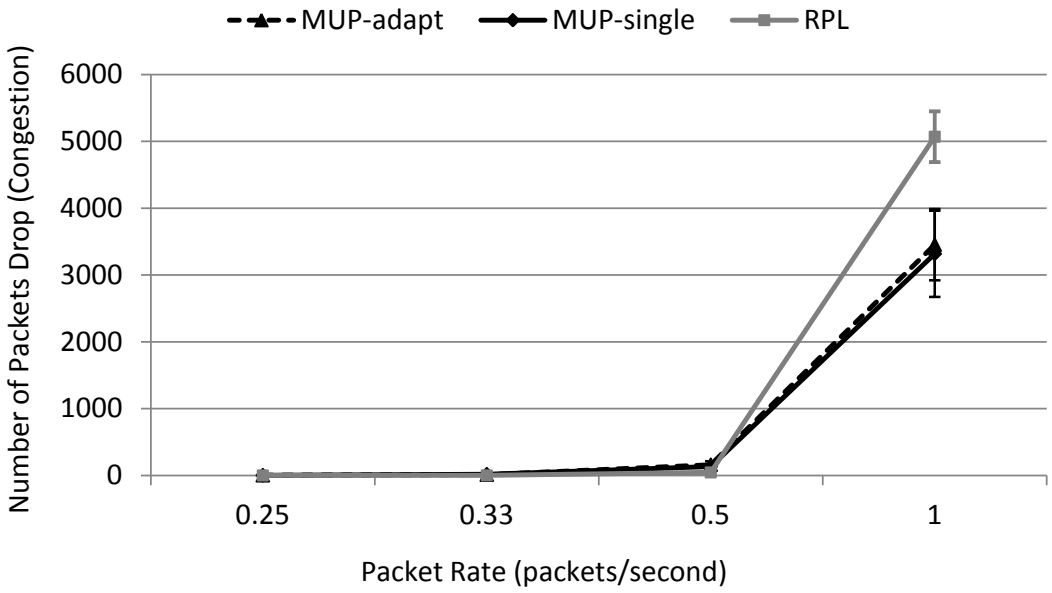


Figure 5.33: Total number of packets dropped due to congestion at safe nodes over different packet rates for MUP and RPL in a forest fire situation.

Figure 5.32 shows the total number of packets dropped due to congestion at unsafe nodes for MUP and RPL in a forest fire situation. As can be observed in the graph, MUP suffers a higher number of packets dropped due to congestion at unsafe nodes when compared to RPL. This is because congestion is likely to

happen at the unsafe nodes when the nodes are not capable of serving high incoming packets from neighbouring nodes during the critical phase. The unsafe nodes simply drop the incoming packets if their buffers are full. Otherwise, MUP provides a lower number of packets dropped due to the congestion at the safe nodes than RPL, as shown in Figure 5.33. This is because MUP always routes packets by avoiding the safe nodes, causing less packets to go through these nodes, which reduces the possibility of congestion. At 1 packet/second, MUP has achieved about 35% lower number of packets dropped due to congestion at the safe nodes than RPL.

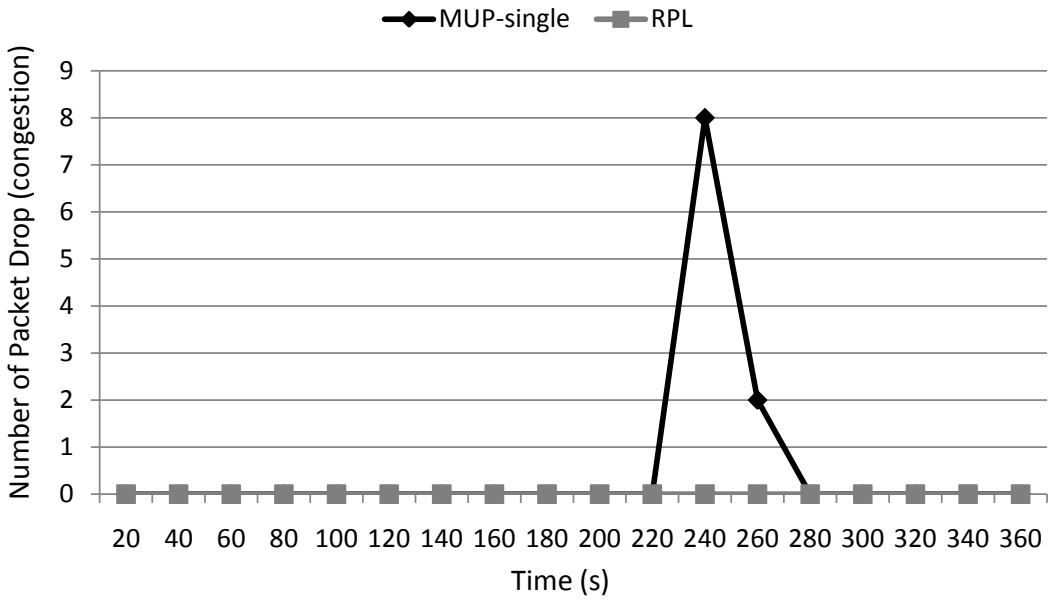


Figure 5.34: Packet drop over time due to congestion at node 13 for MUP (MUP-single) and RPL in a forest fire situation.

Figure 5.34 shows the number of packets dropped due to congestion at node 13 over time for MUP (MUP-single) and RPL at 0.25 packet/second in a forest fire scenario. The total number of packets dropped is calculated for every 20 s interval. Node 13 is located at the centre of the network as described in subchapter 4.6. This is the first node to detect the fire, and after that is burnt in the fire. MUP starts to drop packets due to congestion just after the node changes its health status to UNSAFE when detecting the fire breaking out in the network. After 280 s, node 13 does not experience any packet drop because no packets are forwarded through it after the node changes its health status to ALMOST-FAILED. All neighbouring nodes remove node 13 as their parent because the node is going to failed soon to avoid packet loss when the node is damaged. For RPL, the routing protocol has performed without any packet drops, which means that there is no congestion happening at node 13.

Node 19 is taken as another example of an unsafe node, which is located closest

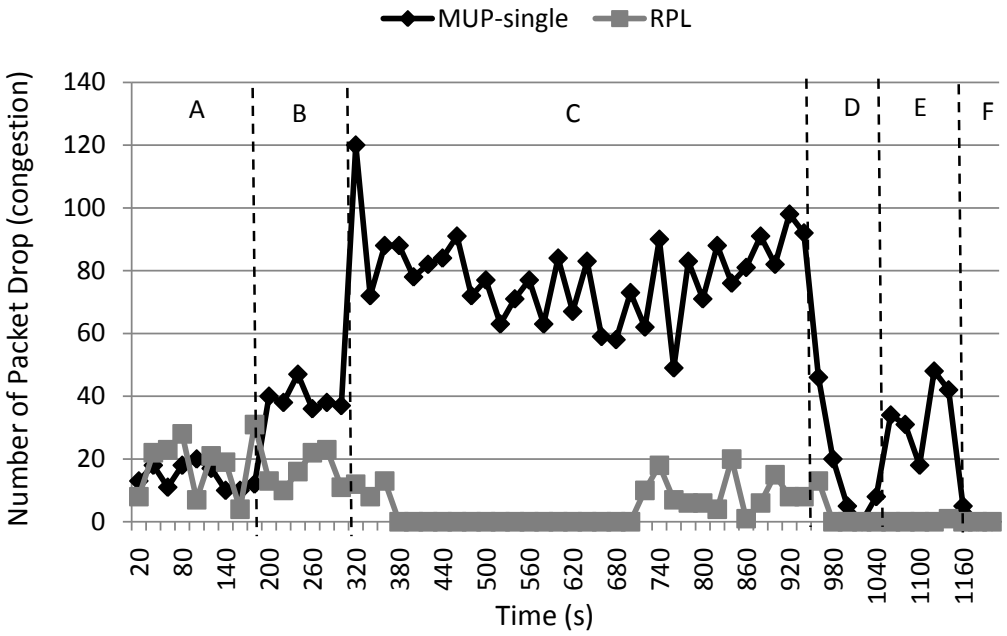


Figure 5.35: Packet drop over time due to congestion at node 19 for MUP (MUP-single) and RPL in a forest fire situation.

to the sink. Figure 5.35 shows the number of packets dropped due to congestion at node 19 over time for MUP (MUP-single) and RPL in a forest fire situation. In order to analyse the graph, it has been divided into five sections based on the MUP response. These sections are labelled as area A, B, C, D and E. Detailed analysis for every section is described below:

Section A: Before the fire starts

MUP provides a similar number of packets dropped due to congestion as RPL.

Section B: Node 13 becomes unsafe

MUP has recorded about twice the number of packets dropped when compared to RPL. This is because node 19 needs to support a higher number of incoming packets from node 13 (which is one of its child nodes) when the node becomes unsafe after it detects that fire has started to occur in the network.

Section C: Node 19 changes its status to lowsafe

The number of packets dropped at node 19 for MUP is increased from the previous section. This is due to node 19 having changed its health status from SAFE to LOWSAFE after receiving a DIO message from node 13, which has an ALMOST-FAILED status indicating the node is going to fail soon. At this stage, it is expected that node 19 needs to handle most of the packets

in the network because the location of the node is one-hop away from the sink.

Section D: The other two nearest nodes to the sink become lowsafe.
The number of packets dropped recorded by MUP is reduced significantly to almost nothing, which is similar to RPL. This is due to a reduction in the number of packets going through node 19 when its child nodes select the other nearest nodes to the sink (node 20 and 24), which have just changed their health status from SAFE to LOWSAFE.

Section E: Node 19 becomes an unsafe node
MUP suffers another increase in the number of packets dropped at node 19. This is caused by the increasing number of packets going through node 19 when the node becomes unsafe after it detects fire. The node changes its health status from LOWSAFE to UNSAFE. RPL provides no packets dropped due to congestion at node 19.

Section F: Node 19 changes its health status to almost-failed
MUP provides almost no packets dropped due to congestion at node 19, which is similar performance to RPL. This is because node 19 is almost damaged in the fire and changes its health status to ALMOST-FAILED. All its child nodes remove node 19 as their parent causing no packets going through the node.

Overall, MUP suffers a higher number of packets dropped due to congestion at the unsafe nodes than RPL during the critical phase. This congestion occurs when the unsafe nodes are not capable of handling the high incoming packets from their neighbours. In the normal phase, MUP provides similar performance to RPL.

5.3.7 Analysis of Number of Packets Dropped due to Exceeding The Maximum Retransmission

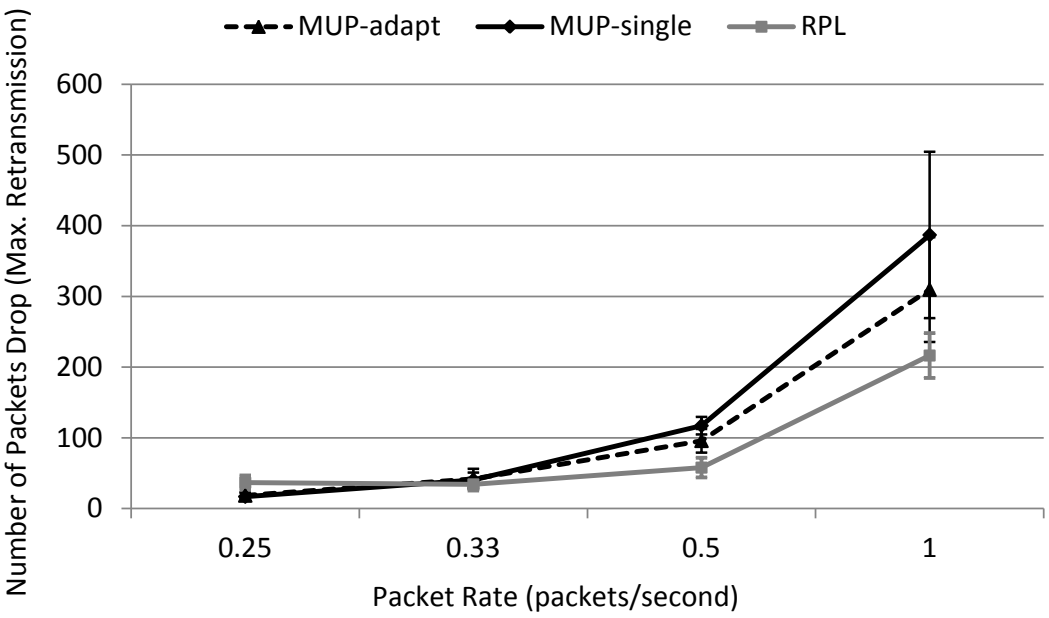


Figure 5.36: Total number of packets dropped due to exceeding the maximum retransmission at unsafe nodes over different packet rates for MUP and RPL in a forest fire situation.

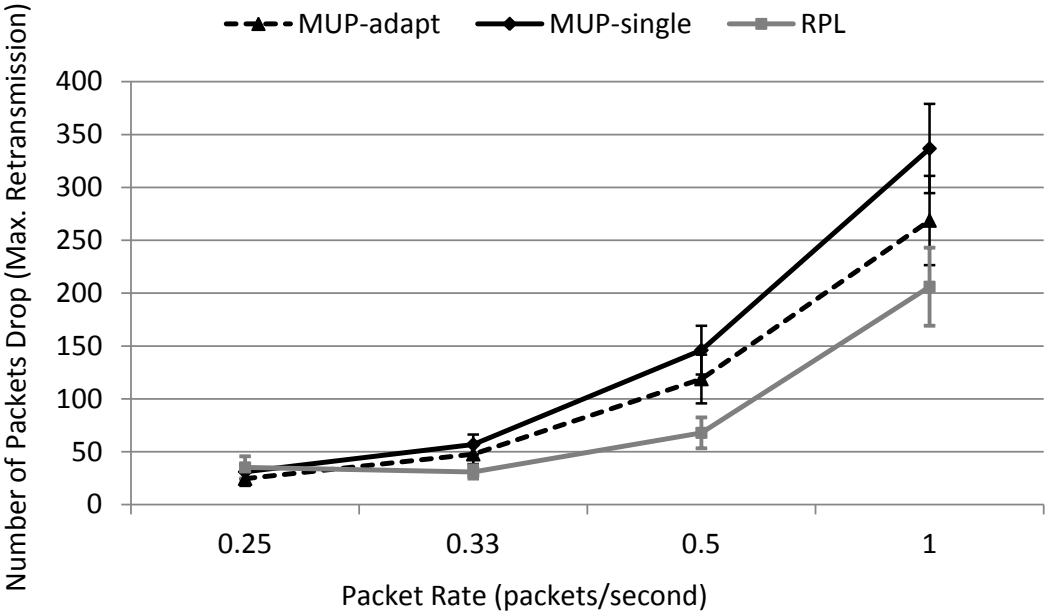


Figure 5.37: Total number of packets dropped due to exceeding the maximum retransmission at safe nodes over different packet rates for MUP and RPL in a forest fire situation.

In terms of the number of packets dropped due to exceeding the maximum retransmission, it is found that MUP suffers more packet drop than RPL. This

poor performance is observed to happen at both unsafe nodes and safe nodes as shown in Figure 5.36 and Figure 5.37 respectively. The main reason for this is that MUP experiences more radio interference when the child nodes need to adjust their transmission time according to the active period of the unsafe node (which is their parent node). This transmission time adjustment is very important because the child nodes must know when the unsafe node wakes-up from its idle state to the listening state. The child nodes must send packets only during the listening state of the unsafe node in order to achieve successful packet delivery. In this situation, there is a high probability of two or more child nodes transmitting packets simultaneously to the unsafe parent which can cause radio interference.

Figure 5.38 shows a captured timeline for MUP (MUP-single) that indicates the transmitting and receiving activities for all nodes during the period of time when node 13 is in the unsafe condition during the forest fire situation. The occasion of radio interferences are labelled clearly with dotted line boxes on the captured timeline. Label 1 and label 3 show radio interferences occurring on node 13 when its two child nodes transmit packets simultaneously. Label 2 shows that an interference could also happen when one of the child nodes and another node within the same coverage area of node 13 transmit packets simultaneously.

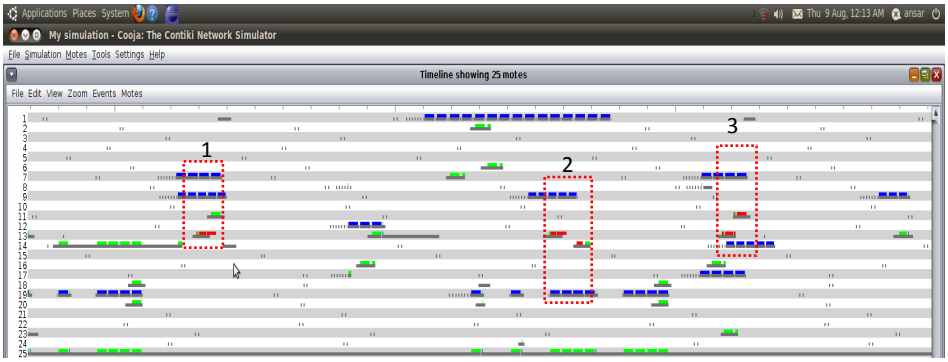


Figure 5.38: Captured timeline of radio interference during the period of time in which node 13 is in an UNSAFE condition for MUP (MUP-single).

Label 1:

Interference happens on node 13 caused by its own child nodes, which are node 7 and node 9.

Label 2:

Interference happens on node 13 caused by one of its child nodes (node 9) and another node (node 19) in the network.

Label 3:

Interference happens on node 13 caused by its own child nodes, which are node 7 and node 14.

5.3.8 Analysis of Generated and Received Packets over Time

Figure 5.39 shows the total number of generated and received packets against time at a packet rate of 0.5 packet/second in a forest fire situation. The graph is plotted by calculating the total number of generated and received packets for every second. Since each node is configured to have a packet rate of 0.5 packet/second, the total number of generated packets in the network during the initial stage of the simulation is equal to 12 packet/second. As can be seen in the graph, the total number of generated packets in the network is similar for MUP and RPL. There are drops in the total number of generated packets occurring at points A, B and C. This is because at these points a few nodes are just burnt in the fire causing a reduction in the number of functioning nodes. In terms of received packets at the sink, the graph shows that MUP provides a lower number of received packets than RPL during the forest fire phase. This is because MUP experiences a higher number of packets lost when the routing protocol focuses its routes through the unsafe nodes.

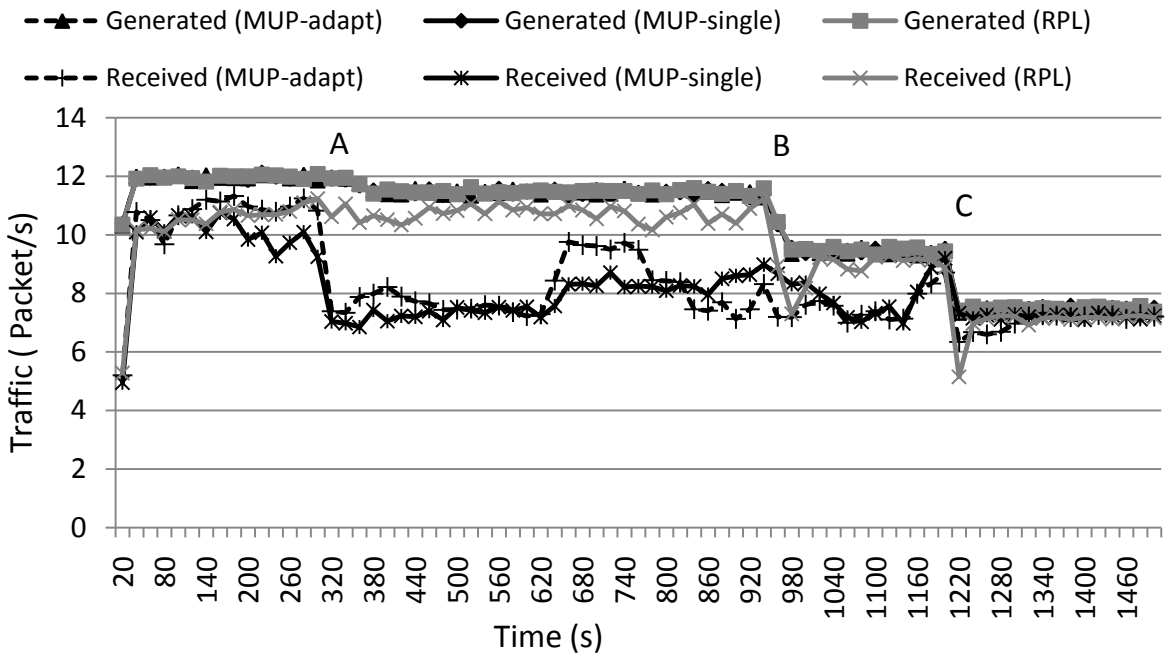


Figure 5.39: Generated and received packets in the network for MUP and RPL at the initial packet rate of 0.5 packet/second in a forest fire situation.

Figure 5.40 shows the total number of generated and received packets against time for MUP and RPL at packet rate 0.33 packet/second in a forest fire situation. The total number of generated packets is almost the same for MUP and RPL, which is the same performance trend observed at packet rate 0.5 packet/second. The number of generated packets is reduced at points A, B and C because of the

reduction in the number of functioning nodes after a few nodes are burnt in the fire. In terms of the number of received packets at the sink, the graph shows that MUP provides a slightly lower number of received packets than RPL. This MUP performance is better when compared to the routing protocol performance observed at packet rate 0.5 packet/second. This improvement is achieved because less packets has entered the network at a lower packet rate than at a higher packet rate, which reduces the number of packets lost in the network.

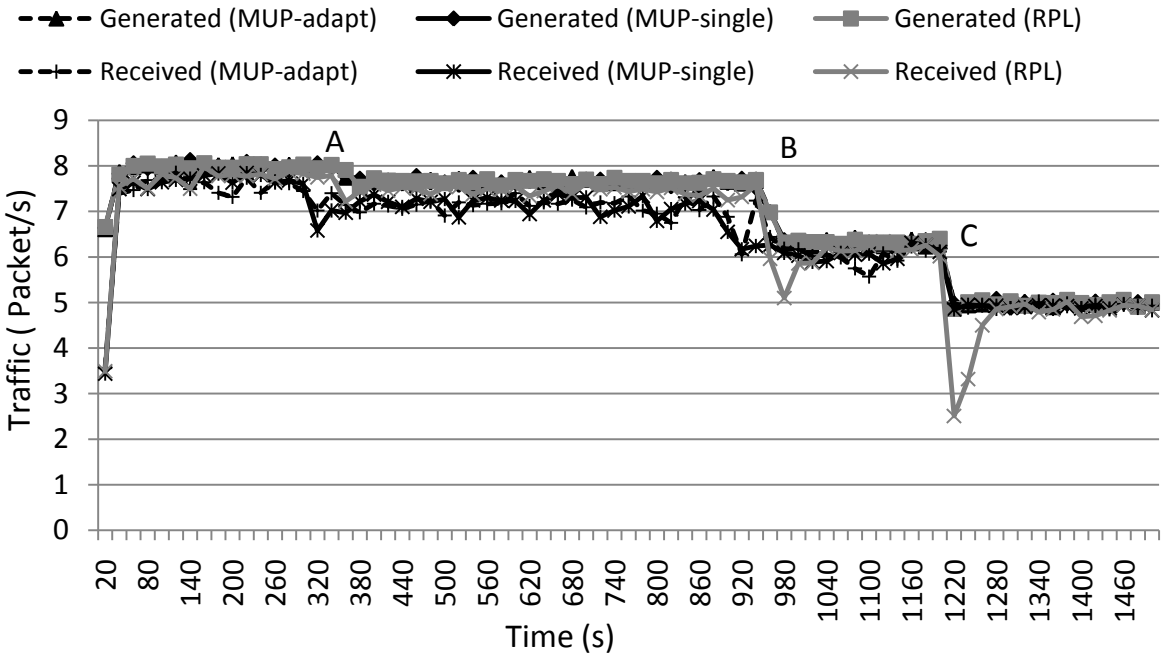


Figure 5.40: Generated and received packets in the network for MUP and RPL at the initial packet rate of 0.33 packet/second in a forest fire situation.

RPL suffers a sudden drop in the number of received packets at points B and C, which is lower than MUP performance. This performance drop occurs because RPL nodes are unable to avoid any node that is almost damaged because the routing protocol does not consider node condition in its routing decision. When a node becomes damaged, its child nodes simply keep sending packets to the damaged node. This situation can cause packet loss until an alternative route is configured avoiding the damaged node. Otherwise, a MUP node has the capability to avoid the almost damaged node. The MUP node removes the almost damaged node as a parent candidate after receiving a DIO message from the node.

5.3.9 Analysis of the Number of Update Routing Messages

Figure 5.41 shows the number of DIO messages sent for each node in the network for MUP and RPL in a forest fire situation. The graph shows that MUP has

recorded a higher number of DIO messages sent when compared to RPL, regardless of the packet rate. This is because MUP introduces an additional inconsistency, which is the change of node health status. Every time a MUP node changes its health status, the node needs to inform all its neighbouring nodes about its current health status by sending DIO messages so that the routes can be adjusted according to the OF. Since a change of parent node is one of the inconsistencies, a neighbour node must reset its trickle timer if it has changed its parent to inform other nodes about its current routing cost through the new parent node. This situation can cause a further increment in the number of DIO messages sent by a MUP node. A MUP node sends an average number of DIO messages which is about 9 messages more than RPL.

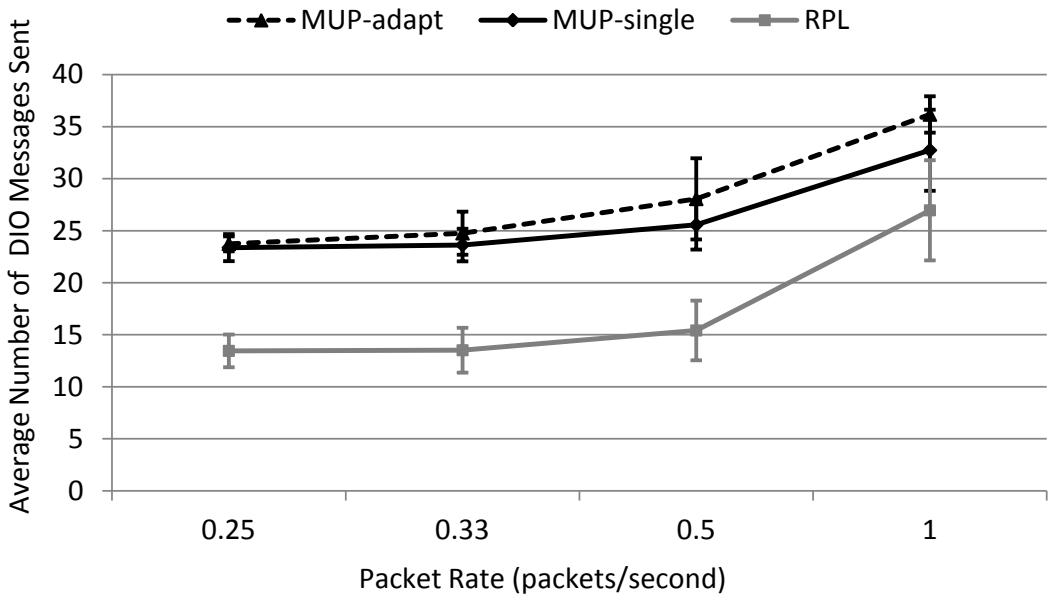


Figure 5.41: Average number of DIO messages sent per node over different packet rates for MUP and RPL in a forest fire situation.

Figure 5.42 shows that MUP provides a higher number of DAO messages sent than RPL. This is because MUP experiences more changes of DAO parent than RPL in the process of focusing routes through the unsafe nodes during the critical phase. Since the DAO parent is the same as the DODAG parent for upward traffic, any changes of the DODAG parent causes a changes in the node's DAO parent automatically. When a node changes its DAO parent, the node needs to send a no-path DAO message to the previous parent about the node unreachability by setting the path lifetime to zero. Furthermore, the node needs to inform its reachability through the new DAO parent by sending another DAO message. This is the reason why MUP has recorded an average number of DAO messages sent which is about 7 messages more than RPL.

Figure 5.43 shows the trickle timer response of node 13 for MUP during the

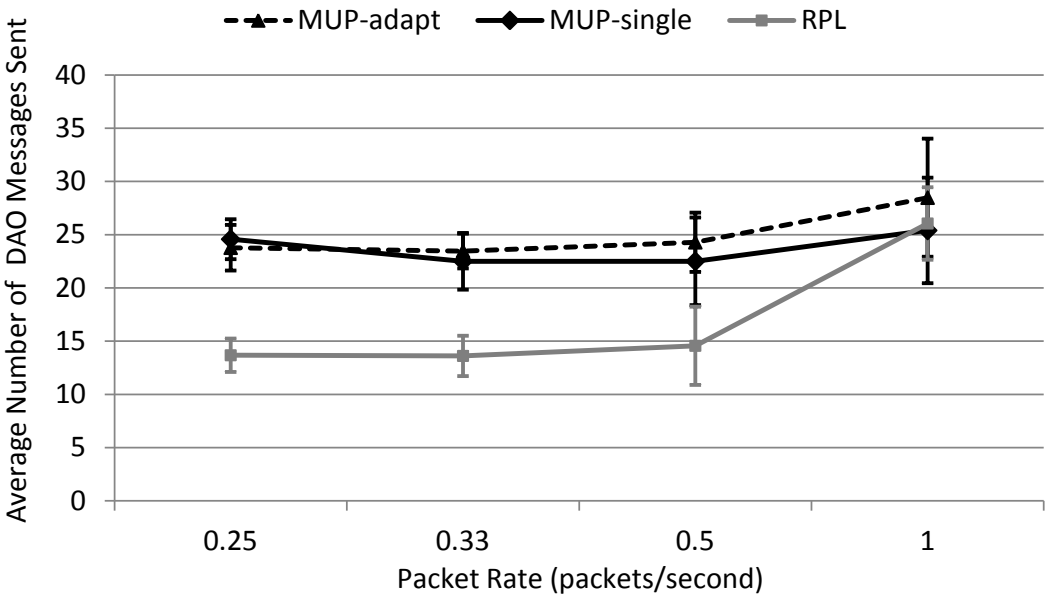


Figure 5.42: Average number of DAO messages sent per node over different packet rates for MUP and RPL in a forest fire situation.

forest fire situation. As can be seen on the graph, there are two occasions where the node resets the trickle timer to the minimum interval due to the changes of node health status, which are labelled as 'A' and 'B'. At point 'A', the node changes its health status from SAFE to UNSAFE after detected fire breaks out at the centre of the network. At point 'B', the node becomes almost failed and changes its health status from UNSAFE to ALMOST-FAILED.

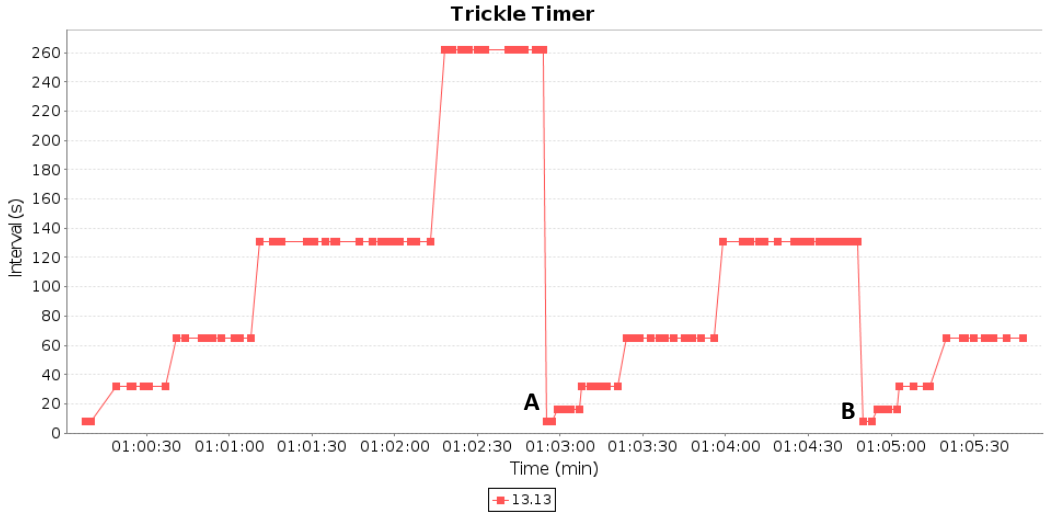


Figure 5.43: MUP (MUP-single) trickle timer response for node 13 in a forest fire situation.

Figure 5.44 shows the trickle timer response of node 19 for MUP during the forest fire situation. There are three occasions where the node resets the trickle timer to the minimum time interval due to the changes of the node's health status,

which happens at points 'A', 'B' and 'C'. At point 'A', node 19 changes it health status from SAFE to LOWSAFE after receiving a DIO message from node 13 that has just changed it health status to ALMOST-FAILED. Then, node 19 changes its health status from LOWSAFE to UNSAFE after detecting fire at point 'B' when the fire reaches the node. The node health status is changed again from UNSAFE to ALMOST-FAILED at point 'C', just before the node is totally damaged in the fire.

Figure 5.45 and Figure 5.46 show the RPL trickle timer responses for node 13 and node 19 respectively during the critical phase. As illustrated on the graphs, the RPL nodes do not reset the trickle timer whatever the environmental conditions in the forest fire. This is due to the fact that RPL does not consider information about the node's health status in the routing decision. Thus, there are no occasions for both nodes where the trickle timer interval is reduced to the minimum value. RPL just keeps doubling the trickle timer interval each time the timer expires until it reaches the maximum interval value.

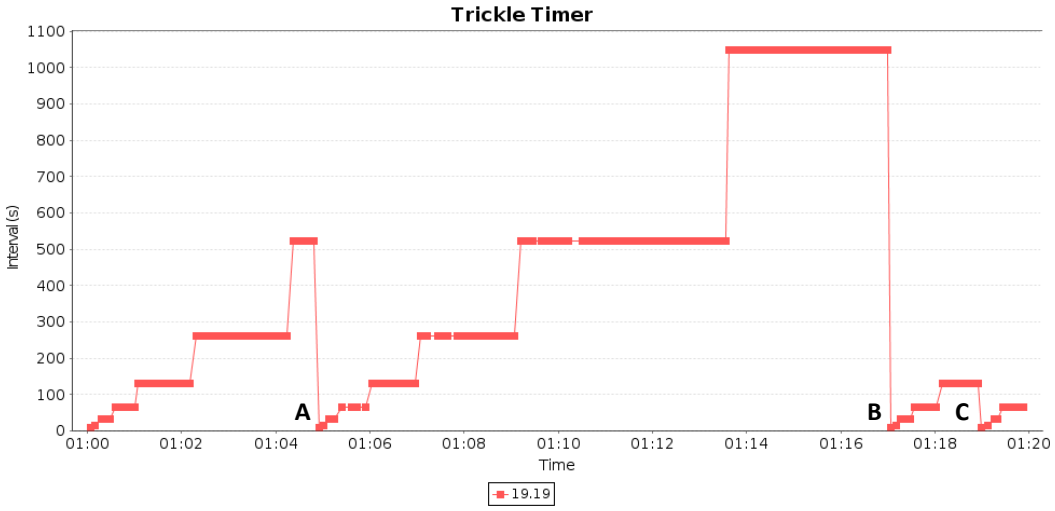


Figure 5.44: MUP (MUP-single) trickle timer response for node 19 in a forest fire situation.

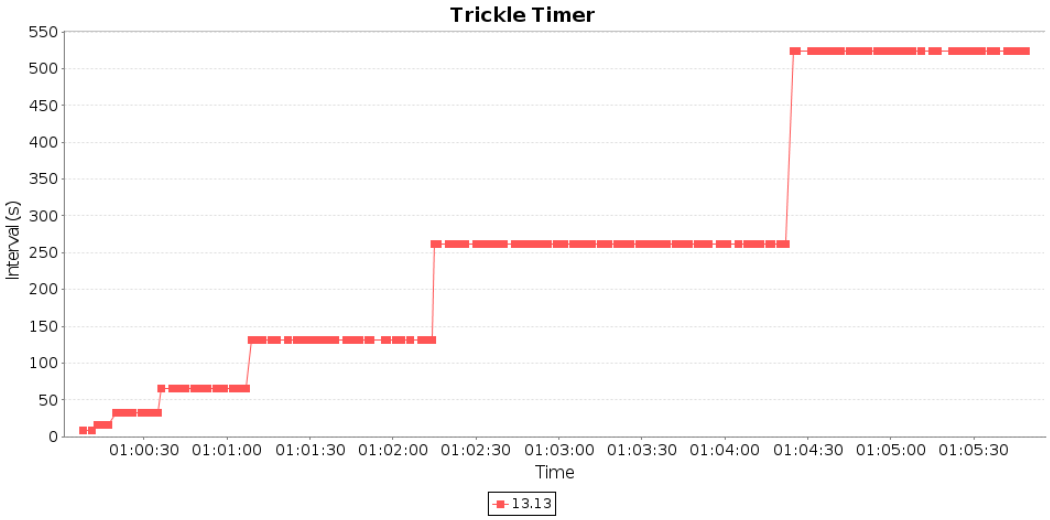


Figure 5.45: RPL trickle timer response for node 13 in in a forest fire situation.

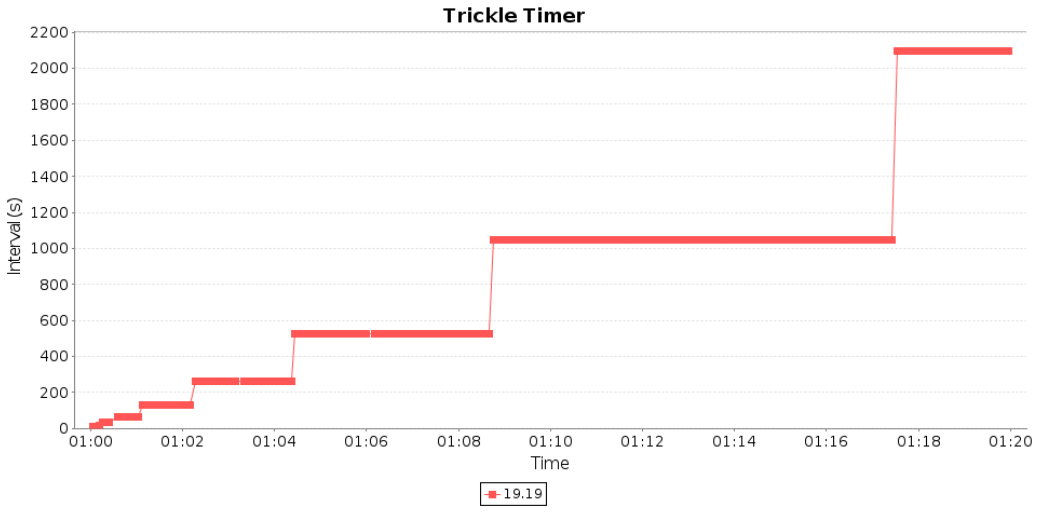


Figure 5.46: RPL trickle timer response for node 19 in a forest fire situation.

5.3.10 Total Number of Packets Collected at the Sink

Figure 5.47 shows the Total number of Packets Collected at the Sink (TPCS) for MUP and RPL over different packet rates in a forest fire situation. As can be seen in the graph, MUP provides a higher TPCS value than RPL. The percentage difference of TPCS values between MUP and RPL is increased with the increasing of the packet rates. At 0.25 and 0.33 packet/second, MUP has achieved slightly higher TPCS values than RPL. At packet rate of 0.5, MUP has recorded about 6% higher TPCS values than RPL. At packet rate of 1 packet/second, MUP has achieved about 8% higher TPCS values than RPL. This is because MUP provides a better improvement of network lifetime at a higher packet rate than at a lower packet rate allowing more packets to be collected by the sink. This network lifetime improvement achieved by MUP at a high packet rate allows more data to be collected from the network. This is a clear indication that MUP provides a better routing performance than RPL under the same network configuration.

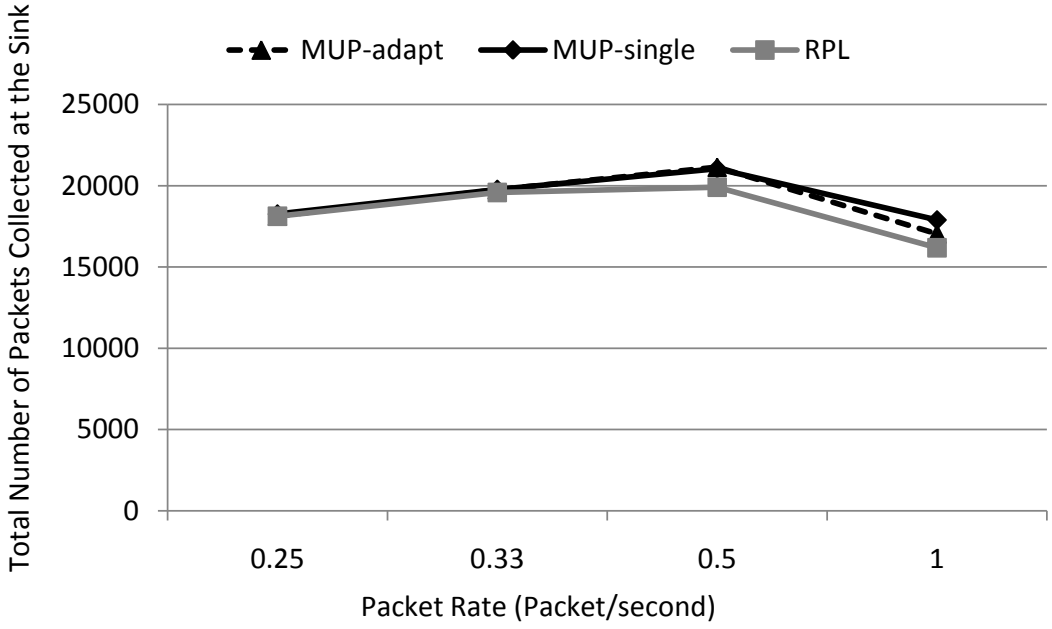


Figure 5.47: Total number of packets collected at the sink (TPCS) over different packet rates for MUP and RPL in a forest fire situation.

5.3.11 Simulation Analysis of MUP in Different Fire Growth Speeds

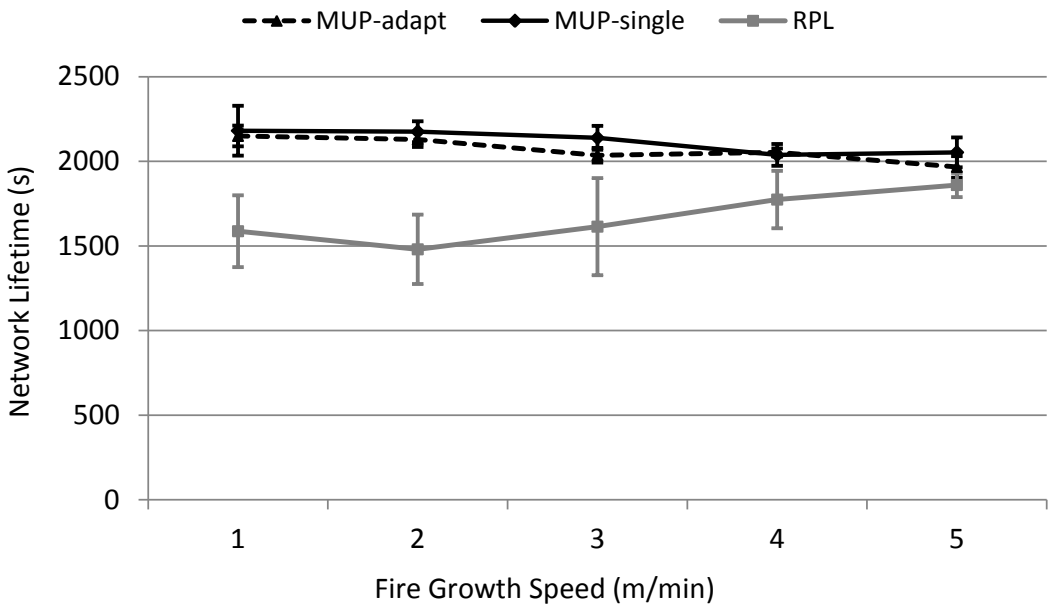


Figure 5.48: Network lifetime over different fire growth speeds for MUP and RPL.

Figure 5.48 shows the network lifetime for MUP and RPL over different fire growth speeds. As can be seen in the graph, MUP provides a longer network lifetime than RPL during the forest fire phase. By taking the average of network lifetime over the fire growth speeds, MUP achieves about 24.5% longer network lifetime than RPL. The network lifetime for MUP drops slowly with the increment of the fire growth speeds. This is due to the fact that faster fire growth speeds mean that MUP has a shorter time to exploit the energy of the unsafe nodes. Other than this, MUP provides a more stable network lifetime when compared to RPL.

Figure 5.49 shows the average energy consumption for unsafe nodes over different forest fire growth speeds for MUP and RPL. The figure shows that MUP provides a higher energy consumption of unsafe nodes than RPL. This is because MUP always route packets through the unsafe nodes to exploit the energy of these nodes. At the same, MUP forwards fewer packets through the safe nodes in order to conserve the energy of the nodes. For this reason, MUP has achieved a lower energy consumption of the safe nodes than RPL, as shown in Figure 5.50.

Figure 5.51 shows the average packet delivery ratio for MUP and RPL for different speeds of fire growth. The graph shows that MUP suffers a lower packet delivery ratio than RPL. By taking an average over the fire growth speeds, MUP has recorded about 11% lower packet delivery ratio than RPL. Figure 5.52 shows that MUP experiences higher end-to-end delay than RPL. MUP requires about

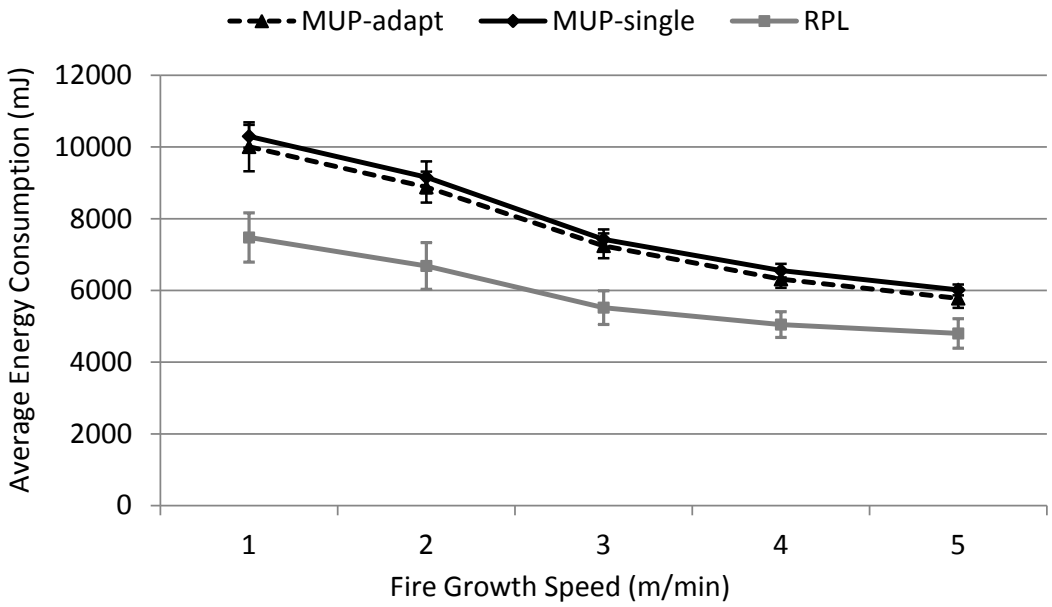


Figure 5.49: Average energy consumption for unsafe nodes over different fire growth speeds for MUP and RPL.

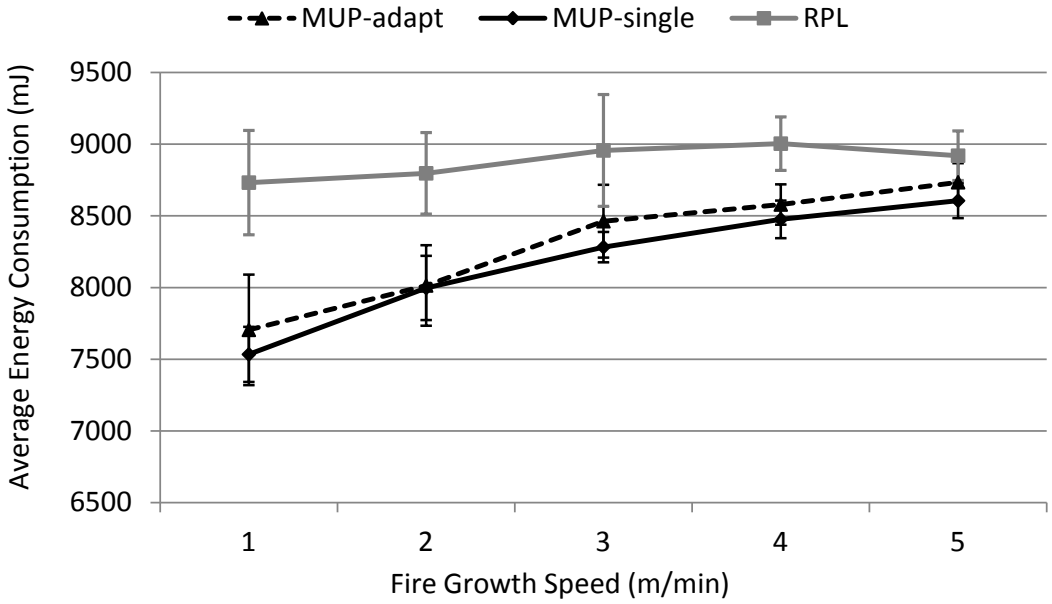


Figure 5.50: Average energy consumption for safe nodes over different fire growth speeds for MUP and RPL.

28% higher end-to-end delay than RPL to deliver a packet to the sink successfully. This is due to the packets being buffered for much longer before they can be transmitted because of congestion and radio interference when MUP forwards the packets through the unsafe nodes.

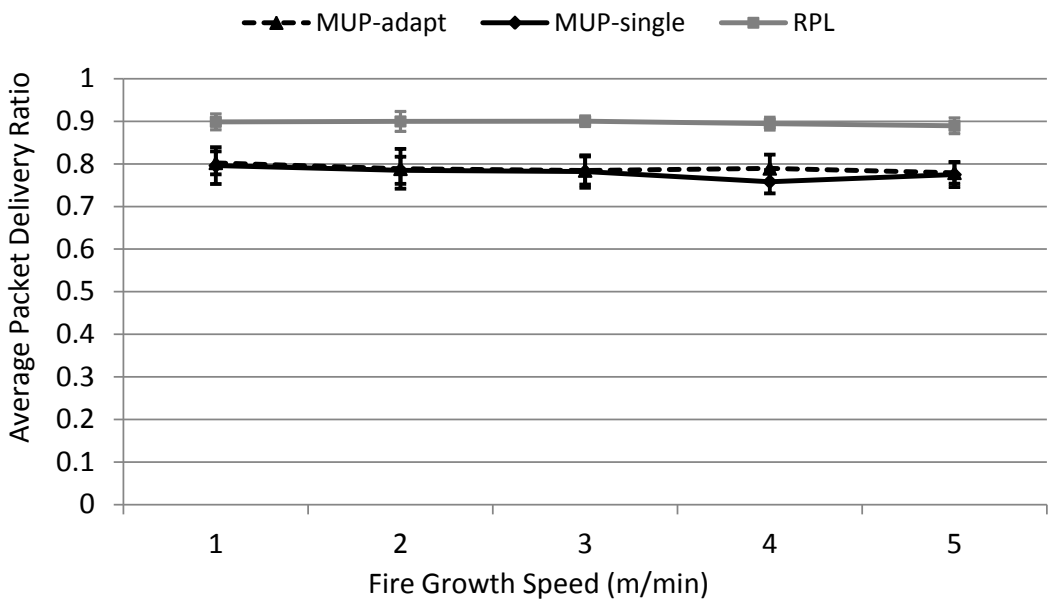


Figure 5.51: Average packet delivery ratio over different fire growth speeds for MUP and RPL.

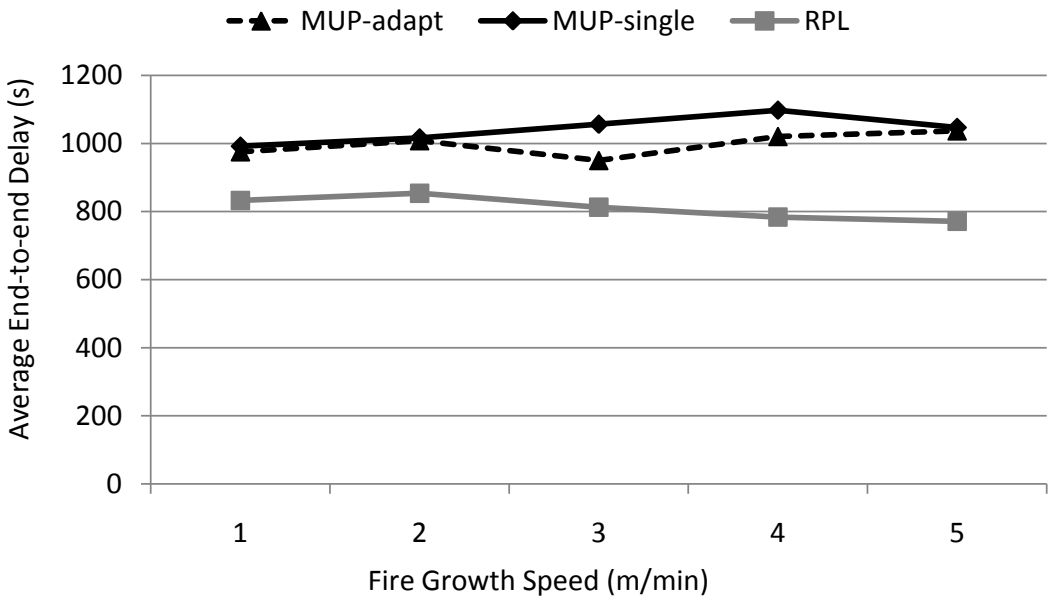


Figure 5.52: Average end-to-end delay over different fire growth speeds.

5.4 Summary

This chapter has presented a simulation study of MUP. As a performance comparison, the simulation study includes the RPL and SAFEST routing protocols. The findings show that MUP provides comparable performance with RPL in a normal situation. During the disaster phase, MUP prolongs the lifetime of the network when compared to RPL and SAFEST. This is achieved by exploiting the energy of the unsafe nodes in order save the energy of the other nodes in the network. MUP aims to route packets through the unsafe nodes, avoiding the safe nodes. Because of this, MUP has recorded a higher energy consumption of the unsafe nodes when compared to RPL and SAFEST. Otherwise, MUP has recorded a lower energy consumption of the safe nodes when compared to RPL and SAFEST. The trade-off is that MUP provides poor packet delivery ratio and end-to-end delay when compared to MUP and SAFEST. This happens mainly due to the higher possibility of congestion and radio interference occurring when MUP concentrates packets through the unsafe nodes. Congestion is likely to happen at the unsafe nodes because the nodes are not able to serve high incoming packets from their neighbour nodes which causes their buffers to become full. When congestion happens, the incoming packets are dropped by the unsafe nodes. In the case of radio interference, a packet is dropped only after the number of retransmission attempts to deliver the packet to the destination node has exceeded the maximum value. In both cases, i.e congestion and radio interference, a packet is buffered much longer in the buffer causing extra delay in delivering the packet to the sink. The study also concludes that MUP sends a higher number of update routing packets when compared to RPL. The main reason is that MUP must inform its neighbour if there are any changes of node health status.

Chapter 6

Experiments with MUP

6.1 Introduction

MUP has been studied through the simulation process in the previous chapter. However, it is important to determine the performance of the routing protocol in a real environment. This chapter discusses the development and implementation of MUP (MUP-single) in a WSN test-bed. The test-bed is made up of Sensinode N740 Nano-Sensor. Each node is deployed in the field by locating it on a stand. The software component is based on the Contiki OS port developed by the Nets research group of Loughborough University for Sensinode/cc2430 [69].

The main experimental work is divided into two phases based on the size of the WSN test-bed: small (10 sensor nodes) and large (25 sensor nodes). The performance of the routing protocol is determined for the small WSN test-bed, which is easier to manage, before proceeding with the more challenging task of the implementation for the large WSN test-bed. The experiment also includes a study of RPL as a comparison. Finally, the experimental results are analysed and the performances of the experiment are compared with the simulation results.

6.2 Hardware Components

The sensor nodes that are used in the WSN test-bed are called N740 Nano-Sensor as shown in Figure 6.1. The N740 Nano-Sensor is a user-programmable sensor node, using Contiki OS firmware which can be programmed into the sensor. It operates at 2.4 GHz radio frequency on IEEE 802.15.4 radio standard, which is based on a TI/Chipcon cc2430 System-on-Chip transceiver. The sensor node uses an 8051 MCU, 8-Bit microcontroller with 128 kB Flash and 8 kB RAM. The N740 Nano-Sensor provides an easy way to build and test 6LoWPAN networks. The N740 Nano-Sensor has a built-in 3-axis accelerometer, LEDs and light sensors

which can be used to build a rapid demonstration; an expansion connector allows more sensors to be attached. The sensor node is a suitable device for software development and research purposes, allowing rapid development and prototyping of any product for IP-based WSNs. The sensor node can be connected to a PC or laptop through a USB port. There are two ways to power the N740 Nano-Sensor either using the USB cable or two AA size batteries.

Figure 6.1: N740 Nano-Sensor

6.3 Software Components

In this experiment, we use three different application examples available in the Contiki OS: the udp-client, the udp-server and the border-router. A combination of the border-router and the udp-client was used in the transmission range measurement and the study of the effect of grass thickness on the node's transmission range. The udp-server and the udp-client were used for the actual experiment in the small and large WSN test-beds.

The udp-client node nodes collect the required data and send it using a UDP packet to the udp-server node, which acts as the sink node. The data contains information about the packet sequence ID, the current health status, the parent address, the number of packet transmissions and receipts, and the number of packets dropped due to congestion and exceeding the maximum retransmission. In addition, information about the time for each component (receive, transmit, CPU and LPM) in active mode, as measured by the Energest module, is also part of the data. The total data length is 30 bytes which is still within the maximum payload size (33 bytes). Because of this, the packet fragmentation function is not needed and can be disabled to reduce the code size.

Table 6.1 shows the code size and memory requirement for the modified modules: udp-client, OF and rpl-dag, which are compiled by using an SDCC compiler. As illustrated in the table, it is shown that both implementations of MUP require extra memory of Flash and RAM when compared to RPL. This is because MUP introduces a new module, which is the critical event detection module, to detect critical events and determine node health statuses. In addition, the new OF and a few modifications to the default Contiki code, introduce extra functions, constants and variables require more memory. MUP-adapt requires more Flash memory than MUP-single that is about 107 bytes Flash to support its adaptive total path cost functionality in the OF but both routing implementations still have the same size of RAM requirement.

Table 6.1: Memory requirement for each modified module (in bytes).

Module	MUP-single		MUP-adapt		RPL	
	FLASH	RAM	FLASH	RAM	FLASH	RAM
udp-client	2656	13	2656	13	2161	13
OF	2002	22	2109	22	1135	15
rpl-dag	12453	758	12453	758	12333	758
Total	17111	793	17218	793	15629	786

6.4 Transmission Range Measurement

6.4.1 Determination of a Suitable Stand Height

The purpose of this range test measurement is to determine a suitable stand height which will be used to elevate each sensor node from the ground in the actual experiment. The height of the stand affects the transmission range of a sensor node. If the sensor node is further from the ground, the transmission range increases and the sensor node provides a wider coverage area. However, the transmission range reduces if the sensor is put nearer to the ground. In the large WSN test-bed, each node must be allowed only to communicate with its surrounding 8 neighbours. A suitable transmission range must be determined so that the WSN test-bed does not cover a large area of deployment making it hard to manage. The same transmission range can be used for the small WSN test-bed which requires a smaller deployment area than the large WSN test-bed.

We consider that a manageable WSN test-bed only covers a 20 m x 20 m area. In the large WSN test-bed, which has 5×5 grid topology configuration, the transmission range for each node must be limited to about 7 m or 8 m, in order to locate

Figure 6.2: Small plastic containers with height 5.5cm.

two neighbouring nodes with a distance of about 5 m. By having this setting, the large WSN test-bed can be deployed within the manageable area. Two methods have been used to limit the transmission range of a sensor node: using the lowest transmit power and adjusting the height of the stand. Initially, the transmission range is tested using a plastic bin with height 26 cm and the transmit power is set to the lowest value. From the measurement results, the sensor node has achieved about a 19 m transmission range which still not suitable for the experiment. For this reason, another stand height is tested using small plastic containers with height 5.5 cm as shown in Figure 6.2. The plastic container has a height lower than the plastic bin to further reduce the node's transmission range.

Figure 6.3: Transmission range measurement setup.

The range test measurement is done on a football field. Three sets of sensor nodes are used in the measurement and each set consists of a border-router and a udp-client node. Both nodes are configured to operate without a radio duty cycle mechanism. The border-router is connected to a laptop via a USB. Tunnelling is established between the border-router node and the laptop using the 'tunslip6' program. The udp-client node is located 1 m away from the border-router as the starting distance. Both sensor nodes are put on the small plastic containers as shown in Figure 6.3. By typing specific instructions at the terminal window, 100 ping commands are sent through the border-router to the udp-client node. Upon receiving each ping message, the udp-client node sends a reply message to the border-router. After the border-router receives the reply message, the message is forwarded to the laptop. Information about each successful pinging process is displayed on the terminal window which includes information about the udp-client IP address, sequence ID, round-trip-time and time-to-live (ttl). Once the sending of 100 ping messages is complete, the statistical results are automatically displayed on the terminal. The same process is repeated for further distances between the udp-client node and the border router: 2, 3, 4, 5, 6, 7 and 8 m. The same measurement is repeated using the remaining sets of sensor nodes.

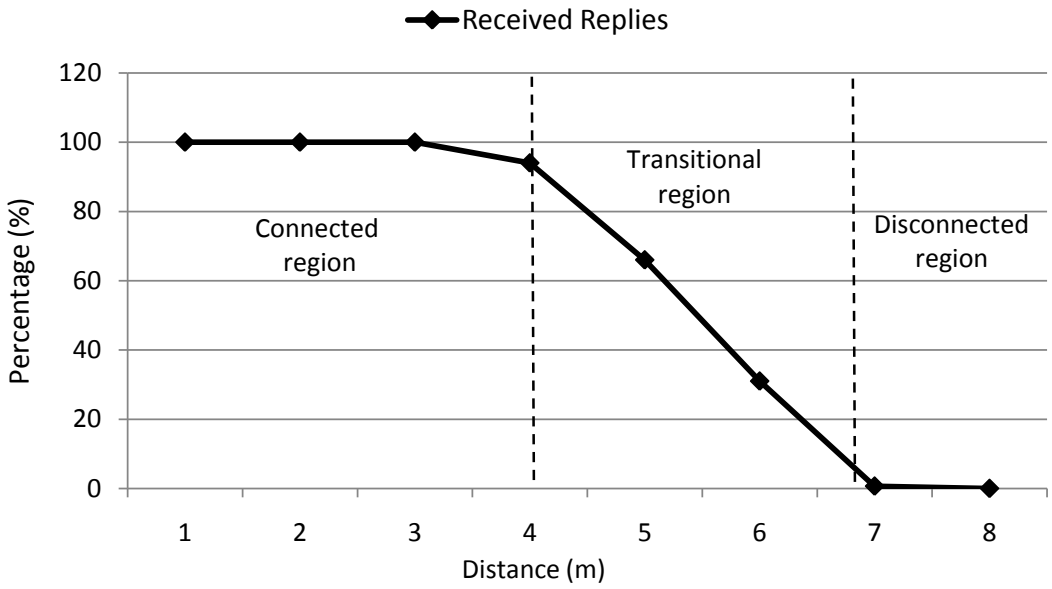


Figure 6.4: Average percentage of received replies over different distances between the border-router and the udp-client node.

The average percentage of received replies for each distance between the border-router and the udp-client nodes is shown in Figure 6.4. The graph is divided into three regions: connected, transitional and disconnected. For distances 4 m and below, this is considered as the connected region, which indicates a good communication link quality giving an average percentage of received replies close to 100%.

However, for 7 m and above, this is considered as the disconnected region because the border router does not receive any reply from the client node due to the very bad communication link. The transitional region exists between the connected and the disconnected region with the average percentage of received replies being reduced linearly with the increment of distance. This happens because of the effect of fluctuation in packet transmission power (non-isotropic transmission) at the transmitter side and the low SNR of the received signal at the receiver side [104]. The transitional region provides an unreliable and unpredictable link which is not suitable for communication. Based on the experimental results, it is shown that the small container can be used as a stand to establish the node's transmission range at about 7 m as required in the actual experiment. Two neighbouring nodes can be located at a distance of 4 m, which still provides a good communication link. At this 4 m distance, the large WSN test-bed can be deployed in an area of 16 m x 16 m, which is within the size of the manageable network.

6.4.2 Effect of Different Grass Thickness on the Transmission Range

The purpose of this measurement is to study the effects of different grass thickness on the node transmission range. Since the actual experiment could take a few months to complete, the variation in the grass thickness of the field cannot be avoided. It is very important to understand whether this variation in grass thickness could influence the node's transmission range. In this measurement, two different grass thicknesses are considered: 2.5 cm and 5 cm. The same measurement configuration as in section 6.4.1 is used in this measurement, which uses a border-router and a udp-client node. Both nodes are configured to have the lowest transmit power and are put on small plastic containers as shown in Figure 6.2. The border-router is connected to a laptop using a USB cable. However, the number of ping commands sent to the udp-client node is increased to 200 ping messages for this measurement instead of the 100 ping messages that were used in the previous measurement. The measurement starts with the 2.5 cm grass thickness and is then repeated with the 5 cm grass thickness. Two sets of sensor nodes consisting of a border-router and a udp-client node are used in the measurement for each grass thickness.

Figure 6.5 shows the average number of received replies over different distances between the border-router and the client node for the two different grass thicknesses. As can be seen in the graph, we observe a similar performance over the two grass thicknesses. This means that the thickness of the grass does not affect the node's transmission range as long a line-of-sight between the border-router

and the udp-client always exists.

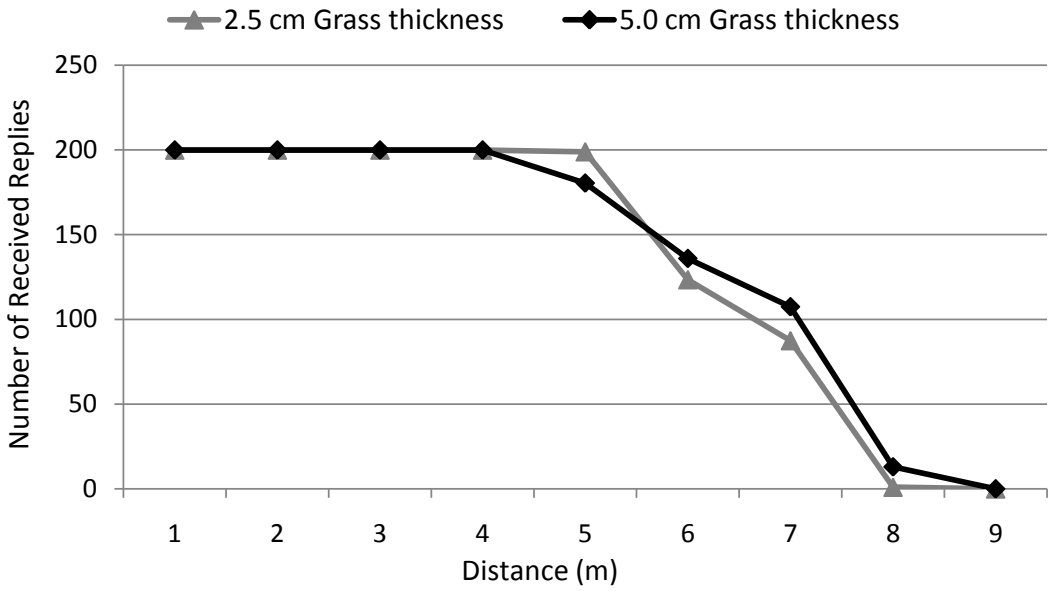
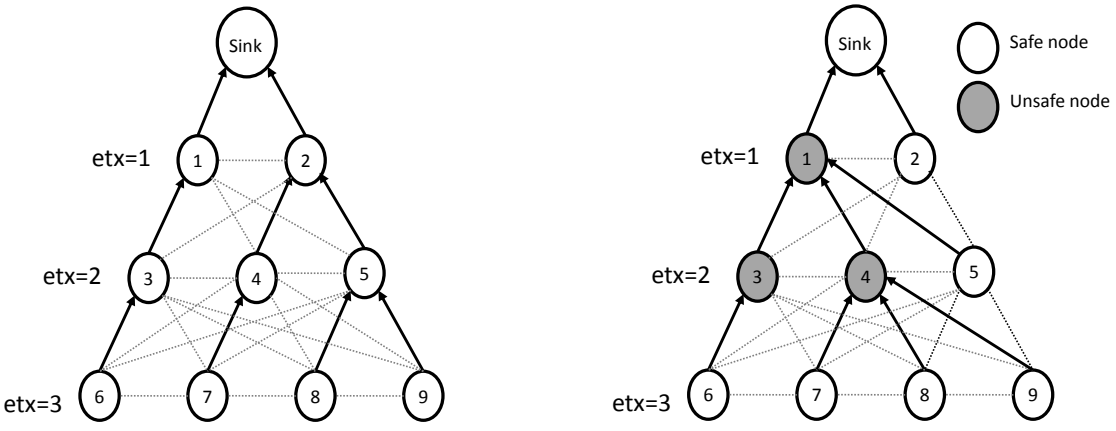


Figure 6.5: Average number of received replies for two different grass thicknesses.

6.5 Performance of MUP in a Small WSN Test-Bed

It is a good approach to determine first the performance of MUP in a small WSN test-bed rather than immediately doing it in a large network. The reason for this is that the experiment is easier to do in the small network than in the large network. In addition, certain issues that may rise during the experiment, preparation of experiment, network management and analysis of experiment results can be understood carefully.



(a) Normal situation (b) Forest fire situation

Figure 6.6: Routing topology in the small WSN test-bed.

In the process of preparing the experiment, it is very important to ensure that each node has a battery with a full energy level to prevent the node becoming non-functional due to depleted energy during the experiment. If this situation happens, the experimental result will be invalid. Because of this, each battery energy level must be checked using a multimeter and any battery with a low energy level must be charged fully or replaced with the other fully charged battery. Other than this, a quick check on the node's functionalities such as neighbourhood discovery, transmitting packet, receiving packet, and LEDs indicator must be carried out.

After the preparation of the experiment has been done properly, a small WSN test-bed consisting of 10 nodes is established for this experiment. The network consists of 9 udp-client nodes and one udp-server node (acting as the sink). The udp-client nodes send data to the sink periodically in four different packet rates: 0.25, 0.33, 0.5 and 1 packet/second. Initially, the network is in a normal situation and all the nodes are always in a safe condition with a SAFE health status, without any node detecting fire as shown in Figure 6.6(a). After about 100 s the experiment starts; a forest fire breaks out in the network. A few seconds after that, nodes 1, 2, and 3 become unsafe almost at the same time to represent the condition of a fire having been detected in the critical region as shown in Figure 6.6(b). These nodes remain in the UNSAFE status for 115 s before they become almost damaged and the nodes need to change their health status to ALMOST-FAILED. About 60 s after that, the nodes are considered totally damaged, and the experiment is stopped. Since nodes are configured to have duty cycling mechanism, we are not leaving the experiment until the end of the network lifetime because it will take a long period of time to finish. Thus, the network lifetime is not measured in the experiment.

First, the energy consumption of all the nodes in the network is analysed. As in the simulation, the analysis of energy consumption is divided into two categories based on the nodes' conditions, either unsafe or safe, during the forest fire phase. A measurement of the energy consumption for each node is taken from the start of the experiment until the end of the experiment. Figure 6.7 shows that MUP-single provides a higher average energy consumption of the unsafe nodes in the critical region than RPL over the packet rates. This is because MUP-single focuses its routes through these unsafe nodes to exploit the energy of these nodes before they are burnt in the fire, in order to save the energy of the safe nodes. This is the reason why MUP-single provides a lower average energy consumption of the safe nodes when compared to RPL as indicated in Figure 6.8. By saving more of the energy of the safe nodes, MUP-single is considered to provide a longer network lifetime than RPL.

Figure 6.9 shows the average packet delivery ratio performance for MUP-single and RPL over different packet rates. As can be observed from the graph, MUP-

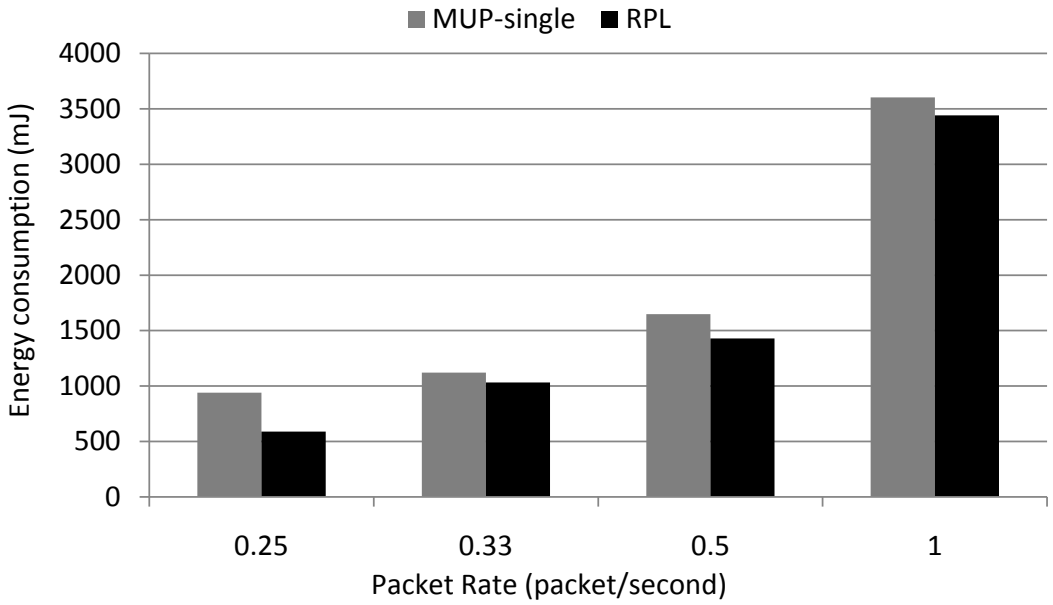


Figure 6.7: Average energy consumption of the unsafe nodes for MUP-single and RPL in a small WSN test bed.

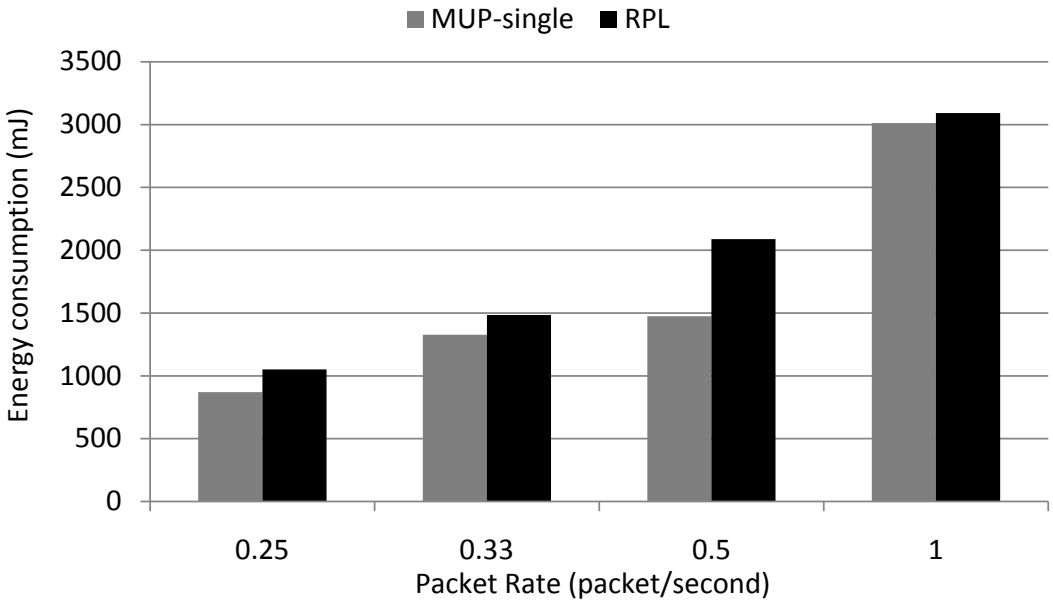


Figure 6.8: Average energy consumption of safe nodes for MUP-single and RPL in a small WSN test bed.

single has provided a lower average packet delivery ratio when compared to RPL, regardless of the packet rate. This is because MUP-single suffers more packet drop when focusing its routes through the unsafe nodes to exploit the energy of the nodes before they become damaged in the fire. By taking the average value over the packet rates, MUP-single has achieved a packet delivery ratio performance of about 86.1%, which is about 5.1% lower when compared to RPL.

There are two reasons for packet drop in the network: congestion and the number of retransmissions exceeding the maximum value. Figure 6.10 shows the total

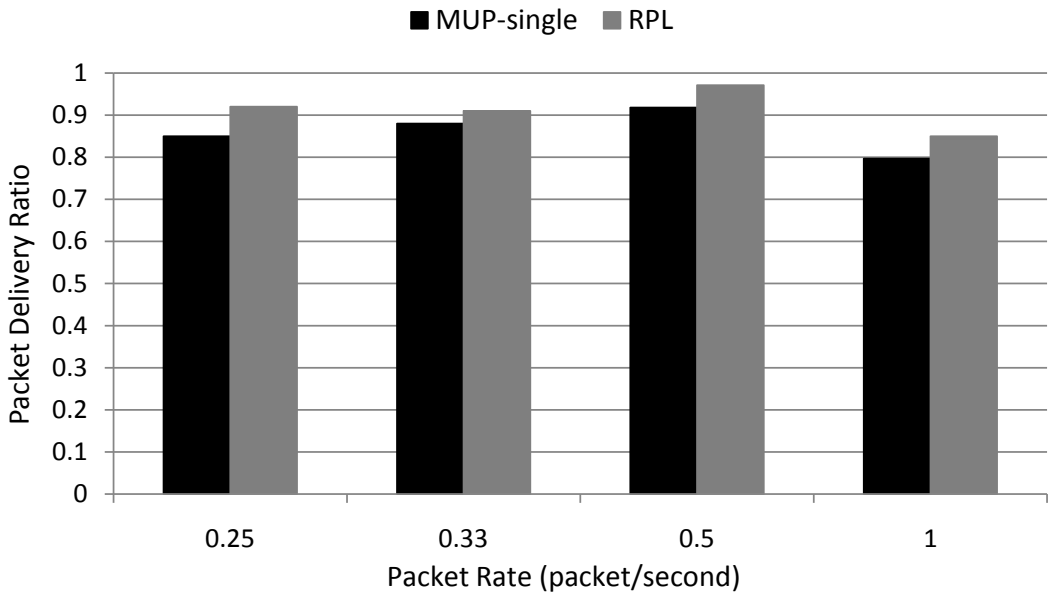


Figure 6.9: Average packet delivery ratio performance over different packet rates for MUP-single and RPL in a small WSN test bed.

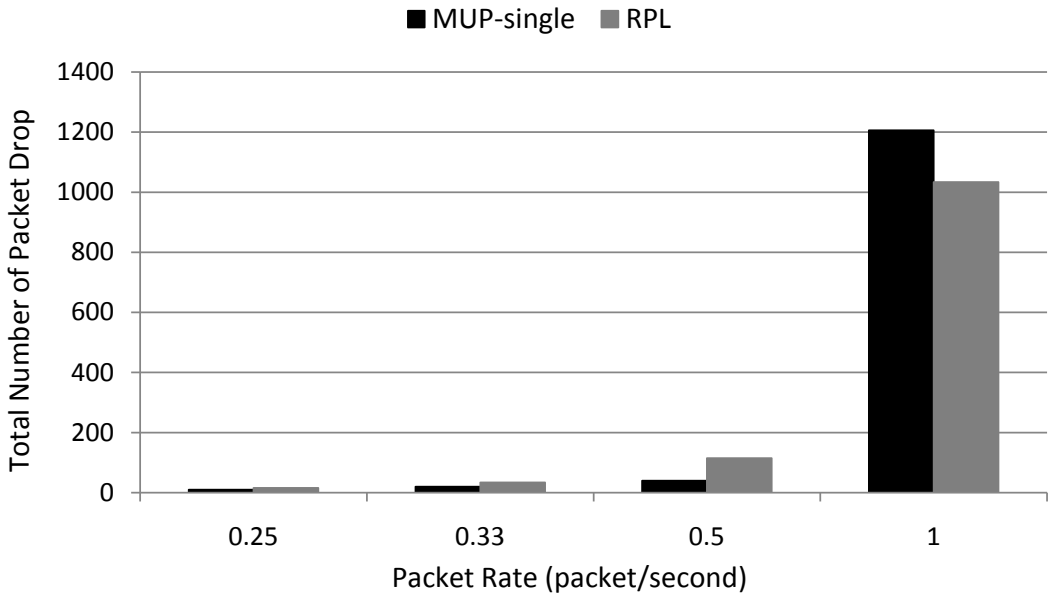


Figure 6.10: Total number of packets dropped due to congestion for MUP-single and RPL in a small WSN test bed.

number of packets dropped due to congestion for MUP-single and RPL. As can be seen from the graph, MUP-single suffers a higher number of packets dropped due to congestion when compared to RPL. This is because MUP-single focuses packets through the unsafe nodes, increasing the chances of congestion occurring at these nodes. In addition, this situation can also increase the possibility of radio interference happening which can cause failure in delivering a packet to the destination successfully. In this case, the node needs to retransmit the packet after waiting for a very short delay. After the number of attempts to retransmit the

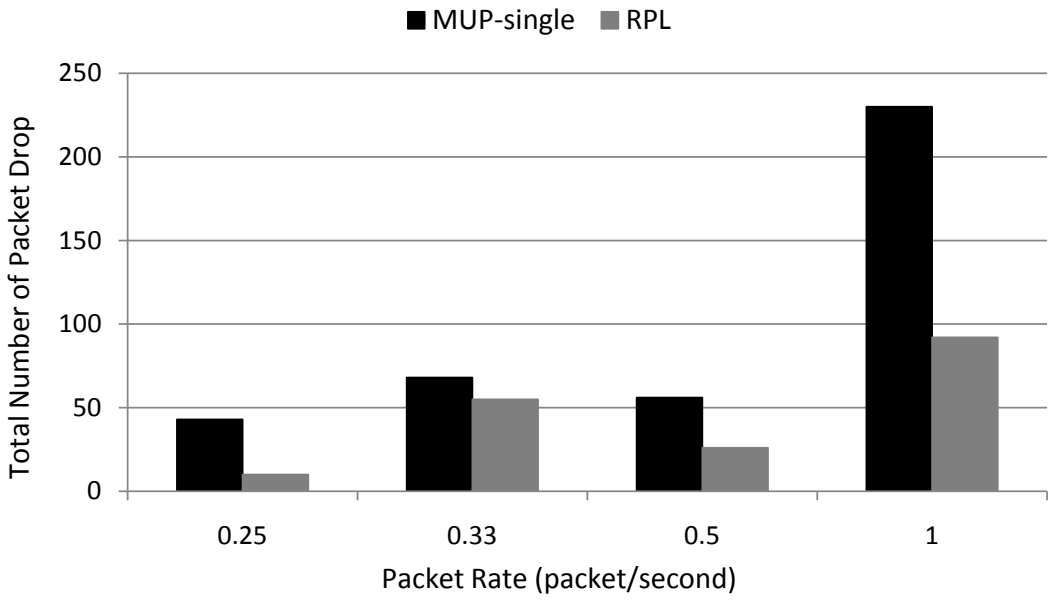


Figure 6.11: Total number of packets dropped due to exceeding the maximum packet retransmission for MUP-single and RPL in a small WSN test bed.

packet has reached the maximum number of retransmissions, the node drops the packet. Therefore MUP-single experiences a higher number of packets dropped due to the number of retransmissions exceeding the maximum value as shown in Figure 6.11.

6.6 Performance of MUP in a Large WSN Test-Bed

After successfully determining the performance of MUP in the small WSN test-bed, the experimental work was continued to determine the performance of the routing protocol in a large WSN test-bed. In order to achieve this, a large WSN test-bed consisting of 24 udp-client nodes and one udp-server node (acting as the sink) was established. The network is arranged in a 5×5 grid topology as shown in Figure 6.12. The sink is located at one corner of the network and connected to a laptop via a USB cable. The network topology in the experiment uses the same configuration that was used in the simulation. Based on the transmission range measurement in section 6.4.1, two neighbouring nodes are separated at a distance of 4 m and each node must be put on a stand (which is a plastic container), in order to achieve a condition where each sensor node is only able to communicate with its adjacent 8 neighbours. The WSN test-bed is deployed in a football field as shown in Figure 6.13. All nodes send packets to the sink at 4 s intervals. An explanation of what happens in the network during the experiment is given as

follows:

1. The experiment starts.
2. A forest fire starts at the centre point between nodes 13, 14, 18, and 19.
3. At 175 s from the experiment's start, these nodes detect fire and have to change their health status from SAFE to UNSAFE.
4. At 290 s, the temperature of these nodes is close to the maximum operating temperature and the nodes become almost damaged. These nodes need to change their health status from UNSAFE to ALMOST-FAILED.
5. At 350 s, the nodes are considered damaged and are disabled from the network by switching them off manually.
6. Just after that, the experiment is stopped.

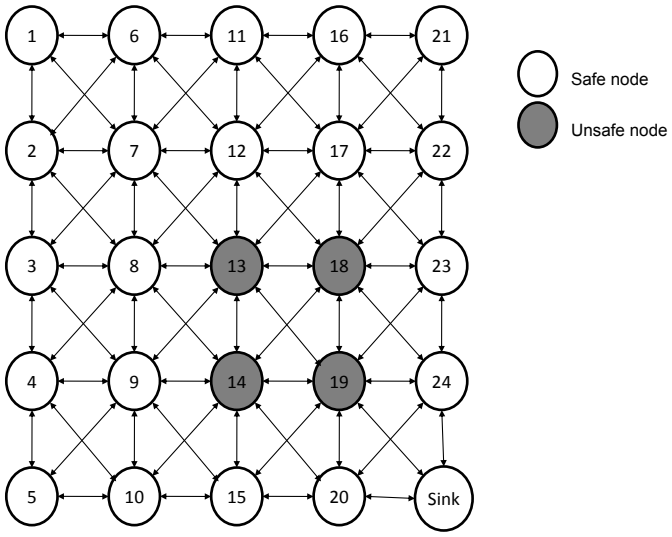


Figure 6.12: Network topology for a large WSN test-bed.

After the preparation of the experiment has been done properly, the WSN test-bed can be deployed on a football field. The deployment of the WSN test-bed takes about 20 min. When everything is ready, the experiment can be started by switching on each sensor node one by one. After all the nodes are switched on, the WSN test-bed is left until the experiment finishes. There is an occasion when we need to enter the WSN test-bed, which is when the unsafe node becomes damaged and needs to be disabled by switching it off manually. For each experimental run, it may take about 6 min to finish.

Table 6.2 shows the performance of MUP-single and RPL at packet rate 0.25 packet/second for the experiment and simulation in the large WSN test-bed.

Figure 6.13: The large WSN testbed deployment on a field.

Table 6.2: Performance of MUP-single and RPL for simulation and experiment in a large WSN test-bed.

Metric	Experiment		Simulation	
	MUP-single	RPL	MUP-single	RPL
Average energy consumption of the safe nodes (mJ)	1875	1950	1812	1893
Average packet delivery ratio	0.745	0.815	0.750	0.821

Based on the experimental results, MUP-single provides a lower average energy consumption of the safe nodes when compared to RPL. As expected, MUP-single always routes packets through the unsafe nodes to exploit the energy of these nodes before they become damaged due to fire, in order to save the energy of the other nodes. MUP-single has recorded an average energy consumption for the safe nodes with a value equal to 1875 mJ, which is about 4% lower than the average energy consumption of the safe nodes recorded by RPL, which is 1950 mJ. By saving the energy of the safe nodes, MUP-single ensures that these nodes always have a higher total remaining energy when compared to RPL, which will contribute to the extension of the network lifetime. In terms of packet delivery ratio, MUP-single achieves about 74.5% packet delivery ratio, which is 7% lower than RPL with its packet delivery ratio being equal to 81.5%.

A simulation is carried out to verify these experimental results. The simulation is configured to be as close as possible to the experiment as described in Appendix A.3. The simulation results show the same performance trend achieved in the experiment, with MUP-single providing a lower energy consumption of safe nodes when compared to RPL. MUP-single has recorded an average energy con-

sumption of the safe nodes with value equal to 1812 mJ, which is lower than the average energy consumption of the safe nodes recorded by RPL, which is 1893 mJ. Even though the average energy consumption of the safe nodes in the simulation is slightly lower than the experimental results, the difference of performance between both routing protocols can be considered to be similar. This difference of performance between both routing protocols for the simulation and experiment is equal to 81 mJ and 75 mJ respectively. In terms of the packet delivery ratio, the simulation provides similar performance with the experiment for both routing protocols.

6.7 Summary

This chapter has described the development and implementation of MUP in WSN test-beds. The test-beds are made up of Sensinode N740 Nano-Sensor. Each node is deployed in the field by locating it on a stand, which is a plastic container with height 5.5 cm. This allows two nodes to be located at a distance 4 m that cover about 20 m x 20 m area, which we consider a manageble size of a WSN test-bed. Two sizes of WSN test-bed are used in the experiment: small test-bed (10 sensor nodes) and large test-bed (25 sensor nodes). Each node is programmed using the Contiki OS, which is the same software code used in the simulation study. The finding shows that MUP requires extra memory of Flash and RAM when compared to RPL. This is because MUP introduces new modules, functions, constants and variables to the default Contiki code. Based on the test-bed results, MUP provides a lower energy consumption of safe nodes when compared to RPL, which is expected to provide a longer network lifetime during a forest fire situation. As a trade-off, MUP suffers a poor packet delivery ratio performance when compared RPL due to congestion and radio transmission interferences. The test-beds results has been verified via a simulation study using lossy links. The simulation results are found to follow a similar performance trend with the test-beds results.

Chapter 7

Conclusions

In normal environment monitoring, a WSN is deployed in an area to sense environmental conditions, such as temperature, humidity, light intensity and pressure, and collect the sensed information. In certain deployment areas, sudden disasters such as forest fires, floods, volcanic eruptions and earth-quakes can happen during monitoring. During such a disaster phase, the events being monitored have the potential to destroy the sensing devices; for example, they can be burnt in a fire, sunk in a flood, burnt in volcano lava, damaged in harmful chemicals etc. There is an opportunity to exploit the energy of these doomed nodes before they are totally destroyed to save the energy of other nodes and prolong network lifetime during the disaster. In order to study this idea, a new routing protocol has been developed which is called the Maximise Unsafe Path (MUP) routing protocol.

MUP is a proactive routing protocol designed specifically for normal applications. The main objective of MUP is to prolong the WSN lifetime during a critical phase. This is achieved by exploiting the energy of the unsafe nodes which are becoming destroyed, in order to save the energy of the other nodes. In order to achieve this, MUP adapts alongside the environment by taking into consideration the environmental conditions in its routing decision. MUP introduces a new parameter to represent the environmental threat to a node, known as the health status. Four level of node health status are defined to classify the different levels of environmental threat: SAFE, LOWSAFE, UNSAFE, and ALMOST-FAILED. Determination of health status is based on the available information from the sensor readings (such as temperature) and the condition of the neighbour nodes. The routing decision-making is made by giving priority to nodes with health status in the following order: UNSAFE, LOWSAFE, SAFE and ALMOST-FAILED. In order to avoid the energy wastage caused by selecting routes that require a higher energy cost to deliver a packet to the sink, MUP always forwards packets through nodes that have the minimum total path cost. The total path cost is calculated based on the ETX metric and is included as an additional parameter in the routing decision.

The MUP design concept consists of three functional modules: routing management, neighbourhood management and critical event detection. These modules are designed as an extension of RPL. The routing management module is implemented by introducing a new OF module. The OF module defines how nodes select and optimise routes within the network. The neighbourhood management module uses the existing neighbour discovery mechanism which is available in RPL. Lastly, the critical event detection module is developed as a new module. Two implementations of MUP have been developed which are MUP-single and MUP-adapt.

MUP has been studied using simulation and test-bed experiments. A forest fire scenario is implemented in the simulator environment using the simplest circular-shaped fire growth pattern for a flat terrain without any wind. A linear increment of the temperature model is used when a node is exposed to fire. The simulation study has included both implementations of the MUP routing protocol. Based on the simulation results, it is found that MUP provides a longer network lifetime when compared to RPL during a forest fire phase. This is achieved because MUP focuses packets through the unsafe nodes to exploit the energy of these failing nodes, which provide a lower energy consumption of the other nodes. Despite having a longer network lifetime, MUP suffers a drop in the packet delivery ratio and end-to-end delay performance. This is caused by packets being dropped due to congestion and the number of retransmissions exceeding the maximum value when packets are focussed through the unsafe nodes. Since congestion is more likely to happen at the unsafe nodes, a packet may get buffered for much longer, which contributes to the extra delay in delivering the packet to the sink. The gain in the network lifetime is proven to be beneficial compared to the drop in the packet delivery ratio and end-to-end delay performance that MUP has to sacrifice. In a normal situation, MUP provides similar routing performance with RPL. In order to determine the performance of MUP in the real environment, the routing protocol has been implemented on WSN test-beds. The performance of MUP is studied in small (10 nodes) and large (25 nodes) WSN test-beds. The experimental results are found to follow the same performance trends observed in the simulation.

In general, MUP has demonstrated that the idea of exploiting the energy of the nodes that are going to be destroyed, in order to save the energy of the other nodes, is able to provide a longer network lifetime during a critical phase. MUP aims to forward packets through the unsafe nodes in the critical area, to exploit the energy of the nodes in order to save the energy of nodes in the safe areas. By saving the energy of the safe nodes, MUP provides a longer network lifetime, allowing the network to operate for as long as possible to detect and monitor the

environment during the critical phase. MUP implementations are not limited to forest monitoring applications but are also suitable for other applications such as floods, active volcanoes and landslide monitoring systems.

7.1 Future Works

Further works can be carried out to enhance the performance of MUP. Recommendations for future work are described as follows:

1. Investigate the performance of MUP with a congestion-aware duty cycle mechanism [63], which is expected to provide an improvement of the packet delivery ratio and end-to-end delay performance.

2. Explore the feasibility of implementing MUP in other types of disaster phase, such as floods, volcanic eruptions and landslides.

References

[1] Adel Ali Ahmed and Norsheila Fisal Fisal. Secure real-time routing protocol with load distribution in wireless sensor networks. *Security and Communication Networks*, 4(8):839–869, 2011.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422, 2002.

[3] Ian F. Akyildiz, Tommaso Melodia, and Kaushik R. Chowdhury. A survey on wireless multimedia sensor networks. *Computer Networks*, 51(4):921 – 960, 2007.

[4] J.N. Al-Karaki and A.E. Kamal. Routing techniques in wireless sensor networks: a survey. *Wireless Communications, IEEE*, 11(6):6 – 28, dec. 2004.

[5] Martin E. Alexander. Calculating and interpreting forest fire intensities. *Canadian Journal of Botany*, 60(4):349–357, 1982.

[6] I. Banerjee, S. Chakrabarti, A. Bhattacharyya, and U. Ganguly. An energy aware routing design to maximize lifetime of a wireless sensor network with a mobile base station. In *Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on*, pages 2135–2141, Sept 2014.

[7] M. Bhardwaj, T. Garnett, and A.P. Chandrakasan. Upper bounds on the lifetime of sensor networks. In *Communications, 2001. ICC 2001. IEEE International Conference on*, volume 3, pages 785 –790 vol.3, 2001.

[8] A. Boukerche, R.W.N. Pazzi, and R.B. Araujo. HPEQ a hierarchical periodic, event-driven and query-based wireless sensor network protocol. In *Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on*, pages 560 –567, November 2005.

[9] Azzedine Boukerche and Anahit Martirosyan. An energy efficient and low latency multiple events' propagation protocol for wireless sensor networks

with multiple sinks. In *Proceedings of the 4th ACM workshop on Performance evaluation of wireless ad hoc, sensor,and ubiquitous networks*, PE-WASUN '07, pages 82–86, New York, NY, USA, 2007. ACM.

[10] Azzedine Boukerche, Richard Werner Nelem Pazzi, and Regina Borges Araujo. A fast and reliable protocol for wireless sensor networks in critical conditions monitoring applications. In *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, MSWiM '04, pages 157–164, New York, NY, USA, 2004. ACM.

[11] David W. Carman, Peter S. Kruus, and Brian J. Matt. Constraints and approaches for distributed sensor network security (final). *DARPA Project report,(Cryptographic Technologies Group, Trusted Information System, NAI Labs)*, 1:1, 2000.

[12] David W. Casbeer, R.W. Beard, T.W. McLain, Sai-Ming Li, and Raman K. Mehra. Forest fire monitoring with multiple small UAVs. In *American Control Conference, 2005. Proceedings of the 2005*, pages 3530–3535. IEEE, 2005.

[13] Alberto Cerpa, Jennifer L. Wong, Louane Kuang, Miodrag Potkonjak, and Deborah Estrin. Statistical model of lossy links in wireless sensor networks. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press.

[14] Jae-Hwan Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *Networking, IEEE/ACM Transactions on Networking*, 12(4):609 – 619, August 2004.

[15] Yunxia Chen and Qing Zhao. On the lifetime of wireless sensor networks. *Communications Letters, IEEE*, 9(11):976 – 978, November 2005.

[16] Chee-Yee Chong and S.P. Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247 – 1256, August 2003.

[17] W. Colitti, K. Steenhaut, N. De Caro, B. Buta, and V. Dobrota. REST enabled wireless sensor networks for seamless integration with web applications. In *IEEE 8th International Conference on Mobile Adhoc and Sensor Systems (MASS), 2011*, pages 867–872, 2011.

[18] W. Colitti, K. Steenhaut, B. Lemmens, and J. Borms. Simulation tool for wireless sensor network constellations in space. In *International Conference on Ultra Modern Telecommunications Workshops, 2009. ICUMT '09*, pages 1 –5, October 2009.

[19] Canadian wildland fire information system : National wildland fire situation report. http://cwfis.cfs.nrcan.gc.ca/en_CA/datamart, 2013.

[20] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. *Wireless Networks*, 11(4):419–434, July 2005.

[21] Eric Den Breejen, Marcel Breuers, Frank Cremer, Rob Kemp, Marco Roos, Klamer Schutte, and Jan S De Vries. Autonomous forest fire detection. In *Proc. 3rd Int. Conf. on Forest Fire Research*, pages 2003–2012, 1998.

[22] Kosmas Dimitropoulos, Kivanc Kose, Nikos Grammalidis, and Enis Çetin. Fire detection and 3D fire propagation estimation for the protection of cultural heritage areas. In *ISPRS Technical Commission VIII Symposium*, volume 38, pages 620–625, 2010.

[23] David M. Doolin and Nicholas Sitar. Wireless sensors for wildfire monitoring. In *Proc. of SPIE Symposium on Smart Structures and Materials*, San Diego, CA, USA, March 2005.

[24] S. Dulman, T. Nieberg, Jian Wu, and P. Havinga. *Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks*, volume 3. March 2003.

[25] A Dunkels, B. Gronvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pages 455–462, Nov 2004.

[26] Adam Dunkels. Rime - a lightweight layered communication stack for sensor networks. In *European Conference on Wireless Sensor Networks (EWSN)*, January 2007, Delft, The Netherlands.

[27] Adam Dunkels, Fredrik Österlind, Nicolas Tsiftes, and Zhitao He. Software-based on-line energy estimation for sensor nodes. In *Proceedings of the 4th workshop on Embedded networked sensors*, EmNets '07, pages 28–32, New York, NY, USA, 2007. ACM.

[28] M. Dwijaksara, Doyoung Chung, Y. Park, Jangseong Kim, and Kwangjo Kim. Secure, fast rebuilding, and energy efficient routing protocol for mission-critical application over wireless sensor networks. In *Proceedings of the Symposium on Cryptography and Information Security, Kokura, Japan*, pages 25–28, 2011.

[29] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next century challenges: scalable coordination in sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom '99, pages 263–270, New York, NY, USA, 1999. ACM.

[30] Jin Fan and D.J. Parish. Using a genetic algorithm to optimize the performance of a wireless sensor network. In *The 8th Annual Postgraduate Symposium, The Convergence of Telecommunications, Networking and Broadcasting, Liverpool John Moores University, 28th-29th June*, 2007.

[31] E. Felemban, Chang-Gun Lee, and E. Ekici. MMSPEED: multipath multi-speed protocol for QoS guarantee of reliability and timeliness in wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 5(6):738–754, 2006.

[32] J. Fleming and R.G. Robertson. Fire management tech tips: The osborne fire finder. Technical report, Technical Report 0351 1311-SDTDC, USDA Forest Service, 2003.

[33] Omprakash Gnawali. The minimum rank with hysteresis objective function. RFC 6719, September 2012.

[34] E.B. Hamida and G. Chelius. Analytical evaluation of virtual infrastructures for data dissemination in wireless sensor networks with mobile sink. In *Proceedings of the First ACM workshop on Sensor and actor networks*, pages 3–10. ACM, 2007.

[35] Carl Hartung, Richard Han, Carl Seielstad, and Saxon Holbrook. Firewxnet: A multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 28–41. ACM, 2006.

[36] Tian He, John A Stankovic, Chenyang Lu, and Tarek Abdelzaher. SPEED: A stateless protocol for real-time communication in sensor networks. *International Conference on Distributed Computing Systems*, 0:46, 2003.

[37] Wendi Rabiner Heinzelman, Joanna Kulik, and Hari Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom '99, pages 174–185, New York, NY, USA, 1999. ACM.

[38] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, page 10 pp. vol.2, January 2000.

[39] Danny Hughes, Phil Greenwood, Gordon Blair, Geoff Coulson, Paul Grace, Florian Pappenberger, Paul Smith, and Keith Beven. An experiment with reflective middleware to support grid-based flood monitoring. *Concurrency and Computation: Practice and Experience*, 20(11):1303–1316, 2008.

[40] J.W. Hui and D.E. Culler. Extending IP to low-power, wireless personal area networks. *Internet Computing, IEEE*, 12(4):37–45, 2008.

[41] *IEEE Std 802.15.4-2011, IEEE Standard for local and metropolitan area networks - Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, 2011.

[42] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, pages 56–67, New York, NY, USA, 2000. ACM.

[43] Sinan Isik, Mehmet Yunus Donmez, Can Tunca, and Cem Ersoy. Performance evaluation of wireless sensor networks in realistic wildfire simulation scenarios. In *Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems*, pages 109–118. ACM, 2013.

[44] Christine Jardak, Krisakorn Rerkrai, Aleksandar Kovacevic, Janne Riihijarvi, and Petri Mahonen. Design of large-scale agricultural wireless sensor networks: email from the vineyard. *Int. J. Sen. Netw.*, 8(2):77–88, August 2010.

[45] Joseph M. Kahn, Randy Howard Katz, and Kristofer S.J. Pister. Emerging challenges: Mobile networking for "smart dust". *Communications and Networks, Journal of*, 2(3):188–196, Sept 2000.

[46] Konstantinos Kalpakis, Koustuv Dasgupta, and Parag Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Network : The International Journal of Computer and Telecommunications Networking*, 42(6):697–716, August 2003.

[47] Intae Kang and R. Poovendran. Maximizing static network lifetime of wireless broadcast ad-hoc networks. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 3, pages 2256 – 2261 vol.3, May 2003.

[48] Jeonggil Ko, Joakim Eriksson, Nicolas Tsiftes, Stephen Dawson-Haggerty, Andreas Terzis, Adam Dunkels, and David Culler. ContikiRPL and TinyRPL: Happy Together. In *Proceedings of the workshop on Extending the Internet to Low power and Lossy Networks (IP+SN2011)*, Chicago, IL, USA, April 2011.

[49] Takuma Koga, Kentaroh Toyoda, and Iwao Sasase. Priority based routing for forest fire monitoring in wireless sensor network. *Journal of Telecommunications & Information Technology*, 2014(3), 2014.

[50] Anis Koubâa, Mário Alves, and Eduardo Tovar. IEEE 802.15.4 for wireless sensor networks: a technical overview. *IPP-HURRAY Technical Report (TR-050702)*, 2005.

[51] E. Kuhrt, J. Knollenberg, and V. Mertens. An automatic early warning system for forest fires. *Annals of Burns and Fire Disasters*, 14(3):151–154, 2001.

[52] S. Kumar and D. Shepherd. SensIT: Sensor information technology for the warfighter. In *Proc. 4th Int. Conf. on Information Fusion*, 2001.

[53] N Kushalnagar, G Montenegro, C Schumacher, et al. IPv6 over low-power wireless personal area networks (6LoWPANs): overview, assumptions, problem statement, and goals. *RFC4919, August*, 10, 2007.

[54] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, page 8 pp., april 2006.

[55] Sang Hyuk Lee, Soobin Lee, Heecheol Song, and Hwang-Soo Lee. Wireless sensor network design for tactical military applications : Remote large-scale environments. In *Military Communications Conference, 2009. MILCOM 2009. IEEE*, pages 1–7, Oct 2009.

[56] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko. The Trickle Algorithm. RFC 6206 (Proposed Standard), March 2011.

[57] Shining Li, Jin Cui, and Zhigang Li. Wireless sensor network for precise agriculture monitoring. In *Intelligent Computation Technology and Automation (ICICTA), 2011 International Conference on*, volume 1, pages 307 –310, march 2011.

[58] S. Lindsey and C.S. Raghavendra. PEGASIS: Power-efficient gathering in sensor information systems. In *Aerospace Conference Proceedings, 2002. IEEE*, volume 3, pages 3–1125 – 3–1130 vol.3, 2002.

[59] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, pages 88–97, New York, NY, USA, 2002. ACM.

[60] A. Manjeshwar and D.P. Agrawal. TEEN: a routing protocol for enhanced efficiency in wireless sensor networks. In *Parallel and Distributed Processing Symposium., Proceedings 15th International*, pages 2009 –2015, April 2001.

[61] A. Manjeshwar and D.P. Agrawal. APTEEN: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. In *Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM*, pages 195 –202, 2002.

[62] Cecilia Mascolo and Mirco Musolesi. SCAR: context-aware adaptive routing in delay tolerant mobile sensor networks. In *Proceedings of the 2006 international conference on Wireless communications and mobile computing*, IWCMC '06, pages 533–538, New York, NY, USA, 2006. ACM.

[63] V. Michopoulos, Lin Guan, G. Oikonomou, and I. Phillips. DCCC6: Duty cycle-aware congestion control for 6lowpan networks. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 278 –283, March 2012.

[64] Vasilis Michopoulus. *Congestion and medium access control in 6LoWPAN WSN*. PhD thesis, Loughborough University, 2012.

[65] Gabriel Montenegro, Nandakishore Kushalnagar, J Hui, and D Culler. Transmission of IPv6 packets over ieee 802.15.4 networks. *Internet proposed standard RFC*, 4944, 2007.

[66] S.D. Muruganathan, D.C.F. Ma, R.I. Bhasin, and A.O. Fapojuwo. A centralized energy-efficient routing protocol for wireless sensor networks. *Communications Magazine, IEEE*, 43(3):S8 – 13, March 2005.

[67] Donggeon Noh, Dongeun Lee, and Heonshik Shin. Mission-oriented selective routing for wireless sensor networks. In *Communications and Networking in China, 2007. CHINACOM '07. Second International Conference on*, pages 809 –813, aug. 2007.

[68] Federation of American Scientists. AN/GSQ-187 REMBASS. Internetfromhttp://www.fas.org/man/dod-101/sys/land/rembass.htm., July 2011.

[69] George Oikonomou and Iain Phillips. Experiences from porting the contiki operating system to a popular hardware platform. In *Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference on*, pages 1–6. IEEE, 2011.

[70] World Meteorological Organization. World: Highest temperature. http://wmo.asu.edu/world-highest-temperature, July 2014.

[71] F. Österlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level sensor network simulation with cooja. In *Proceedings of The 31st IEEE Conference on Local Computer Networks, 2006*, pages 641 –648, November 2006.

[72] Fredrik Österlind, Joakim Eriksson, and Adam Dunkels. Cooja timeline: a power visualizer for sensor network simulation. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 385–386. ACM, 2010.

[73] Jeongyeup Paek, K. Chintalapudi, R. Govindan, J. Caffrey, and S. Masri. A wireless sensor network for structural health monitoring: Performance and experience. In *The Second IEEE Workshop on Embedded Networked Sensors (EmNetS-II), 2005*, pages 1 – 10, May 2005.

[74] Shamim N. Pakzad, Gregory L. Fenves, Sukun Kim, and David E. Culler. Design and implementation of scalable wireless sensor network for structural monitoring. *Journal of Infrastructure Systems*, 14(1):89–101, 2008.

[75] R.W.N. Pazzi and A. Boukerche. Mobile data collector strategy for delay-sensitive applications over wireless sensor networks. *Computer Communications*, 31(5):1028–1039, 2008.

[76] Han Peng, Zhou Xi, Li Ying, Chen Xun, and Gao Chuanshan. An adaptive real-time routing scheme for wireless sensor networks. In *The 21st International Conference on Advanced Information Networking and Applications Workshops, 2007, AINAW '07*, volume 2, pages 918–922, 2007.

[77] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella. Performance study of IEEE 802.15.4 using measurements and simulations. In *Wireless Communications and Networking Conference, 2006. WCNC 2006. IEEE*, volume 1, pages 487 –492, April 2006.

[78] Maneesha V. Ramesh. Real-time wireless sensor network for landslide detection. In *Third International Conference on Sensor Technologies and Applications, 2009. SENSORCOMM'09*, pages 405–409. IEEE, 2009.

[79] H. Rangarajan and J.J. Garcia-Luna-Aceves. Reliable data delivery in event-driven wireless sensor networks. In *Ninth International Symposium on Computers and Communications, 2004. Proceedings. ISCC 2004*, volume 1, pages 232–237 Vol.1, June 2004.

[80] Kay Römer. Tracking real-world phenomena with smart dust. In *Wireless Sensor Networks*, volume 2920, pages 28–43. Springer Berlin / Heidelberg, 2004. 10.1007/978-3-540-24606-0_3.

[81] Alberto Rosi, Matteo Berti, Nicola Bicocchi, Gabriella Castelli, Alessandro Corsini, Marco Mamei, and Franco Zambonelli. Landslide monitoring with sensor networks: experiences and lessons learnt from a real-world deployment. *International Journal of Sensor Networks*, 10(3):111–122, 2011.

[82] C. Schurgers and M.B. Srivastava. Energy efficient routing in wireless sensor networks. In *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE*, volume 1, pages 357 – 361 vol.1, 2001.

[83] Sensinode. http://www.sensinode.com/. Accessed May 23, 2011.

[84] Kewei Sha, Junzhao Du, and Weisong Shi. WEAR: a balanced, fault-tolerant, energy-aware routing protocol in WSNs. *Int. J. Sen. Netw.*, 1:156–168, January 2006.

[85] Kewei Sha, Weisong Shi, and O. Watkins. Using wireless sensor networks for fire rescue applications: Requirements and challenges. In *Electro/information Technology, 2006 IEEE International Conference on*, pages 239–244, May 2006.

[86] Byungrak Son, Yong-sork Her, and J Kim. A design and implementation of forest-fires surveillance system based on wireless sensor networks for South Korea mountains. *International Journal of Computer Science and Network Security (IJCSNS)*, 6(9):124–130, 2006.

[87] M. Soyturk and D.T. Altilar. Reliable real-time data acquisition for rapidly deployable mission-critical Wireless Sensor Networks. In *INFOCOM Workshops 2008, IEEE*, pages 1–6. IEEE, 2008.

[88] Alexandru Stan. Porting the core of the Contiki operating system to the TelosB and MicaZ platforms. *Computer Science, International University Bremen, Campus Ring, Bremen, Germany*, 1:28759, 2007.

[89] Čedomir Stefanović, Vladimir Crnojević, Dejan Vukobratović, Lorenzo Niccolai, Francesco Chiti, and Romano Fantacci. Contaminated areas monitoring via distributed rateless coding with constrained data gathering. In *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*, IWCMC '10, pages 671–675, New York, NY, USA, 2010. ACM.

[90] Sameer Tilak, Nael B Abu-Ghazaleh, and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(2):28–36, 2002.

[91] T.T. Truong, K.N. Brown, and C.J. Sreenan. Using mobile sinks in wireless sensor networks to improve building emergency response. In *Academy Research Colloquium on Wireless as an Enabling Technology*, 2010.

[92] Tsvetko Tsvetkov. Rpl: Ipv6 routing protocol for low power and lossy networks. *Sensor Nodes–Operation, Network and Application (SN)*, 59:2, 2011.

[93] N. A. Vasanthi and S. Annadurai. Pattern based routing for event driven wireless sensor-actor networks. In *Proceedings of the 1st Amrita ACM-W Celebration on Women in Computing in India*, A2CWiC '10, pages 43:1–43:6, New York, NY, USA, 2010. ACM.

[94] Thiemo Voigt, Joakim Eriksson, Fredrik Österlind, Robert Sauter, Nils Aschenbruck, Pedro J. Marrón, Vinny Reynolds, Lei Shu, Otto Visser, Anis Koubaa, and Andreas Köpke. Towards comparable simulations of cooperating objects and wireless sensor networks. In *Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools*, VALUETOOLS '09, pages 77:1–77:10, ICST, Brussels, Belgium,

Belgium, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[95] C. E. Van Wagner. A simple fire-growth model. *The Forestry Chronicle*, 45(2):103–104, 1969.

[96] John Paul Walters, Zhengqiang Liang, Weisong Shi, and Vipin Chaudhary. Wireless sensor network security: A survey. *Security in distributed, grid, mobile, and pervasive computing*, 1:367, 2007.

[97] Bernd-Ludwig Wenning, Dirk Pesch, Andreas Timm-Giel, and Carmelita Görg. Environmental monitoring aware routing: making environmental sensor networks more robust. *Telecommunication Systems*, 43:3–11, 2010. 10.1007/s11235-009-9191-8.

[98] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor network. In *Proceeedings of the Second European Workshop on Wireless Sensor Networks, 2005*, pages 108 – 120, jan.-2 feb. 2005.

[99] Geoffrey Werner-Allen, Konrad Lorincz, Matt Welsh, Omar Marcillo, Jeff Johnson, Mario Ruiz, and Jonathan Lees. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10:18–25, 2006.

[100] Tim Winter (editor), Pascal Thubert (editor), Anders Brandt, Jonathan Hui, Richard. Kelsey, Philip Levis, Kris Pister, Rene Struik, J. P. Vasseur, and Roger Alexander. RPL: IPv6 Routing Protocol for Low power and Lossy Networks. RFC 6550, March 2012.

[101] Ning Xu, Sumit Rangwala, Krishna Kant Chintalapudi, Deepak Ganesan, Alan Broad, Ramesh Govindan, and Deborah Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 13–24, New York, NY, USA, 2004. ACM.

[102] M. Younis, A. Lalani, and M. Eltoweissy. Safe base-station repositioning in wireless sensor networks. In *Performance, Computing, and Communications Conference, 2006. IPCCC 2006. 25th IEEE International*, pages 8 pp. –528, apr. 2006.

[103] Yan Yu, Ramesh Govindan, and Deborah Estrin. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Technical report, Technical report ucla/csd-tr-01-0023, UCLA Computer Science Department, 2001.

[104] Marco Zúñiga Zamalloa and Bhaskar Krishnamachari. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Trans. Sen. Netw.*, 3(2), June 2007.

[105] Yuanyuan Zeng and Guilin Zheng. Delay-bounded and robust routing protocol for emergency applications using wireless sensor networks. In *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, volume 4, pages 37 –41, March 2010.

[106] Jerry Zhao and Ramesh Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 1–13, New York, NY, USA, 2003. ACM.

[107] Gouqing Zhou, Chaokui Li, and Penggen Cheng. Unmanned aerial vehicle (UAV) real-time video registration for forest fire monitoring. In *Geoscience and Remote Sensing Symposium, 2005. IGARSS'05. Proceedings. 2005 IEEE International*, volume 3, pages 1803–1806. IEEE, 2005.

Appendix A

Appendix

A.1 Publication

1. Ansar Jamil, David Parish, Iain Phillips, Raphael Phan, John Whitley and George Oikonomou. Extended Abstract: Maximise Unsafe Path Routing Protocol in Wireless Sensor Networks". *2nd Electronic and Electrical Engineering Departmental PhD Conference*, Loughborough University, UK, May 2011.

2. Ansar Jamil, David Parish, Iain Phillips, Raphael Phan, John Whitley and George Oikonomou. Designing Environmental Aware Routing in Wireless Sensor Network. London, UK, 2011. (Universiti Tun Hussein Onn Malaysia (UTHM) internal publication)

3. Ansar Jamil, David Parish, Iain Phillips, and Raphael Phan. Poster: Maximise Unsafe Path Routing Protocol in Wireless Sensor Networks. *School of Electronic & Engineering PhD Conference*, Loughborough University, UK, 2012.

4. Ansar Jamil, David Parish, Iain Phillips, Raphael Phan, John Whitley, and George Oikonomou. Maximise Unsafe Path Routing Protocol for Forest Fire Monitoring System using Wireless Sensor Networks. *The 3rd IEEE International Conference on Networked Embedded Systems for Every Application (NESEA 2012)*, Liverpool, UK, 2012.

A.2 Datasheet for CC2430

Not Recommended for New Designs

Chipcon Products from Texas Instruments

CC2430

A True System-on-Chip solution for 2.4 GHz IEEE 802.15.4 / ZigBee®

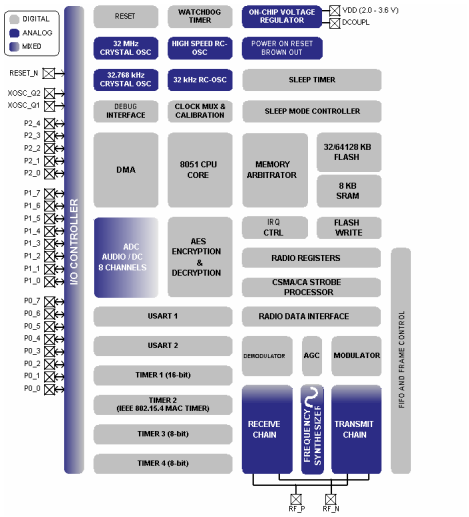
Applications

- *2.4 GHz IEEE 802.15.4 systems*
- *ZigBee® systems*
- *Home/building automation*
- *Industrial Control and Monitoring*
- *Low power wireless sensor networks*
- *PC peripherals*
- *Set-top boxes and remote controls*
- *Consumer Electronics*

Product Description

The ***CC2430*** comes in three different flash versions: CC2430F32/64/128, with 32/64/128 KB of flash memory respectively. The ***CC2430*** is a true System-on-Chip (SoC) solution specifically tailored for IEEE 802.15.4 and ZigBee® applications. It enables ZigBee® nodes to be built with very low total bill-of-material costs. The ***CC2430*** combines the excellent performance of the leading ***CC2420*** RF transceiver with an industry-standard enhanced 8051 MCU, 32/64/128 KB flash memory, 8 KB RAM and many other powerful features. Combined with the industry leading ZigBee® protocol stack (Z-Stack™) from Texas Instruments, the ***CC2430*** provides the market's most competitive ZigBee® solution.

The ***CC2430*** is highly suited for systems where ultra low power consumption is required. This is ensured by various operating modes. Short transition times between operating modes further ensure low power consumption.



Key Features

- **RF/Layout**
  - o 2.4 GHz IEEE 802.15.4 compliant RF transceiver (industry leading CC2420 radio core)
  - o Excellent receiver sensitivity and robustness to interferers
  - o Very few external components
  - o Only a single crystal needed for mesh network systems
  - o RoHS compliant 7x7mm QLP48 package
- **Low Power**
  - o Low current consumption (RX: 27 mA, TX: 27 mA, microcontroller running at 32 MHz)
  - o Only 0.5 µA current consumption in powerdown mode, where external interrupts or the RTC can wake up the system
  - o 0.3 µA current consumption in stand-by mode, where external interrupts can wake up the system
  - o Very fast transition times from low-power modes to active mode enables ultra low average power consumption in low dutycycle systems
  - o Wide supply voltage range (2.0V - 3.6V)
- **Microcontroller**
  - o High performance and low power 8051 microcontroller core
  - o 32, 64 or 128 KB in-system programmable flash
  - o 8 KB RAM, 4 KB with data retention in all power modes
  - o Powerful DMA functionality
  - o Watchdog timer
  - o One IEEE 802.15.4 MAC timer, one general 16-bit timer and two 8-bit timers
  - o Hardware debug support
- **Peripherals**
  - o CSMA/CA hardware support.
  - o Digital RSSI / LQI support
  - o Battery monitor and temperature sensor
  - o 12-bit ADC with up to eight inputs and configurable resolution
  - o AES security coprocessor
  - o Two powerful USARTs with support for several serial protocols
  - o 21 general I/O pins, two with 20mA sink/source capability
- **Development tools**
  - o Powerful and flexible development tools available

Figure A.1: Datasheet of CC2430

A.3 Making the simulation close to the experiment

Simulation is carried out to verify the experimental results in the large WSN test-bed. Even though the simulation does not provide the exact environmental conditions as in the experiment due to the lack of an accurate radio model, it is very important to at least ensure that this simulation is as close as possible to the experiment. Since Cooja is a simulator for Contiki OS, the same program code for the experiment can be used in the simulation directly, with all configurations the same as in the experiment, such as network topology, forest fire scenario, packet rate, transmit buffer size, channel check rate of duty cycle mechanism etc. For this reason, a network topology is established in the simulation environment consist of 25 nodes arranged in 5×5 grid topology. Because of the simulator does not support Sensinode platform, 'Sky' node is used as replacement in the simulation. UDGM with distance loss is selected as the radio model for each node in the network.

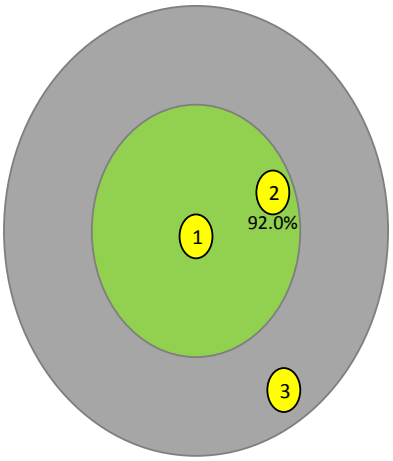


Figure A.2: UDGM radio medium model in Cooja simulator. The green circle denotes the good communication area of node 1 while the grey circle denotes the interference area. The percentage shows the reception ratio at node 2 for transmitted packets by node 1.

UDGM is a unit disk graph consisting of three areas: good communication area, interference area and unreachable area as shown in Figure A.2. Based on the transmission range measurement for a node in the section 6.6, the good communication area is set to be any point within a 6m radius. The interference area is set between 6 m to 8 m area. Over an 8 m radius is considered to be the unreachable area, which is outside the node communication range. This radio model supports the adjustment of link quality between two nodes by providing a plugin to set the transmission and reception ratio of these nodes. In this simulation, the

transmission ratio is set to 100% indicating no error during packet transmission. In order to represent the lossiness of a link in the WSN, the reception ratio setting can be adjusted. This reception ratio setting indicates the lowest value of packet reception ratio for a node located at the edge of the good communication area of a transmitting node. The reception ratio of a node varies depending on its distance from the transmitting node. A node closer to the transmitting node has a better reception ratio than another node located further away.

Figure A.3 shows the packet delivery ratio performance over different packet reception ratio settings for the RPL routing protocol. As expected, the packet delivery ratio performance is increased with the increment of the reception ratio value. This is because at a higher packet reception ratio, a transmitted packet has a higher chance of being received by the destination successfully compared to at a lower packet reception ratio. If we draw a line at the point where the packet delivery ratio is equal to the performance achieved in the experiment, which is about 80.8%, it is found that the reception ratio setting is about 67.5%. This reception setting can be used in the simulation to determine the performances of MUP-single and RPL. The simulation results can be compared with the experimental results.

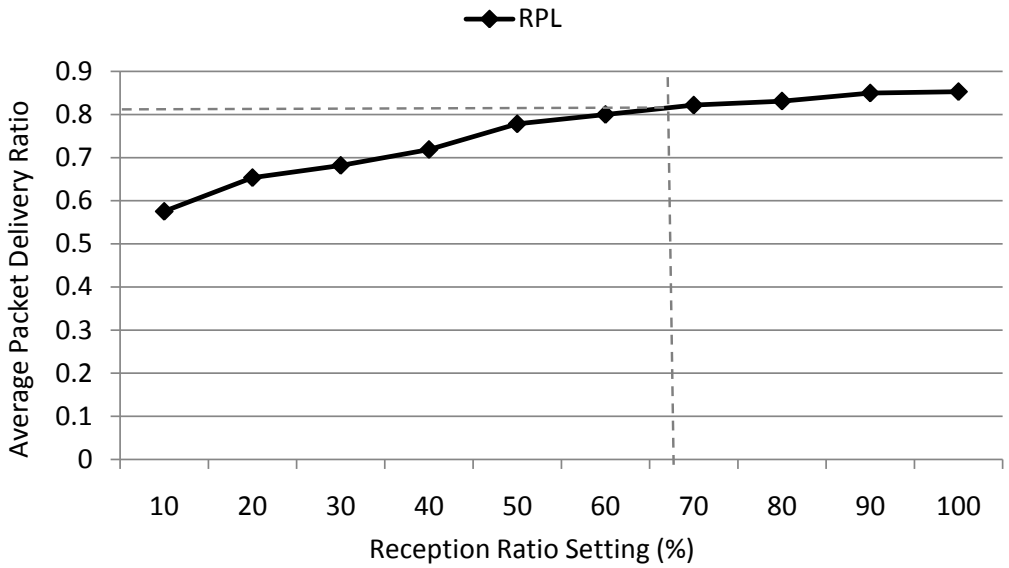


Figure A.3: Packet delivery ratio performance over different reception ratio settings for the RPL routing protocol.

A.4 Simulation results of remaining energy over time for four safe nodes located closest to the sink at different packet rates

A.4.1 Packet rate = 1 packet/second

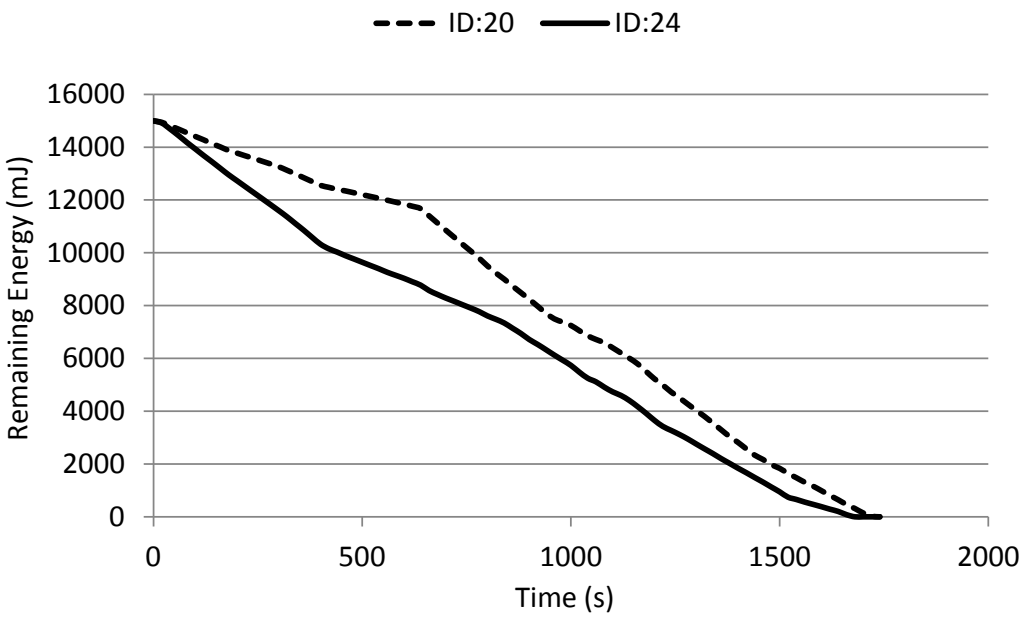


Figure A.4: Remaining energy of nodes 20 and 24 over time for MUP (MUP-single) at 1 packet/second in a forest fire situation.

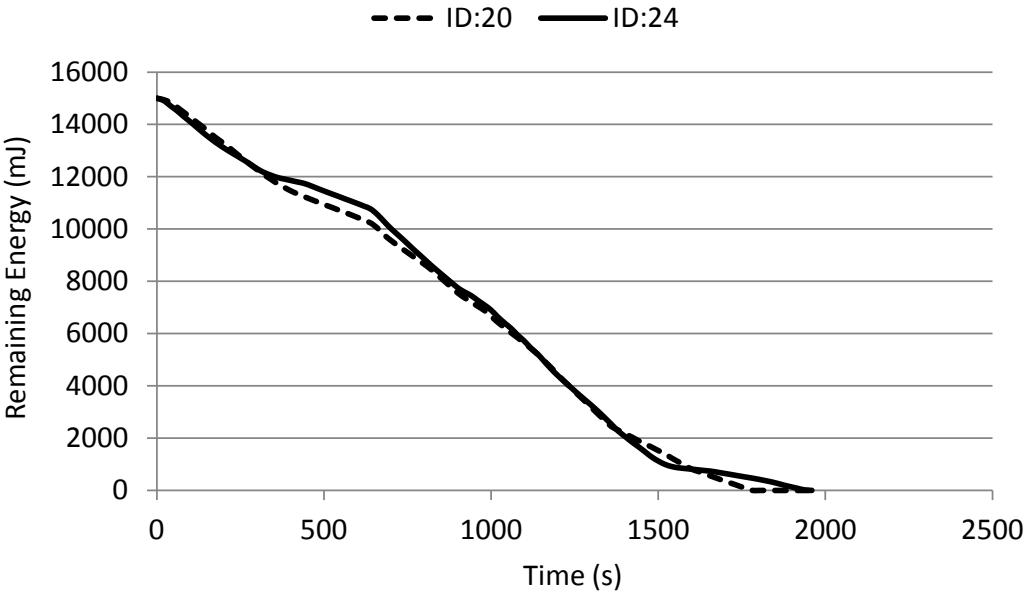


Figure A.5: Remaining energy of nodes 20 and 24 over time for MUP (MUP-adapt) at 1 packet/second in a forest fire situation.

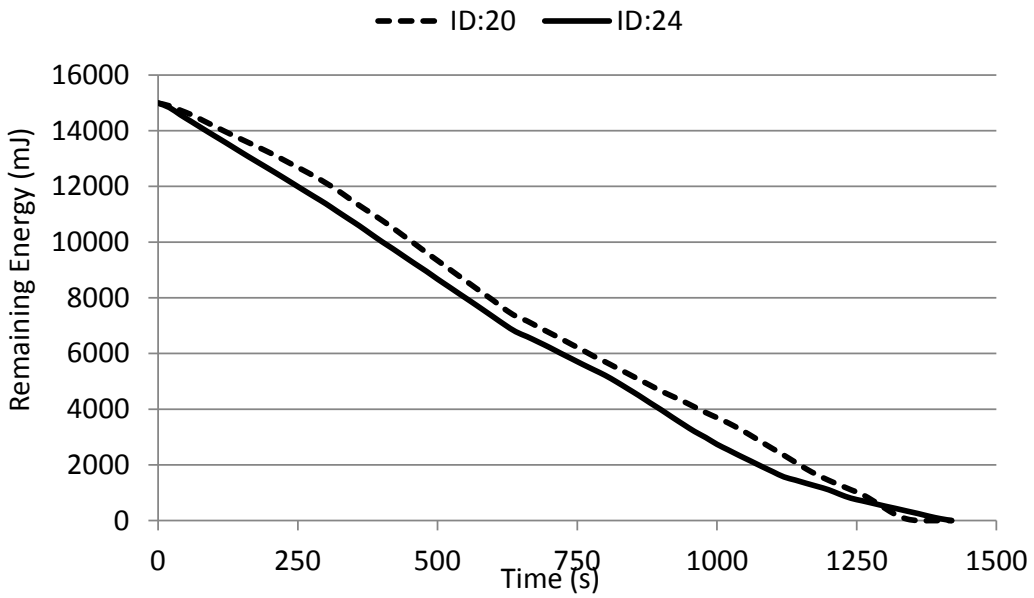


Figure A.6: Remaining energy of nodes 20 and 24 over time for SAFEST at 1 packet/second in a forest fire situation.

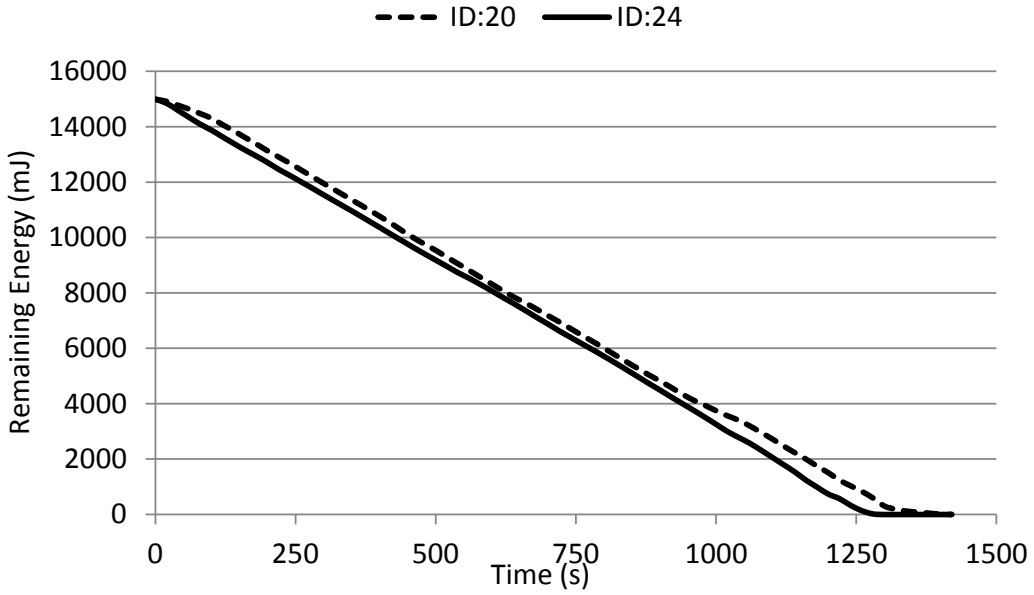


Figure A.7: Remaining energy of nodes 20 and 24 over time for RPL at 1 packet/second in a forest fire situation.

A.4.2 Packet rate = 0.5 packet/second

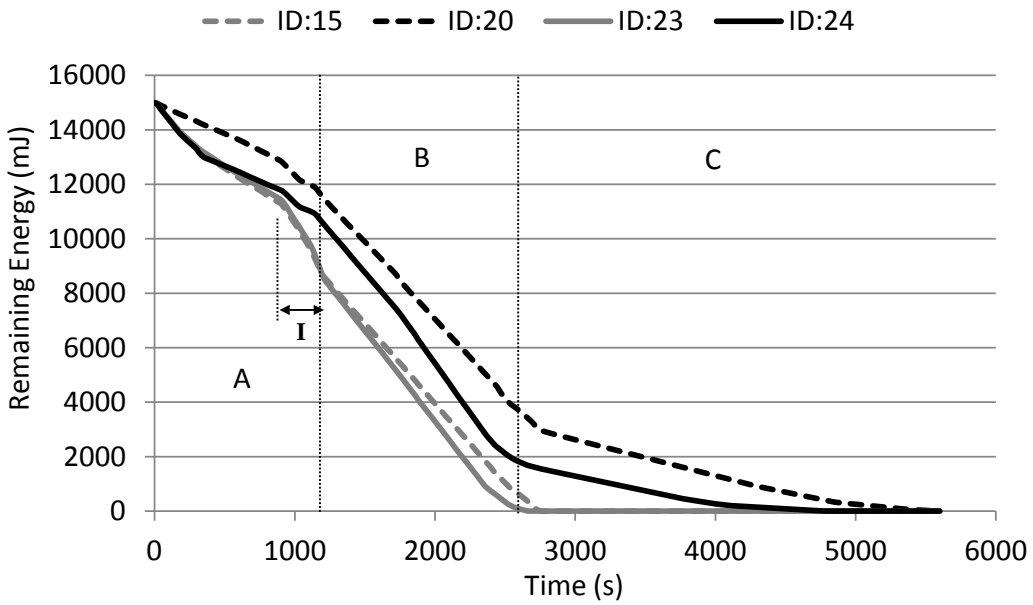


Figure A.8: Remaining energy of nodes 15, 20, 23 and 24 over time for MUP (MUP-single) at 0.5 packet/second in a forest fire situation.

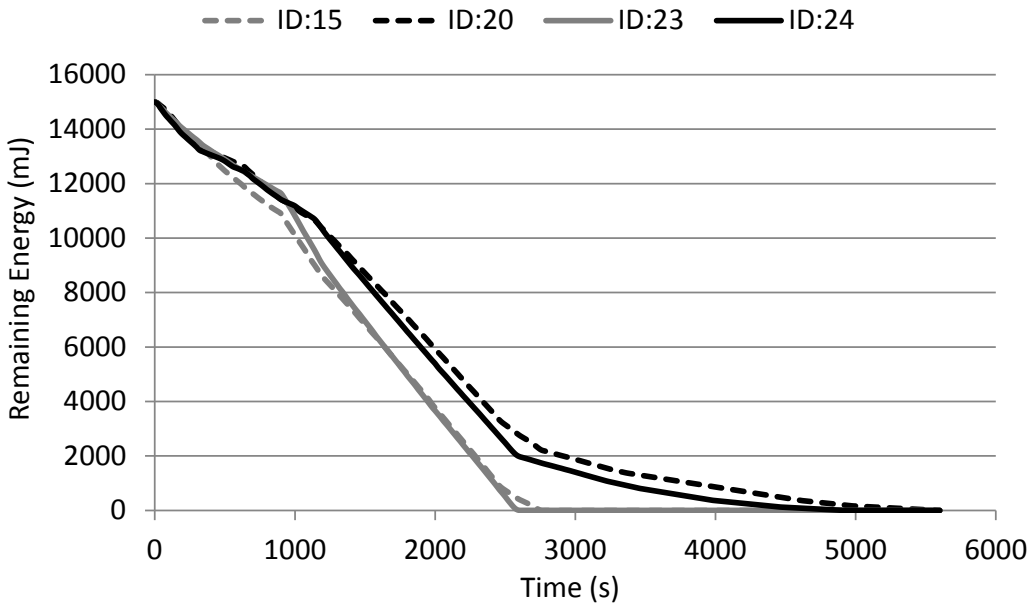


Figure A.9: Remaining energy of nodes 15, 20, 23 and 24 over time for MUP (MUP-adapt) at 0.5 packet/second in a forest fire situation.

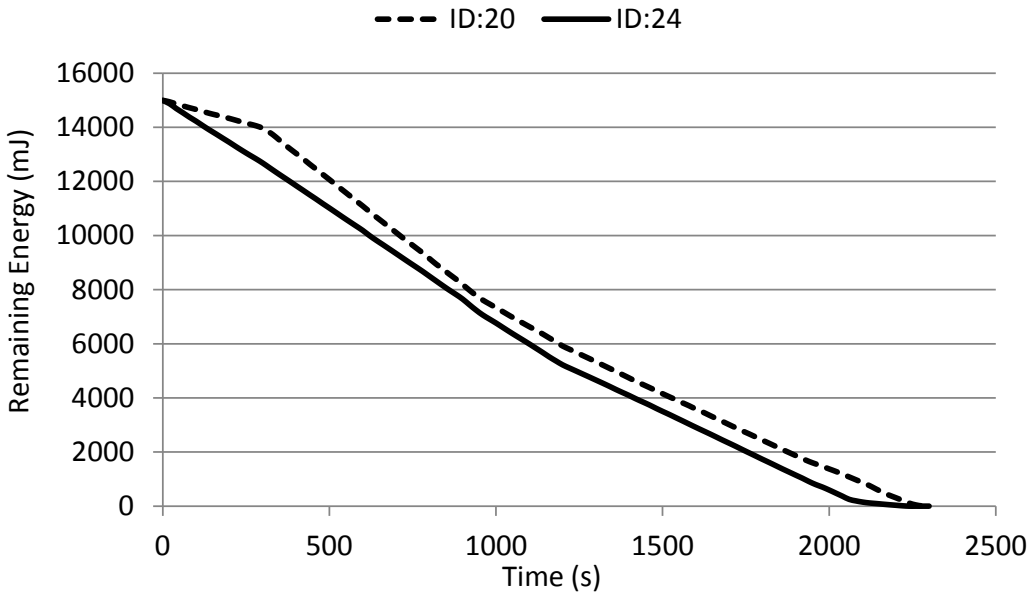


Figure A.10: Remaining energy of nodes 20 and 24 over time for SAFEST at 0.5 packet/second in a forest fire situation.

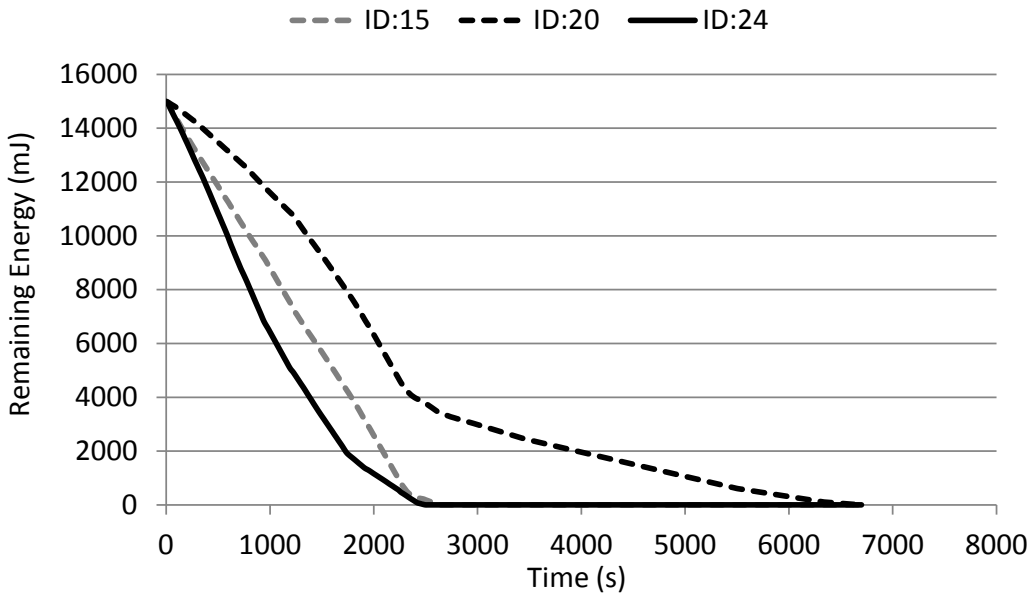


Figure A.11: Remaining energy of nodes 15, 23 and 24 over time for RPL at 0.5 packet/second in a forest fire situation.

A.4.3 Packet rate = 0.33 packet/second

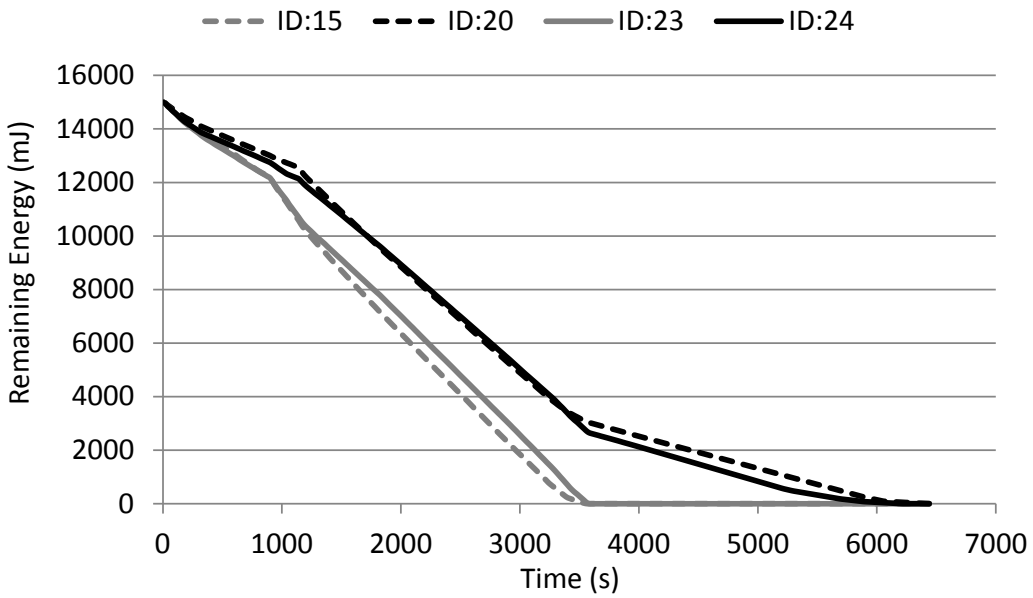


Figure A.12: Remaining energy of nodes 15, 20, 23 and 24 over time for MUP (MUP-single) at 0.33 packet/second in a forest fire situation.

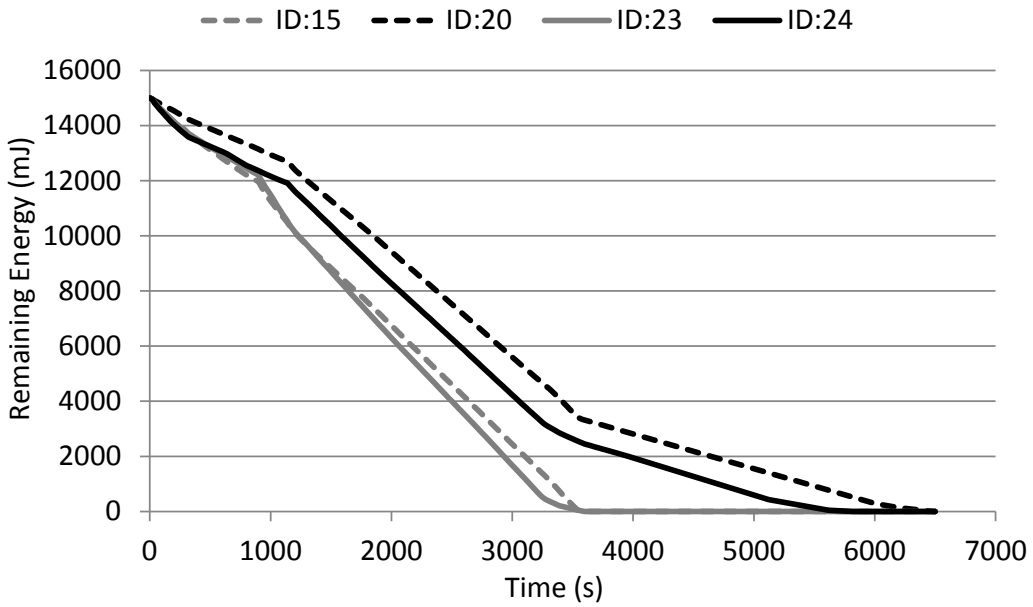


Figure A.13: Remaining energy of nodes 15, 20, 23 and 24 over time for MUP (MUP-adapt) at 0.33 packet/second in a forest fire situation.

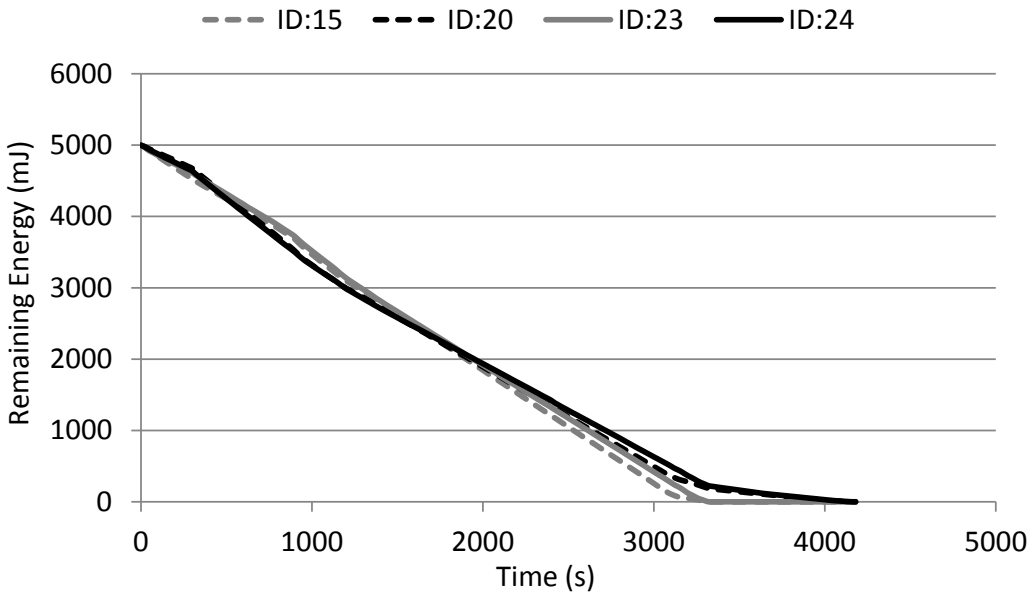


Figure A.14: Remaining energy of nodes 15, 20, 23 and 24 over time for SAFEST at 0.33 packet/second in a forest fire situation.

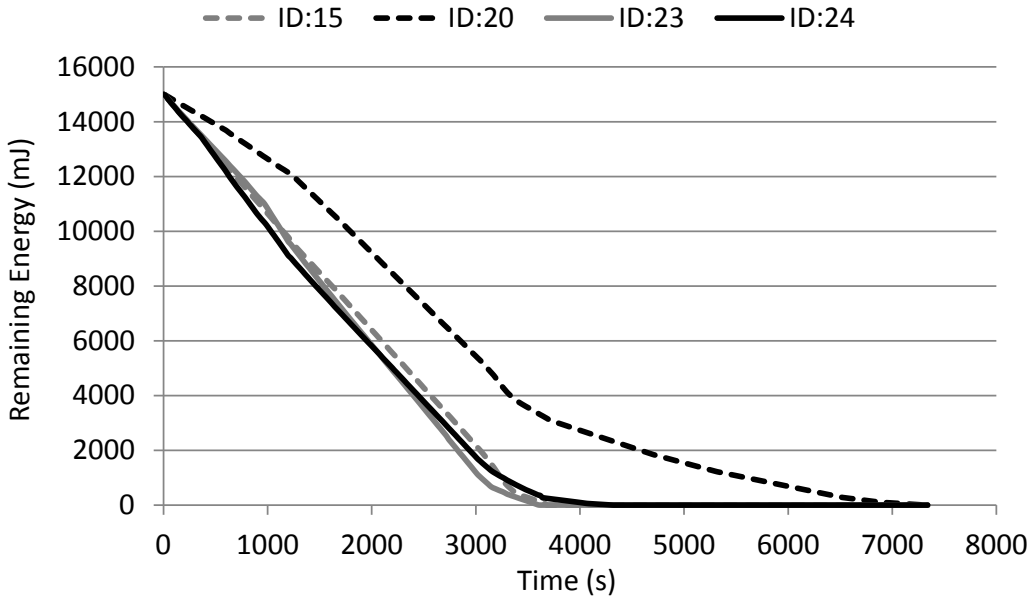


Figure A.15: Remaining energy of nodes 15, 20, 23 and 24 over time for RPL at 0.33 packet/second in a forest fire situation.

A.4.4 Packet rate = 0.25 packet/second

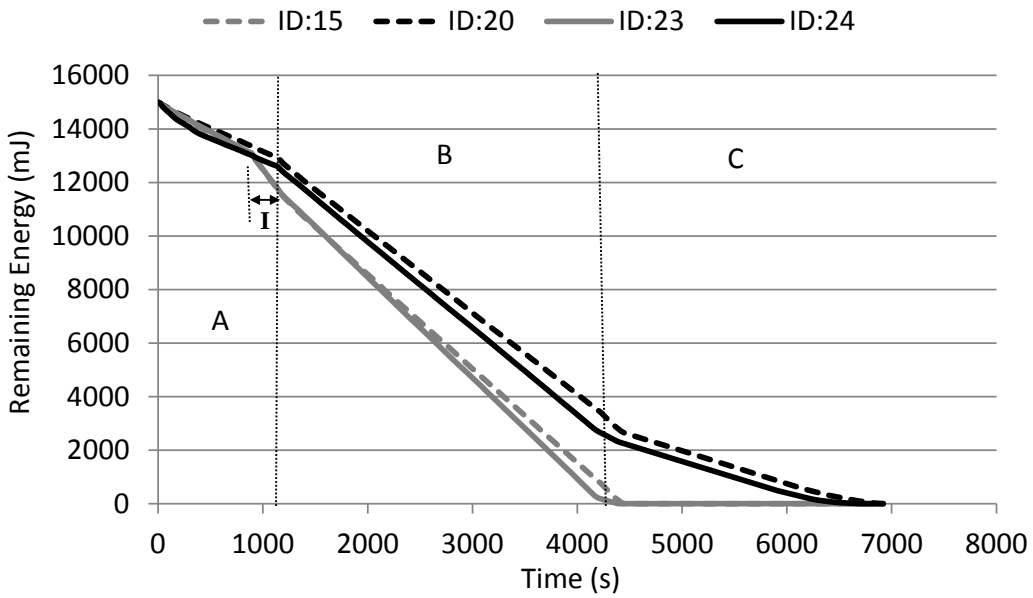


Figure A.16: Remaining energy of nodes 15, 20, 23 and 24 over time for MUP (MUP-single) at 0.25 packet/second in a forest fire situation.

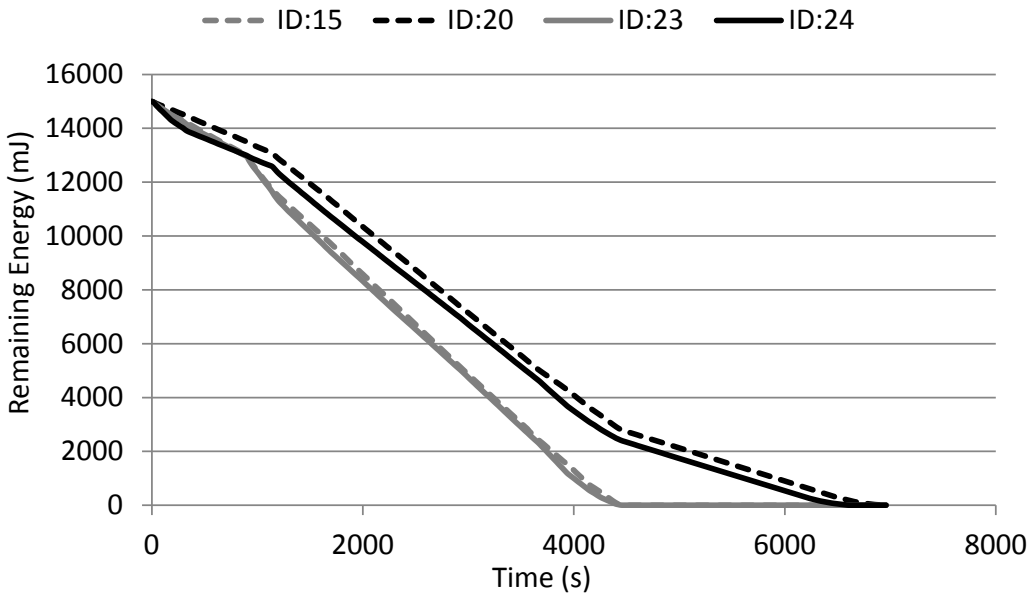


Figure A.17: Remaining energy of nodes 15, 20, 23 and 24 over time for MUP (MUP-adapt) at 0.25 packet/second in a forest fire situation.

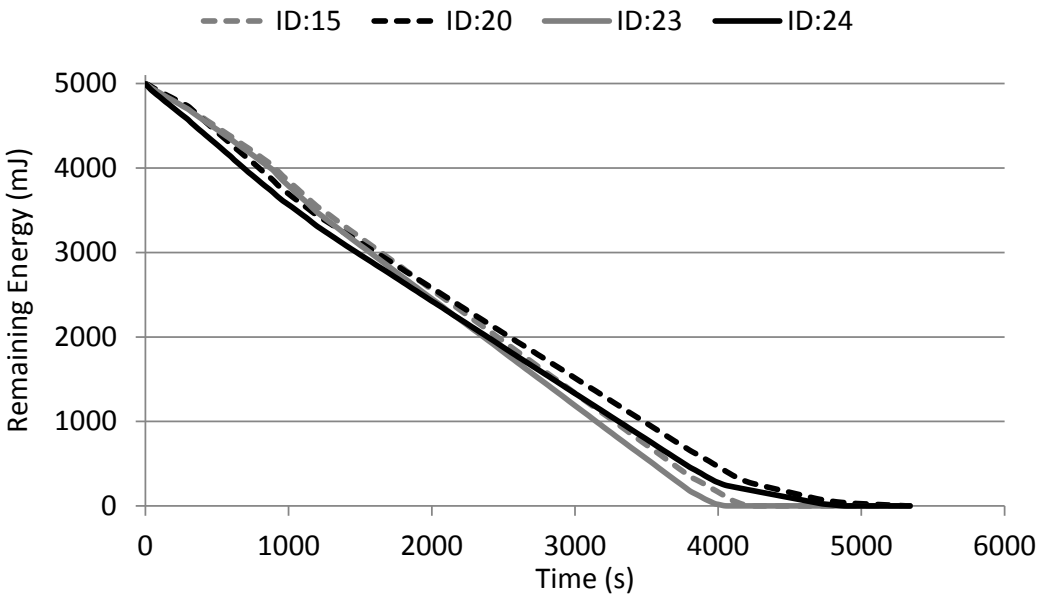


Figure A.18: Remaining energy of nodes 15, 20, 23 and 24 over time for SAFEST at 0.25 packet/second in a forest fire situation.

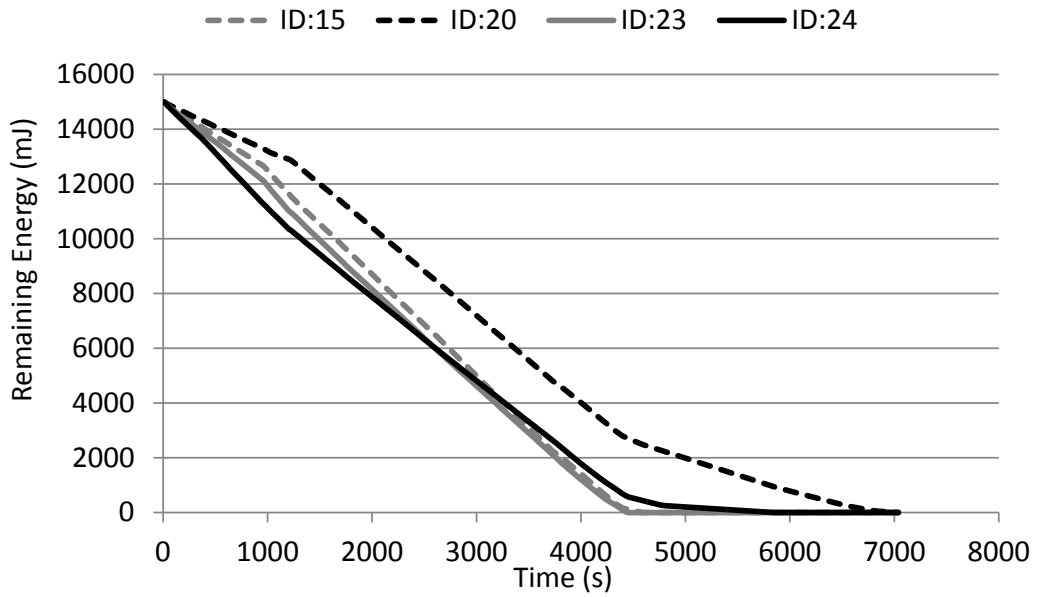


Figure A.19: Remaining energy of nodes 15, 20, 23 and 24 over time for RPL at 0.25 packet/second in a forest fire situation.