Transfer Thesis

Noradila Nordin

Department of Electronic & Electrical Engineering

University College London

Torrington Place

London

WC1E 7JE

noradila.nordin.12@ucl.ac.uk

January 31, 2016

Abstract

This report presents the current state of research work that has been carried out in the context of the PhD. (describe why do it). The PhD work proposes a new decentralised multi-channel tree building protocol with a centralised controller for ad-hoc sensor networks. The protocol alleviates the effect of interference which results in improved network efficiency and stability, and link reliability. The proposed protocol takes into account all available channels to utilise the spectrum and aims to use the spectrum efficiently by transmitting on several channels. The protocol detects which channels suffer interference and changes away from those channels. The algorithm for channel selection is a two-hop colouring protocol that reduces the chances of nearby nodes to transmit on the same channel. All nodes are battery operated except for the low power border router (LPBR). This enables a centralised channel switching process at the LPBR. The protocol is built based on the routing protocol for low power and lossy networks (RPL). In its initial phase, the protocol uses RPL's standard topology formation to create an initial working topology and then seeks to improve this topology by switching channels. The report discusses the main engineering and research challenges raised by the protocol, and describes and explains the principles and mechanisms used to support the proposed protocol. It then presents an extensive evaluation of the protocol and other other approaches. The implementation and evaluation of the protocol is performed using the Contiki framework. The report then describes the future main research issues that will be investigated in the context of this PhD.

In this report,

The proposed approach The report discusses

Acknowledgements

Acknowledge all the things!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context and Motivation

Wireless Sensor Networks (WSN) are ad-hoc networks that consist of sensor nodes that typically use low power radios such as IEEE 802.15.4, a relatively short range transmission standard radio technology in the 2.4 GHz band. The standard allows transmission to occur on several different channels within this band [1]. Unfortunately, the channels used by this technology often suffer interference [2, 3], for example, from Wi-Fi [4, 5] and Bluetooth. Sensor networks have to contend with an increasing number of devices that cause this wireless interference. Organising the network topology around this interference becomes an enabler for increasing transmission efficiency at a smaller energy cost. WSNs need to be able to operate reliably in the presence of such interference. It is important to minimise energy costs in these networks since deployments can be for weeks, months or longer.

Multichannel communication in wireless networks can alleviate the effects of interference which, as a result, can improve the network efficiency and stability, link reliability and minimise latency [6]. It also enables communication between physically proximate nodes to occur simultaneously without the risk of collision when the communicating nodes use different channels. However, not all channels are free from interference; thus, there is a gain to hop to another channel when the quality of the channel deteriorates. Two commonly used types of channel hopping [6] are blind channel hopping and whitelisting. In blind channel hopping, nodes

choose from all available channels. Whitelisting, on the other hand, gives a set list of channels that avoids those that are known to commonly suffer interference. Many studies make use of channel whitelisting such as in Chrysso [7] and MiCMAC [8].

Note that potentially Chrysso and MiCMAC could use all available channels. However, they do not have a mechanism to check the channel condition before using it for packet transmission. MiCMAC sees its performance degraded when using more than 4 channels, thus the decision on specifying 4 channels to be included in their experiment. MiCMAC uses a different channel chosen at random each time it wakes up. It might require several wake up periods which is time consuming, before a clear channel is found from the 16 channels, to deliver the packet. Chrysso on the other hand, switches the affected nodes to a new set of channels upon detecting interference which entails frequent channel switching if all channels are to be considered.

1.2 Problem Statement

It is clear that it is impossible to find a single channel guaranteed free from interference and there is no consensus on the best channel to use. Our work takes into account all available channels to utilise the spectrum and checks the condition of the channels before hopping to avoid those channels with interference. Several previous studies have developed a multichannel MAC layer but, despite the potential benefits none are yet widely implemented in real world deployments.

1.3 Contribution

Important aspects of this work will be investigating lossy multichannel. Designing the protocol for multichannel raises several research challenges. The decision making process is (centralized and decentralized explain here). The main benefits of this approach are (). The work in this PhD will address the issues raised by (). More specifically it will investigate how the channel selection is determined from the nodes interactions. In addition, it will investigate the cross layer interaction.

1.4 Current Work

In the context of the research efforts carried out from the beginning of the PhD research work, the followings have been investigated. A new multi-channel protocol called Multichannel Cross-Layer Routing Protocol (MCRP) has been developed. The proposed approach is so that the nodes are able to communicate on many channels in order to avoid interference and channel congestion in a centralized and decentralized manner. This paper presents a Multichannel Cross-Layer Routing Protocol (MCRP) which consists of two main parts; a centralised intelligence at LPBR, and decentralised nodes. LPBR implements a two-hop colouring algorithm to avoid interference between physically proximate nodes trying to communicate on the same channel. The information on channel interference and network topology from the lower layer is made available to the application layer. This allows the centralised controller (LPBR) to have an overall view of the system to make decisions at the network and MAC layers about which channels nodes should listen on. The systemis fail safe in the sense that the WSN functions if the central system which assigns channels fails temporarily or permanently. We implement MCRP in Contiki [9], an open source operating system for WSNs and evaluate the protocol in Contiki network simulator, Cooja [10]. We demonstrate that MCRP avoids channels with interference which greatly reduces the effects of interference on the network. The performance of the approach has been evaluated using (). The evaluation has been performed with respect to the ().s

1.5 Report Outline

The remainder of the report is organised as follows. Chapter 2 introduces the state-of-the art in the area of multichannel protocols. It also presents the main current research efforts towards () Section **??** presents related work to multichannel protocols. Chapter 3 presents the main features and mechanisms used in MCRP. It describes (). It also presents (). Section **??** describes the key idea of our proposed protocol and the high-level design, and the implementation of the protocol in Contiki. We describe and evaluate the experimental results in Section **??**. Chapter 7

summarises the current work and presents the future research works that will be investigated in the context of this PhD.

Chapter 2

Literature Review

2.1 Wireless Sensor Networks

A WSN is a network of sensor nodes that are used to collect data from the target area over the radio. These data that the sensors send can be normal data packets or sensor measurements data such as the temperature and movement in the specific area where the sensors are located. The sensors can be used for continuous sensing, event detection, location sensing and local control of actuators to control different components in the sensing device, such as adjusting the sensor parameters or move the sensor if it is a mobile sensor.

This chapter describes the available WSN applications, challenges and known issues that occur in WSNs. Many previous studies were done in order to maximise the lifetime of sensor networks while keeping the energy to a minimum. This chapter also briefly describes the existing solutions for energy efficient multi channel at the MAC and network layers which prompted to the work of MCRP.

2.1.1 Applications Overview

There are five types of deployed WSNs: terrestrial WSNs, underground WSNs, underwater WSNs, multimedia WSNs and mobile WSNs which cover different types of environment; to deploy on land, underground and water [11]. Unlike other sensor nodes, multimedia WSNs have the ability to monitor and track events in the form of video and audio as they are equipped with cameras and microphones for multimedia data which can enhance the existing WSN applications [12]. Mobile WSNs

on the other hand, can be any type of sensors that have the capability to reposition and organise itself in the network.

WSNs evolution is driven by a number of emerging applications that focuses on the importance of wireless sensors in applications such as smart grid, areas in smart cities, and automated home, building and industrial applications [13]. Smart grid could save considerable amounts of energy by improving the existing electrical grid power. Smart cities which includes automated home, building and industrial in populated cities can improve the environment quality by allowing automated services such as pollution monitoring and automated energy control (temperature and lighting) which increases the energy saving in the process.

WSNs applications are important as sensor nodes can be easily deployed at all types of environment, installed and require minimal maintenance for a period of time. The main challenges in these applications are in term of reliable event detection, securing high data rates for efficient data routing and dense or sparse nodes deployment.

WSNs applications can be categorised into five main monitoring and tracking applications which are the environmental applications, health applications, home applications, military applications and other commercial applications [14]. These applications are briefly described in the next section with examples for each category.

2.1.1.1 Environmental Applications

The environment applications can be divided into two types; tracking and monitoring. The tracking applications are used to record the movements of animals such as birds, insects and small animals at a certain area. Monitoring applications are used to monitor the environment conditions such as forest fire detection, flood detection, biocomplexity mapping [15], precision agriculture monitoring and volcanic monitoring [16].

In forest fire detection, the sensor nodes are used to relay the exact originated location of the fire to the end users to control it from spreading. ALERT is an example of a flood detection system that is deployed in the United States. ALERT

consists of several types of sensors such as rainfall, water level and weather sensors. In agriculture, the sensor nodes are used to monitor the level of pesticides in drinking water, soil erosion and air pollution in real time. In volcanic monitoring, the sensor nodes allow measurements to be taken from locations that are otherwise inaccessible.

2.1.1.2 Health Applications

Sensor networks in health applications can be used to monitor human physiological data such as detecting elderly people's behaviour in case of a fall, drug administration [17] in hospitals to minimise incorrect prescription of medication to patients, and to monitor and track doctors and patients locations in a hospital. Examples of these are *telecare* and *telehealth* [18].

Telecare is a system of wireless sensors that are placed around the house and can be a personal alarm in form of a small wristband or pendant. These sensors can detect risks such as a fall, motion sensor that turns on the lights at night when someone get out of bed, a pressure mat on the mattress to sense if someone gets back to bed or a sensor on the door in case it is not closed, are a few examples of the system. If a risk is detected, it sends the alert immediately for attention to a telecare monitoring centre.

Telehealth is a small equipment to monitor health from home. It can be used to measure the blood pressure, blood glucose levels, oxygen levels, weight or temperature. The measurements are automatically transmitted to a monitoring centre. The healthcare professional will be contacted if the information raised an alarm for actions to be taken.

2.1.1.3 Home Applications

In home automation [19], the smart sensor nodes and actuators can be buried in the appliances such as vacuum cleaners, microwave ovens and refrigerators which allows them to form an interaction through the Internet. Some of the recent home automation are *Samsung SmartThings* and *Nest Thermostat*.

Samsung SmartThings allows devices at home to be monitored and controlled from a mobile phone such as controlling the thermostats and lighting. Nest Ther-

mostat is a self-learning thermostat that is consists of activity sensors, temperature sensors, humidity sensor and a Wi-Fi radio. These sensors allow Nest to learn the heating and cooling habits which allows it to shuts down due to inactivity to conserve the energy. Nest is weather aware. It uses its Wi-Fi connection to get the weather condition and forecasts, and integrate the information to understand the affects of the outside temperature to the energy usage. Nest is also able to connect with other appliances that are Nest supported. The appliances can automatically start without any need to program it as it learns from other devices.

2.1.1.4 Military Applications

WSNs are used in military applications to monitor friendly forces, equipments and ammunitions by attaching sensors which report the status back to the base station; battlefield surveillance by covering critical terrains, routes, paths and straits with sensors and reconnaissance the opposing forces; assess battle damage, and to detect nuclear, biological and chemical attack by deploying sensors to explore areas and serve as warning systems to avoid casualties.

An example of military application is *PinPtr* [20]. PinPtr is an experimental counter-sniper system. It was developed to detect and locate shooters by measuring shot time of arrival of the muzzle blasts and shock waves from the sensors that are densely deployed. The measurements are routed to the base station where the shooter's location is computed. PinPtr was demonstrated and evaluated in realistic urban environment from various US Army test facilities.

2.1.1.5 Other Commercial Applications

Other available commercial applications are environmental control in office buildings such as controlling the air flow and temperature for different part of the building; car thefts monitoring and detection within specific region; inventory control management to track and locate the inventories in the warehouses; machine diagnosis in order to predict equipment failure for maintenance through vibration signatures gathered by sensors [21]; and vehicle tracking and detection for parking purposes such as the *Smart Parking* from *Streetline* and *SmartPark*.

Smart Parking solutions are used in more than 40 cities and universities in

North America and Europe. The system could make intelligent decisions using the data from the real time and historical analytical reports to improve the parking ecosystem. The system detects vehicle occupancy in real time which simplifies the parking experience by guiding drivers to the available spaces. It can also guide officers to unpaid violations and overstays as the arrival and departure times are recorded; and to detect if a car is parked over the no parking and restricted zones.

SmartPark is a parking solution in the UK, currently operating in Birmingham and in the central London Borough of Westminster. These applications enable drivers to find vacant space within the busy town and city centres quicker.

2.1.2 WSN Challenges and Issues

WSNs are widely used in various kinds of applications. This is because sensor nodes can be densely deployed, easy to install and require minimal maintenance over a period of time. However, WSNs suffer from limited hardware resources which only allow limited computational functionalities to be performed. It also suffers from limited energy capacities as the sensors are battery powered and they will become faulty and not able to function once the certain threshold of energy level is reached. It also operates in an unreliable radio environment that is noisy and error prone which drain the sensors batteries at a higher rate.

These constraints have a major impact on the sensors performance. In order to prolong the sensors lifetime thus, the network lifetime, the sensors need to be able to cope with the limitations and be as energy-efficient as possible to guarantee good overall performance.

2.2 Maximising Lifetime and Minimising Energy

In WSNs, it is necessary to estimate the nodes power consumption before they are deployed to enable accurate forecast of the energy consumption. The estimations are used to determine the nodes lifetime before maintenance and batteries replacements are required in order to have a functional network. Unfortunately, the node lifetime is very dependent on the radio environment that can be unstable, noisy and error prone which makes energy consumption to vary [22]. The network lifetime,

however, depends on various factors such as the network architecture and protocols, channel characteristics, energy consumption model and the network lifetime definition. In order to increase the network lifetime, these information regarding the channel and residual energy of the sensors should be exploited.

There are various definitions of network lifetime that have been used. These definitions are application-specific as some applications might tolerate a considerable number of loss nodes, while some applications require a higher number of nodes which any loss is considered critical to the network such as in sparsely deployed nodes of an area. The definitions impact the performance differently, depending on the applications. The various definitions are:

- **The first node to die** - The network lifetime is defined as the first node to fail in the network [23, 24]. In [25], the simulation ends when a node reaches the energy level of zero.
- **The number of alive nodes** - The network lifetime is the number of remaining nodes as a function of time. The network has a longer lifetime with a higher number of remaining nodes [26, 27].
- **The number of nodes still connected to the sink** - The network is alive based on the remaining number of nodes to have coverage to connect to the sink [27].
- **The fraction of alive nodes** - The network lifetime is defined by the percentage of surviving nodes above a threshold.
- **Packet delivery ratio** - The network lifetime ends when the packet delivery ratio drops dramatically. GAF [28] uses this definition where it is possible when the traffic is kept constant.
- **The first failure in data transmission** - The sensor does not have enough energy for transmission [23].

Network lifetime is strongly related to the remaining energy of all nodes. However, maximising the minimal energy of all the nodes is not the best way to prolong

the network lifetime as it will place heavy burden to the key nodes such that nodes that are close to the sink. These nodes drain their batteries quicker than other nodes which as a result, shorten the network lifetime.

There are four ways that have been explored from many studies to maximise the network lifetime, which are by introducing (i) energy efficient MAC protocols, (ii) energy efficient routing protocols, (iii) controlling the transmission power and (iv) using energy harvesting. These options are described in details in the next few sections, introducing the differences and advantages, and the existing proposed solutions.

The aim of the WSN design is to extend the network lifetime under the given energy and node constraints without jeopardizing reliability and communications efficiency of the network.

2.2.1 MAC Protocols

Many energy efficient MAC protocols have been proposed to prolong the network lifetime. The radio module that is controlled by the MAC protocol is the major energy consumer in WSNs. The radio uses nearly the same energy in all active operation modes such as the transmit, receive and idle modes [22]. Thus, it is important to reduce the radio usage to conserve the nodes energy.

The main causes of energy consumption are nodes collision, overhearing and idle listening [29, 30]. Collision happens when nodes that is within each other transmission range transmits simultaneously. The energy used in the collided transmissions is wasted as none of the nodes would receive the transmitted packet. Multi channel is one of the solutions to overcome collision. Overhearing happens when a node receives irrelevant packets or signals that are not intended to the node. As the radio uses nearly the same energy for all operations, this drains the node energy unnecessarily. In idle listening, the node keeps its radio on while listening to the channel for potential packets. The node does not know when it will be the receiver of the packet. Considerable amounts of energy are wasted as the node keeps its radio on for a longer period listening to an idle channel when it does not receive or transmit packets.

A vast number of energy efficient MAC protocols have been developed to overcome these problems through *duty cycling*. Duty cycled MAC protocols allow the node to periodically alters the sleep state and listen state. By lowering the duty cycle, the node sleeps for a longer period instead of being permanently active. However, the node needs to have frequent check interval to avoid deafness problem while keeping overhearing to a minimum. This reduces the energy consumed by idle listening and overhearing.

Many MAC protocols such as YMAC [31] uses duty cycle as the indicator to evaluate the energy efficiency performance. This is because it is difficult to measure the nodes energy consumption. However, there are studies that managed to estimate the energy consumption. This is described in Chapter 6.

2.2.2 Routing Protocols

Various energy efficient routing protocols for WSNs have been proposed and developed to ensure efficient packet delivery to the destination. The strategies that are used in routing protocols should ensure minimum energy consumption in order to prolong the lifetime of the network.

A major issue in WSNs routing protocol is in finding and maintaining the optimal routes that are energy efficient. This is due to the energy constraints and unexpected changes in node status such as node failure or unreachable. This causes the topology to be altered frequently to adapt to the changes. Abrupt topology modification is important to avoid the network from being disconnected which leads to higher rate of packet loss at the involved nodes as the routes are not updated.

There a several routing techniques such as flat, hierarchical and location-based routing protocols that are application dependent. Hierarchical structure, as an example, has a balanced energy structure as the packets are transmitted from the lower layer nodes to the upper layer nodes. These different techniques are explained in detail in Section 2.4.

The routes that are formed are based on the routing metric [32] that attempts to transmit the packet to the receiver by selecting the most efficient path that the protocol calculated. The path may be the shortest path, lowest expected transmis-

sion count path [33] or path that maximises the network lifetime by considering all nodes remaining energy. However, in order to achieve the best network lifetime, the total energy consumption of the network and the nodes minimal remaining energy should be combined for a better balance in the network [24].

2.2.3 Transmission Power Control

Topology control term has been used to mean two different things in WSNs literature. Several authors define topology control as routing protocol techniques. Another definition of topology control is power control techniques which act on the nodes transmission power level [35]. Topology control term has been interchangeably used with power control. To avoid confusion, the term power control is used in this thesis.

In power control, a node has control over the transmission range of the node's radio which can be manipulated to benefit the network. The power adjustment approach allows the node to vary the transmission power thus range to form a connected network that minimise the energy incurred in transmission. The nodes collaboratively adjust to find the appropriate transmission power which enables the nodes to transmit at a lower transmission power than at the maximum. However, a sparse network would require a higher transmission power than a dense network to be able to transmit to the nearest node.

The power control technique eliminates links that are wasting the energy resources by fixing the area of coverage thus routing. This reduces collisions as inefficient links of long distance nodes are discarded. However, the nodes need to change the transmission power to adapt to any area coverage changes in order to modify the routing.

As the transmission ranges are relatively short, the nodes can simultaneously transmit packet without interfering each other, thus reducing congestion from retransmissions. Although power control improves the network traffic flows, it does not reduce the nodes power consumption as it depends on the radio duty cycle. Power savings due to transmission powers are negligible [29].

2.2.4 Energy Harvesting

As mentioned previously, sensor nodes have limited energy capacities as they are battery powered. However, the number of deployed nodes within the specific area has an effect to the nodes energy usage. In a densely deployed nodes area, short range transmission between the nodes could reduce the energy consumption while a sparsely deployed nodes area have a longer range transmission which require higher energy usage. In the situation where the nodes are not densely deployed, energy harvesting may be an option to increase the nodes energy level.

Energy harvesting is when a node tries to replenish its energy by using other energy sources such as solar cells [89, 90], vibration [91], fuel cells, acoustic noise and a mobile supplier [11]. Solar cell is the current mature technique to harvest energy from light. There is also work in using robots as mobile energy supplier to deliver energy to nodes. This allows a longer network lifetime as the node has restored its energy.

However, energy harvesting depends on various environment factors such as light, vibration and heat to be generated and converted to the usable electrical energy. There are also other different powering mechanisms that are available such as rechargeable battery with regular recharging from the sunlight [29].

2.3 Multichannel MAC Protocol

In single channel MAC protocols, nodes are configured to use a single channel throughout the nodes lifetime. Frequency agile MAC protocols on the other hand, allow the nodes to switch to different channels during run time. This is possible as recent radio chips take less than 100μs to switch to a different channel. The channel switching delay is negligible which attracts multi channels to be used in WSNs. Multi channels have the advantage of an increase in robustness against external and between nodes interference which as a result, improves the network traffic flow.

2.3.1 Introduction

There have been many proposals in multichannel communication which uses the duty cycling technique to alter the nodes sleep and listen states. The duty cycling is

an important mechanism that helps reducing the nodes energy consumption. However, adjusting the duty cycle does not solve the interference problem as external interference is unpredictable. Multi channel is a preferable solution to improved resilience against interference.

Existing duty cycled multichannel MAC protocols can be categorised into two types; synchronous and asynchronous systems. These are also referred as reservation-based protocol and contention-based protocol by some authors. A synchronous system is a system that requires a tight time synchronisation between nodes. It uses time-scheduled communication where the network clock needs to be periodically synchronised to compensate for time synchronisation error in order for the nodes not to drift in time [31]. The system requires dependency on the time synchronisation and network topology. The knowledge of the network topology is required to be able to establish a schedule for the nodes to access the channel to communicate with the other nodes.

Asynchronous system on the other hand, does not require synchronisation and topology knowledge but instead is a sender or receiver initiated communication. The nodes compete to access the channel to transmit such that the node postpones its transmission if it senses that the channel is busy, by sending preamble packets, to avoid interfering with the current transmission. In asynchronous systems the nodes are able to self-configure without time synchronization and this can have advantages. There are many studies done in multichannel for both categories.

Multichannel communications have potential benefits for wireless networks that include improved resilience against external interference, reduced latency, enhanced reception rate and increased throughput. A set of existing multichannel MAC protocols are reviewed and compared, highlighting their features and limitations.

2.3.2 Synchronous Systems

In synchronous systems, the multichannel MAC protocols employs time division multiple access (TDMA). It allows the channel to be divided into different time slot. TDMA-based MAC protocols allocate time slots to the nodes for data transmission

or reception [31]. This helps to avoid collision between nodes during transmission as the nodes have their own time slot. However, it has a higher latency as the node has to wait to its assigned slot before it is able to transmit a packet.

TSCH [36], MC-LMAC [38] and YMAC [31] are a few examples of the existing synchronous systems. These multichannel MAC protocols are selected for review.

2.3.2.1 TSCH

The Timeslotted Channel Hopping (TSCH) [36] is MAC protocol that uses time synchronisation and channel hopping to increase reliability in the network. The nodes in TSCH are fully synchronised. The nodes are assumed to be equipped with clocks as the nodes need to maintain tight synchronisation. The clocks in different nodes could drift in time, thus the nodes need to periodically resynchronise its clock with the time-source neighbour in the absence of data to transmit. The nodes also provide their time during synchronisation to the neighbours. When the nodes have data to send, the timing information is added to the packet which simplifies the synchronisation process as the nodes are resynchronise each time they exchange data.

It is designed for optimisation, customisation and it simplifies the process of merging TSCH with protocol stack based on IPv6, 6LoWPAN and RPL. TSCH defines the mechanism to set up the schedule and control the resources allocation to each link in the network topology for execution. It also defines the mechanism that signals when a node cannot accept an incoming packet. However, it does not define when the node should stop accepting packets.

In TSCH, time is sliced up into time slots that are appropriate for the traffic flow size. The time slot is set to be long enough to enable the sender node to send a maximum size of MAC frame to the receiver node and for the receiver to send an acknowledgement (ACK) frame to notify the sender that the frame has been successfully received.

Figure //// shows the TSCH schedule and terminologies used.

Slotframes contains a group of time slots of equal length and priority where the

slotframe repeats continuously over time. The size of the slotframe is not appointed by TSCH. Shorter slotframe has the advantage of more available bandwidth as the result of frequent repetition of the same time slot but at the cost of higher power consumption.

A single element in the TSCH schedule is called as a *cell*. The cell can instruct the node to transmit, receive or sleep. It can also be marked as both transmitting and receiving. However, transmission takes precedence over reception. The TSCH schedule also indicates the channel and address of the node for communication. The channel in TSCH is referred as *channelOffset*, which is the row in the TSCH slotframe. In a transmit cell, the outgoing buffer is checked for a packet that matches the scheduled neighbour for that time slot. Similarly, in a receive cell, the node listens during the reserve cell for possible incoming packets. Each scheduled cell is dedicated for the node. However, a cell can be shared where multiple nodes can transmit on the same frequency at the same time. TSCH defines a backoff algorithm to avoid transmissions from nodes in the shared cells from congesting the network.

Absolute Slot Number (ASN) is a timeslot counter in TSCH that calculates the communication frequency for the sender and receiver nodes. The calculation from ASN and *channelOffset* is translated into a different frequency at different slotframe cycles. The ASN value changes at the next iteration which results in a different frequency computed for the cycle. This results in *channel hopping* where the pairs of neighbours hop between different channels at each iteration.

The advantage of channel hopping is to have retransmission on a different channel than it was transmitted previously. The new channel is likely to be a more stable link. Otherwise, it will hop to another channel on the next cycle. This increases the likelihood of succeeding than retransmitting on the same channel, thus, forming a more stable topology. Nodes on different channels are allowed to run simultaneously on the same time slot as it does not interfere with each other transmissions. Channel hopping technique helps to combat external interference and impact the nodes differently on the channel.

2.3.2.2 MC-LMAC

Present a multi-channel MAC protocol, MC-LMAC, designed with the objective of maximizing the throughput of WSNs by coordinating transmissions over multiple frequency channels. MC-LMAC takes advantage of interference and contention-free parallel transmissions on different channels. It is based on scheduled access and dynamically switches their interfaces between channels. Time is slotted and each node is assigned the control over a time slot to transmit onUse 5 channels in the experiment.) The sequence number generation algorithm must guarantee that there is only one node among one-hop neighbors on any particulate channel. The receiving node transmits a smal a particular channel.

Multi-Channel Lightweight Medium Access Control (MC-LMAC) is a schedule-based multi-channel MAC protocol that takes advantage of contention and collision-free parallel transmissions on different channels. The main design is based on single-channel LMAC [37]. LMAC protocol enables the communicating entities to access the wireless medium on a schedule basis in which each node periodically uses a timeslot for transmission. Present a new multi-channel MAC protocol with a fully distributed scheduling mechanism that does not require a centralized scheduler to allocate timeslots. Nodes discover and take control of their slots and channels in a localized way by only exchanging information within their local neighborhood in MC-LMAC. MC-LMAC not only support many-to-one communication toward the sink node but also broadcasts and local-gossip operations.

MC-LMAC uses a common channel during the control period of each timeslot to let the receivers be informed about the requests and channels on which data will be sent. In MC-LMAC communication on a common channel at the beginning of each timeslot lets the new node collect full information about its neighborhood before starting transmission. If a node are switching between channel dynamically - MC-LMAC, all the receivers of a broadcast are informed on the common channel at the beginning of each slot. MC-LMAC does not require a dedicated broadcast channel. At the start of each timeslot, all nodes are required to listen on a common channel (different from the dedicated broadcast channel, this can be used for data

exchanges as well) in order to exchange control information.

MC-LMAC protocol we assume scheduled access, where each node is granted a timeslot and performs its transmissions within this timeslot without contention.

To access the medium and send messages,nodes select/control a timeslot together with a frequency on which the transmission do not conflict with the other concurrent transmissions.

In the initialization state, nodes sample the medium for an upcoming packet to synchronize with the network and enter the synchronization state. Every node synchronizes with its parents. Prior to data transmission, the nodes send control messages which include information about the current slot and frame numbers. Upon the reception of a message during initialization, a node records the current slot and frame numbers which are sent in the control message. The timing scheme is started by the sink node at network initialization. When the neighbors of the sink receive the transmission, they synchronize their clocks with the sink's clock. The synchronization continues hop-by-hop as each node synchronizes with its parent node. The nodes detect synchronization errors by comparing the received slot and frame numbers in the control message with their local slot and slot numbers. If a difference is detected, nodes transmit back to the initialization state.

The nodes choose a time slot autonomously such that a node's transmissions in that slot does not conflict with the transmissions of other nodes in the same slot. If there is no conflict, a node uses the same time slot in the upcoming frames. Time slot selection process takes place either during network initialization or whenever a conflict occurs and a node is required to select a new time slot to eliminate conflict. If the time slots are selected during network initialization, the sink node starts the selection process by getting the control of a time slot. When a node joins a network, first it has to discover a free time slot to transmit its data. Potential receivers should transmit a list of the time slots during which they are already receiving. The scheme lets the new node determine the list of free slots that can be used without possible collisions. A node randomly selects its time slot from the set of free slots. All the nodes are given an opportunity to select an empty slot. This guarantees that every

node can select a slot to carry out its transmissions without conflicts. All the nodes keep a bit vector called occupied slot vector. It is used for storing the information about the slots occupied by neighbors and is transmitted during the node's time slot to share this information with potential transmitters.

In MC-LMAC time slots are selected with channels. A node can use the same time slot that is used by a 2-hop neighbor on a different frequency so that parallel transmissions are not disturbed at common neighbours. Consequently, more transmissions can take place with the same number of time slots. In MC-LMAC, a node occupies slot vectors per channel and selects a time slot to be used on a particular channel. Rules of MC-LMAC, a node does not select a time slot on any of the frequencies which is used by the neighbors.

A time slot consists of a common frequency (CF) phase and a split phase. In the CF phase, all nodes switch to the common control channel to address their destinations and to be informed whether they are addressed in the current slot. In the split phase, senders and intended receivers switch to the channel on which the control message and data transmission will take place. During the CF phase, the intended destination id and node id are transmitted. This enables the sender to notify the destination node and invite it to switch its radio to the sender's channel. The sender's channel number is equal to the index or CF slot number where it notifies the destination node. Therefore, no extra information is needed to be transmitted. In the split phase, the sender first sends a control message which can be considered as a preamble packet, and then continues with the transmission of the data message.

The receivers listen during the whole CF phase in order to be informed about the intended destinations. It switches its transceiver on the sender's associated frequency or to standby mode by entering into a passive state for the remainder of the time slot to conserve energy. Nodes can also send broadcast messages by transmitting a broadcast address during the CF slot. All the nodes receiving the broadcast request switch to the sender's frequency.

In MC-LMAC the duration of the CF period increases with more channels and this causes the nodes to spend more energy on listening for the potential incoming

packets. [38]

2.3.2.3 YMAC

Proposes an energy efficient multi-channel MAC protocol Y-MAC. A light-weight channel hopping mechanism. Y-MAC avoids redundant channel assignment by not allocating fixed channels to the nodes. Initially, messages are exchanged on the base channel. When a traffic burst occurs, a receiver and potential senders hop to one of other available channels, according to the hopping sequence. Each node is guaranteed to receive at least one message on the base channel.

Y-MAC is a TDMA-based multi-channel MAC protocol. In general, TDMA-based MAC protocols allocate a time slot to each node in the network. The allocated time slot is used for data transmission or data reception according to the protocol.

In Y-MAC, time is divided into several fixed-length frames and each frame is composed of a broadcast period and a unicast period. Every node must wake up at the start of the broadcast period to exchange broadcast messages. If there are no incoming broadcast messages, each node turns off its radio, until its own receive time slot to save energy. Y-MAC separates broadcast traffic from unicast traffic, this makes broadcasting more reliable (separate broadcast message queue and unicast message queue). Broadcast messages are exchanged only within the broadcast period. At the beginning of the broadcast period, every node tunes to the base channel.

Y-MAC propose a light-weight channel hopping mechanism exploiting multiple channels to reduce the packet delivery latency in unicast. If a node receives a unicast message on the base channel, it hops to the next channel to receive the following message. The next channel is calculated by the hopping sequence generation algorithm. Any nodes that have pending messages destined to the same receiver also hop to the same channel and compete again. In this way, bursts of messages ripple across channels and only one node uses the base channel at any one time. (successive packets are sent each on a different frequency following a pre-determined hopping sequence. This hopping sequence starts at the base station. Use 5 channels in the experiment.) The sequence number generation algorithm must guarantee

that there is only one node among one-hop neighbors on any particulate channel. The receiving node transmits a small and independent packet at the start of the time slot to notify the contention losers whether it will wait during the next time slot or not (unicast time slot - contention loser can retry in the next time slot on the next channel).

There's a tradeoff between the number of time slots and the delivery latency. The more time slots we have, the more nodes we can allocate exclusive time slots to, but delivery latency increases due to the prolonged length of the frame period.

A sender and a receiver have to agree on the communication channel as well as the transmission timing. This necessitates time synchronization algorithms. In Y-MAC, sensor nodes synchronize their upcoming timer events by exchanging the time remaining in the current superframe period. Implemented by adjusting the expiration times of timer events. Time synchronized nodes periodically broadcast the information required for time synchronization. This consists of the time remaining to the start of the next frame period. Sink periodically broadcasts control messages to initiate the network. A node which is trying to join the network turns on its radio transceiver to receive this timing information. Once a node receives the first control message, it sets its time remaining to the next frame period to equal that of the sender. When the receiving node receives the time synchronization information from the sending node, it averages the time remaining and adjusts the expiration time of its timer event for timing error compensation. As a result, the starting points for the next frame period of these two nodes get closer. To lessen the control overhead for time synchronization, the timing information is included in control messages the every node periodically broadcasts to maintain network connectivity.

The medium access design of Y-MAC is based on synchronous low power listening. We define the time slot length to be long enough to receive one message. Contention winner (contention between potential senders) can transmit a message to the destination node. If the channel is clear, a preamble is transmitted until the end of the contention window to suppress competing transmissions. The receiver wakes up at the end of the contention window to receive the data. [31]

2.3.3 Asynchronous Systems

Recent asynchronous multi channel MAC layers are Chrysso and MiCMAC. MiC-MAC is built based on ContikiMAC, the default radio duty cycling in Contiki 2.7 that works in a single channel. The details of these are explained below.

2.3.3.1 ContikiMAC

ContikiMAC radio duty cycling mechanism is the default radio duty cycling mechanism in Contiki 2.7. It uses a power efficient wake up mechanism with a set of timing constraints to allow device to keep their transceivers off. The wireless transceiver consumes as much power when passively listening for transmissions from other devices as it does when actively transmitting, so the transceiver must be completely turned off to save power. ContikiMAC keep their radios turned off for roughly 99% of the time. ContikiMAC uses only asynchronous mechanisms, no signalling messages, and no additional packet headers. ContikiMAC packets are ordinary link layer messages. ContikiMAC uses a fast sleep optimization, to allow receivers to quickly detect false positive wake-ups (fast sleep optimization to allow receivers to quickly go to sleep when faced with spurious radio interference), and a transmission phase-lock optimization. The idea of periodic wake-ups has been used by many protocols, such as B-MAC, X-MAC and BoX-MAC. The phase-lock optimization has been previously suggested by WiseMAC and has since been used by other protocols as well.

ContikiMAC uses a fast sleep optimization, to allow receivers to quickly detect false-positive wake-up and a transmission phase-lock optimization, to allow run-time optimization of the energy-efficiency of transmissions.

ContikiMAC is a radio duty cycling protocol that uses periodical wake-ups to listen for packet transmissions from neighbors. If a packet transmission is detected during a wake-up, the receiver is kept on to be able to receive the packet. UNICAST - When the packet is successfully received, the receiver sends a link layer acknowledgement. To transmit a packet, a sender repeatedly sends its packet until it receives a link layer acknowledgement from the receiver. Acknowledgement transmission is done as part of the unicast packet reception. Packets that are sent as broadcasts do

not result in link layer acknowledgements. Instead, the sender repeatedly sends the packet during the full wake-up interval to ensure that all neighbors have received it. Since a broadcast transmission does not expect any link layer acknowledgement, the transmitter can turn of its radio between each packet transmission to save power.

ContikiMAC wake-up frequency of 8Hz which results in a wake-up interval of 125 ms. Radio duty cycle increase with the wake-up frequency; more wake-ups, the total power consumption of the network increase (channel check rate higher than 8Hz).

ContikiMAC wake-ups use an inexpensive Clear Channel Assessment (CCA) mechanism that uses the Received Signal Strength Indicator (RSSI) of the radio transceiver to give an indication of radio activity on the channel. If the RSSI is below a given threshold, the CCA returns positive, indicating that the channel is clear. If the RSSI is above the threshold, the CCA returns negative, indicating that the channel is in use.

Detection - ContikiMAC CCAs do not reliably detect packet transmission: they only detect that the radio signal strength is above a certain threshold. The detection of a radio signal may mean that a neighbor is transmitting a packet to the receiver, that a neighbor is transmitting to another receiver, or that some other device is radiating radio energy that is being detected by the CCA mechanism. ContikiMAC must be able to discern between these events and react properly.

Fast Sleep - The fast sleep optimization lets potential receivers go to sleep earlier if the CCA woke up due to spurious radio noise. Specific pattern of ContikiMAC transmissions: If CCA detects radio activity but the radio activity has a duration that is longer than the maximum packet length, the CCA has detected noise and can go back to sleep (if the activity period is not followed by a silence period). If the radio activity is followed by a silence period that is longer than the interval between two successive transmissions, the receiver can go back to sleep. If the activity period is followed by a silence period of the correct length, followed by activity but no start of packet could be detected, the receiver can go back to sleep.

Transmission Phase-Lock - A sender can learn of a receiver's wake-up phase

by making note of the time at which it saw a link layer acknowledgement from the receiver. The sender can assume that the reception of a link layer acknowledgement means that the sender has successfully transmitted a packet within the receiver's wake-up window and thus the sender has found the receiver's wake-up phase. The sender can commence its successive transmissions to this receiver just before the receiver is expected to be awake. The transmission will be significantly shorter than a normal transmission, because it occurs just before the neighbor is expected to be awake. Reducing the length of the transmission thus reduces radio congestion. The phase-lock mechanism is implemented as a separate module from ContikiMAC. The phase-lock mechanism maintains a list of neighbors and their wake-up phases.

Fast sleep and phase-lock optimizations significantly reduce power consumption. This is because of a phase-locked transmission being shorter than non-phased-locked transmissions, leading both to less energy being spent on transmissions and to less radio congestion [39].

2.3.3.2 MiCMAC

Propose a practical extension of low-power listening, MiCMAC, that performs channel hopping, operates in a distributed way, and is independent of upper layers of the protocol stack. MiCMAC, a channel-hopping variant of ContikiMAC. MiCMAC is based on low-power listening and have nodes wakeup periodically on different channels. MiCMAC is practical and independent from other layers in the protocol stack. MiCMAC employs pseudo-random channel hopping sequences. MiCMAC inherits its basic design from ContikiMAC and extends it for efficient multi-channel support. Inherit its design and integrate channel hopping in it. Each time a node wakes up to listen, it hops (switches) channel according to a pseudo-random sequence. Sender schedules the packet for sending just before Receiver's expected wake-up, switches to Receiver's expected channel, samples it to ensure it is clear, sends the packet and waits for acknowledgement (ACK). If Sender receives the ACK, it knows that communication was successful thus it updates its information of Receiver's wake up time and channel and goes back to sleep. Otherwise, Sender assumes that its information of Receiver's wake-up time and channel

is wrong and needs to be updated. Each node switches its channel periodically on every wakeup cycle following a pseudo-random sequence. We generate the pseudo-random channel numbers using a Linear Congruential Generator (LCG). The sequences they generate are uniformly distributed and they are computationally simple.///what is it? reference!///. The generated sequences appear random and contain each possible number in the range exactly once before repeating the whole sequence again. Advantage when we want to find a node's wakeup channel. Do blind channel hopping because of simplicity as local blacklisting would involve some overhead for synchronizing the blacklists among neighbours. "Previous work has shown that even random blind channel hopping improves network connectivity, efficiency and stability when compared to single-channel"///reference??//. To increase optimizations, MiCMAC uses of predefined hopping sequence. Instead of calculating the hopping sequences at runtime, we provide a static table of all sequences used on the network. Each node simply selects its sequence according to its MAC address. MiCMAC extends ContikiMAC's phase-lock with a $channel - lock$ to anticipate the wakeup channel.

When communicating with a neighbour for the first time, the sender picks any channel and transmits strobes repeatedly for a maximum of number of channels wakeup (e.g over four wakeups for four channels). Doing so guarantees that an idle receiver will wake up exactly once on the channel where the strobing occurs. Upon successful unicast reception, the receiver sends an ACK frame that includes the pseudo-random generator parameters so that the sender can compute the next wakeup channels. Next time the same pair of nodes communicates, sender will calculate the next wakeup channel, by generating the receiver's next wakeup channel; taking into account the number of periods elapsed since the last successful unicast.

MiCMAC supports broadcast by two variants of MiCMAC. i) MiCMAC basic support for broadcast is by strobing only one of the possible channels continuously for N times the wakeup period (sending strobes over one channel for exactly 4 wakeup periods when using 4 channels). Downside is the increased cost in energy and increased channel use. ii) MiCMAC-BC where nodes wake up on a dedicated

broadcast channel at every period in addition to their baseline wakeup on the unicast pseudo-random channel. Broadcast transmissions are always done over this channel for a duration of only one wakeup period. Downsides are reduced robustness as all broadcast occur on the same channel and two wakeups are needed instead of one at every period (one wakeup for pseudo-random generated channel, one for broadcast channel).

MiCMAC did not require any change in RPL routing nor other layers. Run RPL with ETX as a metric and the MRHOF objective function. The link ETX between two nodes is updated at every transmission attempt, independent of the channel, resulting in an aggregated estimate over all channels in use.

When using 16 channels, the performance degrades due to using all (including bad) channels and due to increased cost of broadcast and channel-lock operations. MiCMAC sees its performance degrade when using more than 4 channels. The per-channel measurements are not strictly required for MiCMAC to operate but do them for the sake of fair comparison. Channel diversity which increases the number of usable links due to different signal propagation obtained when hopping to a new channel. Channel diversity also leads to a more stable topology, reduced number of parent switches. MiCMAC hides losses from the routing layer, resulting in a more stable topology. [8]

2.3.3.3 Chrysso

//inner, outer, scan Chrysso is a multi-channel protocol extension, that leverages the channel diversity of sensor node radio. Chrysso mitigates the effect of external interference by switching only the affected nodes to a new set of channels effectively evading the interference source on spot. It is specifically tailored to data collection applications (*Collect* routing protocol), and allocates channels for individual parent-children groups with the parent coordinating a channel switch upon detecting interference. For efficiency nodes normally operate on two channels only, one for incoming and one for outgoing traffic. We also present a novel scanning procedure probing all channels that can be used either at bootstrap or when losing contact with the parent. Chrysso maintains a pre-defined logical list of available channels

so that a parent and children are consistent with their view on the next channel. Chrysso implementation uses five 802.15.4 channels: 26, 14, 20, 11 and 22 in the specified order.

The core of Chrysso is the set of control loops that manages whether a parent-children pair should stay on the same channel or switch to a new channel.

The inner loop is responsible for coordinated channel switching between a parent and its children as soon as external interference is detected. When interference completely block any communication, a coordinated channel switch cannot be performed. For this case, Chrysso uses the outer loop, a watchdog mechanism that initiates an autonomous channel switch.

Inner loop - A child node periodically collects data from the channel quality monitor and piggybacks that onto the data packet. A child node responds to channel switching requests from its parent. Parent uses protocol-stack specific policies to determine whether or not the current measured channel quality indicates external interference. The parent computes an average over the backoff values (congestion backoffs as a measure of interference) and checks whether it exceeds a predetermined threshold. If this is the case, having notified all its children, the parent node switches to the next logical inchannel.

Outer loop - The outer loop functions as a watchdog for case of severe interference that disrupt the operation of the inner loop. A node (in child and parent role) decides independently to switch channels based on the protocol-specific policies. The decisions involve no explicit coordination between nodes. For detecting severe interference, a child node monitors the number of failed transmissions (messages that were never sent) and switches its outchannel if the failure ratio exceeds a threshold. Autonomous channel switches should only be performed when the inner loop fails to trigger a channel switch for an extended duration. Likewise, the parent switches to the next inchannel when the ratio of received packets drops below a pre-set value. Child node, if the number of failed transmissions exceeds a certain fraction of the total transmission attempts, then the child node switches its outchannel instantly and autonomously. Likewise, a parent node keeps a record of

the number of packet received. If the fraction of received over expected packets falls below a threshold, the parent switches its inchannel instantly and autonomously.

For a child node, both channel switches and routing switches necessitate a reset of its collected channel quality statistics. The child node also clears its routing entries. The watchdog initiates the scan mode if the node does not find a parent available on the new channel.

Scan mode is invoked at network bootstrap or whenever a sensor node loses connectivity to its parent and has no other entry in its neighbor table (following a channel switch by the outer loop). During scan mode, a sensor node sweeps across the list of channels. The previously used outchannel is blacklisted for that period.

***Fundamental problem of multi-channel protocols: channel deafness (not hearing a packet on a different channel). [7]

Neighbour discovery - Because neighboring nodes operate of different channels, topology information available at a child node is only partial, and also subject to change, as neighbours may switch channel during network operation. Chrysso employs a special neighborhood discovery phase, called *scanmode* to find a new parent. Scan mode is only triggered on demand as it incurs additional overhead on processing and energy consumption. A node performing neighborhood discovery scans through the list of available channels to search for a new parent.

2.3.4 Comparison and Discussion

The main features of the presented MAC protocols are summarised in table 2.1 Existing MAC protocols suffer from several issues:

1. Synchronous vs asynchronous design
   Both

2. Sender vs receiver initiated design

3. Channel hopping design

4. Broadcast support

Protocol	Medium Access	Channel Assignment	Channel Switching	Common Period	Broadcast
MC-LMAC	TDMA	Senders	Once per time slot	CF	Yes
Y-MAC	TDMA + collision window	Dynamic	Once per time slot	CP + CFs	Yes
TSCH	TDMA + collision window				
MiCMAC	MiCMAC	Dynamic	One per wake up time	Yes	Yes
Chrysso	Operates over ContikiMAC	Dynamic	Change channel when bad	No	No

Table 2.1: Comparison of studied MAC protocols

2.4 Routing Protocols (Network Layer Protocols)

In WSN, the sensor nodes have a limited transmission range. The network layer is responsible in routing the data across the network from the source to the destination. Routing protocols for WSNs are responsible for maintaining the routes in the network and ensure reliable multi-hop communication. Routing protocols in WSNs differs from traditional routing protocols depending on the Operating System. Contiki provides IP communication in both IPv4 and IPv6. However, as sensors have a small amount of memory, uIP, which is a small RFC-compliant TCP/IP stack that makes it possible to communicate over the Internet [40, 41]. uIP () to reduce the resources it requires. uIP implementation is designed to have only the absolute minimal set of features needed for a full TCP/IP stack [40, 41].

2.4.1 Introduction

Flooding and gossiping are two classical mechanism to relay data in sensor networks without the need for any routing algorithms and topology maintenance. In flooding, each sensor receiving a data packet broadcasts it to all of its neighbors and this process continues until the packet arrives at the destination or the maximum number of hops for the packet is reached. Gossiping is a slightly enhanced version

of flooding where the receiving node sends the packet to a randomly selected neighbor, which picks another random neighbor to forward the packet to and so on. Easy to implements but have drawbacks include implosion caused by duplicated messages sent to the same node, overlap when two nodes sensing the same region send similar packets to the same neighbor and resource blindness by consuming large amount of energy without consideration for energy constraints. Gossiping cause delays in propagation of data through the nodes. [42]

In order to maximize the use of multichannel in improving packet delivery, routing topology plays a big role in providing an optimized routing tree to the network that is scalable and energy efficient. Network design objectives - scalability (densely deployed. since the number of sensor nodes are in the order of tens, hundreds or thousands, the network protocols designed should be scalable). Reliability - must provide error control and correction mechanism to ensure reliable data delivery over noisy, error-prone and time-varying wireless channels. Low power consumption - sensor nodes are battery powered, it is crucial to reduce the power consumption of sensor nodes so that the lifetime of the sensor nodes and the network is prolonged). Adaptability - nodes may fail, join or move; network protocols should be adaptive to such density and topology changes. Channel utilization - since sensor nodes have limited bandwidth resources, communication protocols designed for sensor network should efficiently make use of the bandwidth to improve channel utilization (MAC layer??). Routing protocol approaches can be classified into () types which are flat based and data centric, hierarchical, location based and network flow and quality of service (QoA) aware.

At the network layer, the main aim is to find ways for energy-efficient route setup and reliable relaying of data from the sensor nodes to the sink so that the lifetime of the network is maximized.

2.4.2 Classification of Routing Protocols

Many routing algorithms (protocols) were developed for WSN. All major routing protocols proposed for WSNs can be divided into ///().

Almost all of the routing protocols can be classified as data centric, hierarchi-

cal, location based, network flow and quality of service (QoS)-aware based, and hybrid. Data-centric protocols are query-based and depend on the naming of desired data, which helps in eliminating many redundant transmissions. Hierarchical protocols aim at clustering the nodes so that cluster heads can do some aggregation and reduction of data in order to save energy. Location based protocols utilize the position information to relay the data to the desired regions rather than the whole network. Network flow modelling and protocols that strive for meeting some QoS requirements along with the routing function. [42] Hybrid approach uses some form of hierarchical technique in combination with other approaches such as network flow to achieve a stable and additional energy saving(???).

The routing protocol is highly influenced by the data delivery model, especially in regard to the minimization of energy consumption and route stability. Similar to MAC protocols(????), the routing protocols can b for continuous, event-driven, query-driven and hybrid. In continuous delivery mode, each sensor sends data periodically. In event-driven and query-driven models, the transmission of data is triggered when an event occurs or a query is generated by the sink. Some networks apply a hybrid model using a combination of continuous, event-driven (sender) and query-driven (receiver) data delivery. [42]

2.4.2.1 Flat based and Data Centric

In data-centric routing, the sink sends queries to certain regions and waits for data from the sensors located in the selected regions. Since data is being requested through queries, attribute-based naming is necessary to specify the properties of data.

Data centric - all communication is neighbor-to-neighbor (no need addressing mechanism).

SPIN [43] is the first data-centric protocol, which considers data negotiation between nodes in order to eliminate redundant data and save energy.

Later, Directed Diffusion [44] has been developed and has become a breakthrough in data-centric routing. There are many other protocols that have been proposed either based on Directed Diffusion (Rumor Routing [45], GBR [46], CADR

[47])or following a similar concept (TEEN [48] which is also a hierarchical-based, ACQUIRE [49]).

SPIN - the idea behind SPIN is to name the data using high-level descriptors or meta-data. Before transmission, meta-data are exchanged among sensors via a data advertisement mechanism, which is the key feature of SPIN. Each node upon receiving new data advertises it to its neighbors and interested neighbors (sensors advertise the availability of data allowing interested nodes to query that data). Difference: In SPIN, sensors advertise the availability of data allowing interested nodes to query that data. (sensor asks others if they want its data) Advantages: of SPIN is that topological changes are localized since each node needs to know only its single-hop neighbors. Disadvantage: SPIN's data advertisement mechanism cannot guarantee the delivery of data. If the nodes that are interested in the data are far away from the source node and the nodes between the source and destination are not interested in that data, such data will not be delivered to the destination at all.

Directed Diffusion - is an important milestone in the data-centric routing research of sensor networks. The idea aims at diffusing data through sensor nodes by using naming scheme for the data. Direct Diffusion suggests the use of attribute-value pairs for the data and queries the sensors in an on demand basis by using those pairs. The interest is broadcast by a sink through its neighbors. Each node receiving the interest can do caching for later use. The interests in the caches are then used to compare the received data with the values in the interests. A gradient is a reply link to a neighbor from which the interest was received. By utilizing interest and gradients, paths are established between sink and sources. Several paths can be established so that one of them is selected by reinforcement. The sink resends the original interest message through the selected path with a smaller interval. When a path between a source and the sink fails, a new or alternative path should be identified. Directed Diffusion search among other paths which are sending data in lower rates. Difference: In Directed Diffusion, the sink queries the sensor nodes if a specific data is available by flooding some tasks. (sink to nodes) Advantages: Caching is a big advantage in term of energy efficiency and delay. Energy efficient since it

is on demand and there is no need for maintaining global network topology. Disadvantages: Cannot be applied to all sensor network applications since it is based on a query-driven data delivery model.

2.4.2.2 Location Based

Sensor nodes are addressed by means of their locations. Location information is needed in order to calculate the distance between two particular nodes so that energy consumption can be estimated. If the region to be sensed is known, using the location of sensors, the query can be diffused only to that particular region which will eliminate the number of transmission significantly.

GEAR [50]- Geographic and Energy-Aware Routing is an energy efficient routing protocol proposed for routing queries to target regions in a sensor field. The sensors have localization hardware equipped (GPS unit) so that they know their current positions. The sensors also are aware of their residual energy as well as the locations and residual energy of each of their neighbors. GEAR uses energy aware heuristics that are based on geographical information to select sensors to route a packet towards its destination region. GEAR uses energy aware and geographically informed neighbor selection heuristics to route a packet towards the target region, the idea is to restrict the number of interests in Directed Diffusion by only considering a certain region. In GEAR, each node keeps an estimated cost and a learning cost of reaching the destination through its neighbors. The estimated cost is a combination of residual energy and distance to destination.

GAF [28] - Geographic adaptive fidelity (GAF) is an energy-aware location-based routing algorithm. GAF conserves energy by turning off unnecessary nodes in the network without affecting the level of routing fidelity. It forms a virtual grid for the covered area. Each node uses its GPS-indicated location to associate itself with a point in the virtual grid. Nodes associated with the same point on the grid are considered equivalent in terms of the cost of packet routing. Such equivalent is exploited in keeping some nodes located in a particular grid area in sleeping state in order to save energy. Before the leaving time of the active node expires, sleeping nodes wake up and one of them becomes active. GAF keeps a representative node

always in active mode for each region on its virtual grid. Increases the lifetime of the network by saving energy.

2.4.2.3 Network Flow and QoS-aware

Network flow - route setup is modeled and solved as a network flow problem. QoS-aware protocols consider end-to-end delay requirements while setting up paths un the sensor network.

[51] Maximum lifetime energy routing solution - the main objective of the approach is to maximize the network lifetime by carefully defining link cost as a function of node remaining energy and the required transmission energy using that link. Use Bellman-Ford shortest path algorithm to find the least cost paths to the destination (are found).

Sequential assignment routing (SAR) [52] - is the first protocol for sensor networks that includes the notion of QoS in its routing decisions. It is a table-driven multi-path approach. The SAR protocol creates tree rooted at one-hop neighbors of the sink by taking QoS metric, energy resource on each path and priority level of each packet into consideration. One of these paths is selected according to the energy resources and QoS on the path. Advantages: less power consumption than the minimum-energy metric algorithm which only focuses the energy consumption of each packet without considering its priority. SAR maintains multiple paths from nodes to sink - ensure fault-tolerance and easy recovery. Disadvantages: the protocol suffers from the overhead of maintaining the tables and states at each sensor node especially when the number of nodes is huge.

2.4.2.4 Hierarchical

Scalability is one of the major design attributes to sensor networks. A single-tier network can cause the gateway to overload with the increase in sensor density. The single-gateway architecture is not scalable for a larger set on sensors covering wider area of interest. The aim of hierarchical routing is to efficiently maintain the energy consumption of sensor nodes by involving them in multi-hop communication within a particular cluster and by performing data aggregation and fusion in order to decrease the number of transmitted messages to the sink.

Low-energy adaptive clustering hierarchy LEACH [26] is one of the first hierarchical routing approaches for sensor networks. The idea proposed in LEACH has been an inspiration for many hierarchical routing protocols, TEEN [48], APTEEN [53], PEGASIS [54], Hierarchical-PEGASIS [55] and HEED [56]. LEACH is one of the most popular hierarchical routing algorithms for sensor networks. The idea is to form clusters of the sensor nodes based on the received signal strength and use local cluster heads as routers to the sink. All the data processing such as data fusion and aggregation are local to the cluster. LEACH uses a load balancing mechanism that periodically rotates the role of cluterhead nodes. Cluster heads change randomly over time in order to balance the energy dissipation of nodes. The nodes die randomly and dynamic clustering increases lifetime of the system. LEACH is completely distributed and requires no global knowledge of network. However, LEACH uses single-hop routing where each node can transmit directly to the cluster-head and the sink. Therefore, it is not applicable to networks deployed in large regions. Disadvantages: dynamic clustering brings extra overhead (head changes, advertisements). Cluster heads consumes a larger amount of energy (do data aggregation and fusion tasks to reduce the number of data transmissions) when they are located further away from the sink.

RPL/////

2.4.2.5 Hybrid Based

CTP

** Contiki Collect protocol and CTP are state-of-the-art address-free data collection protocols that provide a way for nodes to send data packets towards a data sink. Nodes do not need to know the address of the sink. Use ETX finding paths that minimize the number of packet transmissions to reach the root. Neither CTP nor Contiki Collect are IPv6-based. Contiki Collect uses the Contiki Rime stack [13].

The data collection is an address-free protocol that sends messages towards a sink node somewhere in the network. The protocol is address-free in the sense that the originating nodes do not send their messages to a specific addressed node. Instead, the nodes send their messages towards the nearest sink in the network. The

protocol does two things. It first builds a tree that originates at the sink nodes. The nodes build the tree by sending periodic announcements containing the number of hops away from the sink. After having built the tree, the nodes start sending messages towards the root of the tree. The protocol sends the messages using hop-by-hop reliable unicast [57].

2.4.3 RPL Routing Protocol

2.4.3.1 Graph Building Process

2.4.3.2 Types of RPL Messages

2.4.3.3 Objective Function

2.4.3.4 Trickle Timer

The protocol makes use of IPv6 and supports not only traffic in the upwards direction, but also traffic flowing from a gateway node to all other network participants. RPL is a distance vector routing protocol that makes use of IPv6. A Destination Oriented Directed Acyclic Graph (DODAG) which is routed at a single destination is built. The graph is constructed by the use of an Objective Function (OF) which defines how the routing metric is computed. OF specifies how routing constraints and other functions are taken into account during topology construction. The protocol tries to avoid routing loops by computing a nodes position relative to other nodes with respect to the DODAG root, called Rank and increases if nodes move away from the root and decreases when nodes move in the other direction. RPL specification defines 4 types of control messages for topology maintenance and information exchange. DODAG Information Object (DIO) is the main source of routing control information. It may store information like the current Rank of a node, the current RPL Instance, the IPv6 address of the root, etc. Destination Advertisement Object (DAO) enables the support of down traffic and is used to propagate destination information upwards along the DODAG. DODAG Information Solicitation (DIS) makes it possible for a node to require DIO messages from a reachable neighbor. DAO-ACK (optional) is sent by a DAO recipient in response to a DAO message. RPL specification defines all 4 types of control messages as

ICMPv6 information messages with a requested type of 155. RPL adapts the sending rate of DIO message by extending the Trickle algorithm. Upward routing is a standard procedure which enables network devices to send data to a common data sink, also called sometimes a gateway or root node. (BORDER ROUTER??!!) The Mode of Operation (MOP) field is set by the DODAG root A DIO message may be extended by the use of options. DODAG Configuration option plays a crucial role for parameter exchange. MaxRankIncrease field defines an upper limit for the Rank. MinHopIncrease field stores the minimum increase of the Rank between a node and any of its parent nodes. 3 types of nodes in a RPL network. 1. Root nodes which are commonly referred in literature as gateway nodes that provide connectivity to another network. 2. Routers which may advertise topology information to their neighbors. 3. Leafs that do not send any DIO messages and only have the ability to join an existing DODAG. The construction of the topology starts at a root node that begins to send DIO messages. Each node that receives the message run an algorithm to choose an appropriate parent. The choice is based on the used metric and constraints defined by the OF. Afterwards, each of them computes its own Rank and in case a node is a router, it updates the Rank in the DIO message and sends it to all neighboring peers. *In most sensor node deployments several data collection points (root nodes) are needed. Whenever the sending timer expired, RPL doubles it up to the maximum value. Whenever RPL detects an event which indicates that the topology needs active maintenance, it resets the timer to minimum value. Router nodes forward DIO control messages for topology maintenance - such messages are sent in a multicast manner to the neighboring nodes. RPL node does not process DIO messages from nodes deeper (higher Rank) than itself. RPL metric - status includes typical resources such as CPU usage, available memory and left energy. Node energy consumption - node should consider the energy level of its neighbors before picking them as possible parents. RPL metric specification defines 3 possible states for the first information field: powered, on batteries and scavenger **** This may be a rough estimation of how much load a node experiences for a given period of time. ETX - is an approximation of the expected number

of transmissions until a data packet reaches the gateway node*. A node that is one hop away from the root with perfect signal strength and very little interference may have ETX of 1. ETX is bidirectional single-hop link quality computation between 2 neighbor nodes.* A metric called Packet Reception Rate (PRR) is calculated at the receiver node for each window of received packets. Downward routing - by supporting P2MP traffic it is possible for a network administrator to control nodes that are even not in range. RPL specification defines 2 modes of operation for supporting P2MP. 1. Non-storing mode which makes use of source routing. In this mode each node has to propagate its parent list up to the root. After receiving such topology information, the root computes the path to the destinations. Each node has to extend the DAO message. After collecting the needed information, the root pieces the downward route together. If it needs to send a data packet to a given destination the IPv6 Source Routing header is used. 2. Storing mode which is fully stateful. Each non-root and non-leaf network participant has to maintain a routing table for possible destinations. DAO messages are used by RPL nodes to propagate routing information in order to enable P2MP traffic. DAO is no longer propagated to the DODAG root. Instead, it is sent as unicast to all parent nodes which maintain additional downward routing tables [58].

Routing protocol called RPL. RPL does not define any specific routing metrics, path costs or forwarding policies. RPL leaves this open so that different networks can apply different mechanisms to meet different objectives such as minimizing latency or minimizing energy consumption. ContikiRPL implementation of the RPL protocol which allows replaceable routing objective functions. ContikiRPL is the main IPv6 routing protocol in Contiki. RPL is a distance-vector protocol for IPv6 networks comprising low-power devices connected by lossy links. The protocol maintains Directed Acyclic Graph (DAG) topologies toward root nodes. The topologies are built proactively according to an objective function. It is flexible regarding the rules to form topologies and to select next-hops for individual packets. Routing decisions are taken by the objective function, which essentially specifies the constraints and metrics used in a network. One objective function uses a simple

hop count and one uses expected transmissions (ETX) to do the forwarding decision. The simple hop-count objective function results in a shorter path length at the expense of higher power consumption [59].

DIS - may be used to solicit a DIO from a RPL node. A node may use DIS to probe its neighborhood for nearby DODAGs. DIO - carries information that allows a node to discover a RPL Instance, learn its configuration parameters, select a DODAG parent set and maintain the DODAG. DAO - used to propagate destination information Upward along the DODAG. In Storing mode, the DAO message is unicast by the child to the selected parent(s). In Non-Storing mode, the DAO message is unicast to the DODAG root. The DAO message may optionally be acknowledged by its destination with a Destination Advertisement Acknowledgement (DAO-ACK) message back to the sender of the DAO. DAO-ACK message is sent as a unicast message packet by a DAO recipient (a parent or DODAG root) in response to a unicast DAO message [60].

RPL is a routing protocol that provides any-to-any routing in low-power Ipv6 networks, standardized by the IETF in March 2012. Its design is largely based on CTP, the reference data collection protocol for sensor networks. The RPL topology is a DODAG (Destination Oriented Directed Acyclic Graph) built in direction of the root, typically an access point to the Internet. Any-to-any traffic is routed first upwards, i.e. towards the root until a common ancestor of destination and source is found, and then downwards, following the nodes routing table. RPL uses a simple rooted topology instead of a full mesh; it is devoted to the maintenance of reliable paths to a single destination. The purpose of this strategy is to scale to large networks while containing the routing overhead, at the price of increased hop count (routing via a common ancestor). RPL terminology, the distance from a node to the root according to the routing metric is called rank. RPL requires sharing routing tables among siblings. In RPL, nodes propagate their routing entries through unicast (so-called DAO messages) to their parents. RPLs rank hysteresis mechanism prevents nodes from switching parent for too little rank improvements. Contiki-MAC has wakeup consists of two clear channel assessments and has a phase-lock

mechanism where senders record their neighbors wake-up phase and use it to make the next transmissions cheaper. *Experience an outage during which they cannot receive or send any data. This reflects for example scenarios where battery maintenance requires to disconnect a part of the network, or where external interference (e.g. WiFi or Bluetooth) affects communication. RPL experiences a sharp drop in the reliability during the first outage, consequence of failed MAC transmissions. Nodes react by switching parent, which heals the topology and slowly improves reliability. The next outages result in less churn (less agitate) [61].

////OBJECTIVE FUNCTION

////TRICKLE TIMER A protocol uses Trickle to periodically advertise the most recent data it has received, typically through a version number. Routing control traffic - a protocol uses Trickle to control when it sends beacons that contain routing state. Once the RPL network is established, it reduces the rate of control messages, exponential increase. To avoid control message explosion, nodes suppress transmissions if it hears too many messages from other - called the Trickle algorithm.

Dynamically adjusting transmission windows allows Trickle to spread new information on the scale of link-layer transmission times while sending only a few messages per hour when information does not change.

To save energy the DIOs are sent periodically controlled by the trickle timer whose duration is doubled each time it is fired. The value of trickle timer starts from the lowest possible value lmin and is doubled each time it is transmitted until it reaches its maximum possible value of lmax [62].

2.4.4 Comparison and Discussion

Chapter 3

Multichannel Cross-Layer Routing Protocol

WSNs often suffer from frequent occurrences of external interference such as Wi-Fi and Bluetooth. Multichannel communications in wireless networks can alleviate the effects of interference to enable WSNs to operate reliably in the presence of such interference. As a result, multichannel solution can improve the network efficiency of spectrum usage, network stability, link reliability, minimise latency and minimise the number of packet loss, hence, retransmission.

This chapter presents Multichannel Cross-Layer Routing Protocol (MCRP), a decentralised cross-layer protocol with a centralised controller. Our cross layer multichannel protocol focuses on the network and application layers. This allows channel assignment decisions to be made thoroughly without being limited by the low layer complexity. The system has two parts: a central algorithm which is typically run by the LPBR and selects which channel each node should listen on; and a protocol which allows the network to communicate the channel change decision, probe the new channel and either communicate the success of the change or fall back to the previous channel. MCRP concentrates on finding channels for the nodes that are free from or have low interference. It allows the allocation of these channels in a way likely to minimise the chances of nodes which are physically near to communicate on the same channel. Hence, it reduces cross interference between different pairs of nodes.

3.1 MCRP Design

The design of the multichannel protocol is based on several crucial observations:

i. **Channel assignment** - Sensors have limited memory and battery capabilities. In order to maximise the sensors lifetime, a centralised LPBR that has larger memory and fully powered is used for decision making. LPBR has complete knowledge of the topology which enables it to make good channel assignment decisions based on a two-hop colouring algorithm.

ii. **Interference** - External interference cannot be predicted, thus channels cannot be allocated beforehand as it varies over time and locations. It is impossible to determine a single channel that is free from interference at any location. The protocol checks the channel condition each time before deciding on a channel change to reduce interference and maximise throughput.

iii. **Frequency diversity** - Multichannel increases the robustness of the network towards interference. However, applying multichannel to the existing RPL may hinder detection of the new nodes and cause problems for maintaining the RPL topology. Two mechanisms are introduced to overcome this problem. These solutions are explained in details in Chapter 4. Existing nodes maintain a table of the channels on which their neighbours listen and use unicast to contact those nodes. New nodes listen on a Contiki default channel (26) and when connecting search through all channels. As in RPL, periodically all nodes broadcast RPL control messages on the default channel in an attempt to contact new nodes.

MCRP is build based on these observations to overcome the shortcomings of sensors and sensor networks. The rest of this chapter focuses on MCRP designs in channel selection strategy, channel switching decisions, channel quality checking and the reconnection strategy.

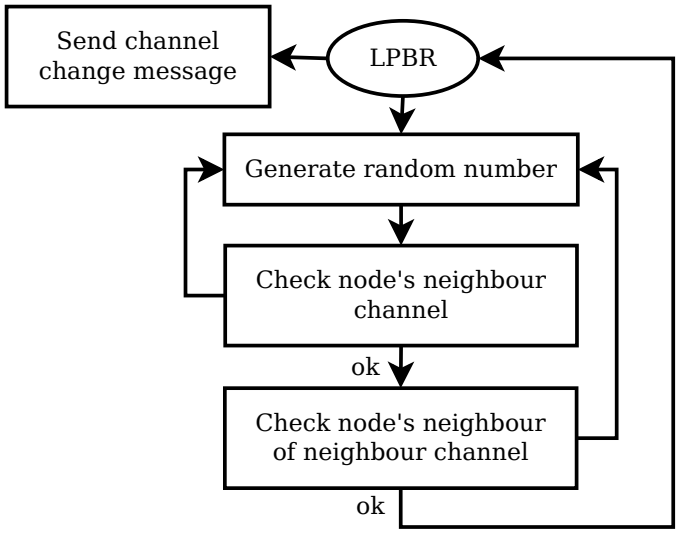


Figure 3.1: Channel selection strategy

3.2 Channel Selection Strategy

One main advantage of the proposed system is generality. Any algorithm can be used at the LPBR to assign channels. MCRP uses a two-hop colouring algorithm to select a channel to be assigned to a node. The two-hop colouring algorithm attempts to ensure that nearby nodes do not communicate on the same channel and risk interfering with each other. The protocol is inspired by the graph colouring problems [63]. The core idea is that no node should use the same listening channel as a neighbour or a neighbour of a neighbour (two hops). This allows fair load balancing on the channels and reduces channel interference that could occur when two nearby nodes transmit together on the same channel. The nodes used in this for this experiment have a transmission range of approximately 20-30 metres indoors and 75-100 metres outdoors [64]. It could be the case that many nodes in a sensor network are in the transmission range of each other and potentially interfered with.

All nodes are initialised to channel 26 which is the common default channel for Contiki MAC layer since it often has fewer interference problems with Wi-Fi and other sources. The studies in [7, 8, 6] use a set list of whitelisted channels in their experiments and have channel 26 in common. The usual RPL set up mechanism is used to exchange control messages that are required to form an optimised topology

before channel assignments can take place. The nodes will only be on the same channel once during the initial setup. This enables the node to detect and find nearby neighbours that are in range before it can decides on the best route based on the list of neighbours it can be connected to.

In the two-hop colouring algorithm, the LPBR chooses a node to which it will assign a channel to listen on. The selection is random (from channels 11 to 26) based on the full range available [1]. The channels that were tested to have severe interference for one node might gives good result for another node depending on the location of the node which might not be within the range of where the channel has severe interference previously. MCRP has its channel quality checking mechanism before it decides on a channel which allows random channel selection to take place.

The protocol checks neighbours and neighbours of neighbours to see if any of those are listening on this channel already. If any are, a new channel is picked from the remaining list of available channels. If the LPBR has knowledge of existing bad channels then those channels can be blacklisted. Knowledge of channel interference which is gained by probing can be used to decide that a channel should not be used. If a channel is found then the channel switching protocol is triggered. If no channel can be found meeting these conditions, the current channel is kept. Figure 3.1 summarised the strategy in LPBR channel selection.

The node selection algorithm must only attempt one channel change at a time to ensure probing is done on the correct new channel and for the node to finalise the channel to be used before another node attempts a channel change. The protocol ascertains that the channel change attempt will always result in a message returned to the LPBR either confirming the new channel or announcing a reversion to the old channel. Until one or other of these happens, no new channel change will be made to enable the neighbours transmitting on the correct channel.

3.3 Channel Switching

Figure 3.2 shows the state machine for the channel switching protocol. As explained in the previous section, a choice of a new channel by the channel selection protocol

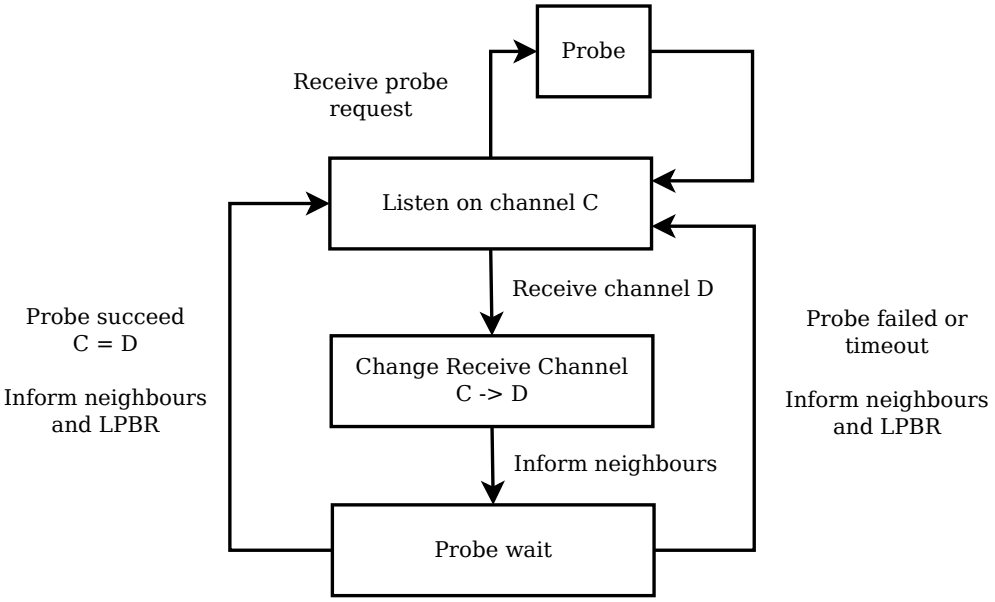


Figure 3.2: Channel switching processes

causes a change channel message to be sent to the appropriate node. Upon receiving a channel change message, a node *N* stores its current channel *C* and communicates to all its neighbours the new channel *D* that it wishes to change to. Those neighbours will update their neighbour tables to ensure that they now send to node *N* on channel *D*. The node *N* begins the channel quality checking process with each neighbour in turn by sending them a probe request. If this process fails for any neighbour then the node reverts to channel *C*. If all channel quality checks succeed, the node *N* will listens on channel *D*. In both cases, node *N* informs its neighbours of the decision to channel *C* or *D* and informs the LPBR of the channel checking results. The channel checking process uses probe packets that might interfere with other transmissions temporarily. However, it is important to emphasise that the network remains fully functional and connected at all stages of this protocol.

3.4 Channel Quality Checking

The channel quality checking is invoked each time a node changes channel after receiving a message from the LPBR. A node *N* changing to channel *D* informs all neighbours in turn, of the new channel *D* it will be listening on as described

in the previous section. It then enters the *Probe Wait* state and begins channel quality checking with each tree neighbour in turn. In describing the channel quality checking process, it is worth emphasising the distinction between neighbours and tree neighbours. Node neighbours are all nodes that a given node knows it could transmit to. Tree neighbours are the nodes that a node does transmit to through the topology formed by the RPL protocol.

In the *Probe Wait* state, node N sends a *Probe* message to each neighbour in turn. The neighbours respond to the message by sending eight packets to N on the new channel D. The buffer can accommodate eight packets at a time. As the packets might not be sent immediately due to wakes up and collisions, sending more packets would have the risk of being dropped. The condition of the channel is further investigated through the number of retransmissions and packet collisions of the probing packets for accuracy of the channel condition.

If the probing process times out (because of some communication failure) or the number of probe packets received is above a threshold (currently set to 16, including retransmissions and collisions) then node N immediately exits *Probe Wait* state and reverts to channel C its previous channel.

All neighbours are informed of the change back to channel C and the LPBR is informed of the quality check failure with a summary of all probes received. If, on the other hand, all channel quality checks succeed, the change to channel D becomes permanent for node N and it informs the LPBR of the results of the probing (numbers of packets received) and the channel change.

Probing is essential to make the channel change decision. It gives a quick overview of the channel condition based on the number of probing messages received. It is worth noting that probing is only done between the node and the tree neighbours. Neighbours that are not tree neighbours will not use the node as a route during their transmission thus, there is no need for probing to take place with those neighbours. However, the neighbours still need to know the channel value given that RPL control messages are sent to neighbours directly without using the routes.

3.5 Reconnection Strategy

RPL topology stability (using routing metric) remains the same in multi channel [65, 60]. The nodes can still change the parents as usual as all neighbours know each other new channels. The neighbours that are not part of the route do not probe the parent when making the channel decision. However, the neighbours are informed of any channel changes.

This enables the topology to be optimised when communication fails and further improved through MCRP as the nodes have knowledge of the listening channels of all other nodes within the range. If a new node tries to join the topology, it sends a RPL control message through all channels as the listening nodes are unlikely to be on the default channel.

The listening nodes send a broadcast on a default channel to discover new nodes (in Contiki default, new nodes will start on channel 26) and send RPL messages through unicast when the neighbours are known to reduce unnecessary transmissions in broadcast. New nodes and nodes which fall off the network can now rejoin on many potential channels.

Chapter 4

Implementation

MCRP is implemented on the TelosB mote platform. It uses Contiki, a lightweight operating system as the software development platform that supports the standard IPv6. The implementations of MCRP across the layers in Contiki are described in details, specifying the changes that were introduced and undertaken in addition to the default parameters and settings in Contiki.

4.1 Contiki

Contiki is defined by four layers network stack: the network layer, the MAC layer, the radio duty cycling (RDC) layer and the radio layers. The network layer includes support for TCP, UDP, IPv6, IPv4, RPL routing protocol and 6LoWPAN. IPv6 over Low Power Wireless Personal Area Networks (6LoWPAN) is a header compression and fragmentation format for IPv6 packets delivery over IEEE 802.15.4 networks [66]. Contiki implements the minimal set of IPv6 protocol required, 6LoWPAN

Contiki	IoT/IP	Applications
Network	Application	HTTP
	Transport	TCP, UDP
	Network, Routing	IPv6, IPv4, RPL
	Adaptation	6LoWPAN
MAC	MAC	CSMA/CA
RDC	Duty Cycling	ContikiMAC
Radio	Radio	IEEE 802.15.4

Table 4.1: Contiki network stack

adaptation layer for IPv6 header compression and fragmentation which is routed over the low power and lossy networks (LLN) in RPL.

Contiki's configuration options for communications, buffer management and network interface are explained before looking into MCRP implementation for ease of reading.

4.1.1 Communication Stacks

Contiki contains two communication stacks, uIP and Rime. uIP [67] is a small RFC-compliant TCP/IP stack that is designed to contain only the essential (required/necessary) features to provide Contiki with TCP/IP networking support to allow Contiki to communicate over the Internet compared to the traditional TCP/IP that requires (high) resources to fit in a limited RAM capabilities of a sensor. The minimal set of features includes IP, ICMP, UDP and TCP protocols compared to the traditional TCP/IP that requires (high) resources that could not be supported in the limited RAM capabilities sensor. The uIP is mostly concerned with the TCP and IP protocols and upper layer protocols [40, 41].

Rime is Contiki's lightweight communication stack that aims to simplify the sensor network protocols implementation by reusing code in a layered manner [68]. Rime combines layers of simple communication abstractions to form a powerful high-level abstraction ranging from best-effort anonymous broadcast to reliable network flooding. Parts of the Rime stack can be used by the underlying MAC or link layer. Additional protocols that are not in Rime can be implemented on top of the stack.

Applications in Contiki can decide to use one of the communication stacks available, both or none at all. uIP can run over Rime and similarly, Rime can run over uIP [69].

4.1.2 Buffer Management

Chameleon [57] is a communication architecture in Contiki that consists of Rime communication stack and a set of packet transformation modules. It uses an abstract representation of the information which allows access to the low-level features of

the underlying MAC and link layer protocol from the applications and layers implemented on top of the Chameleon architecture. It also allows the output from the protocol stack to be adapted by other communication protocols. In Chameleon architecture, the parsing of its header is separated from the communication stack. This allows uIP or Rime communication stack to be used as described in Section 4.1.1. Chameleon architecture enables the layers to access information without violating the layering principle.

In buffer management module of Chameleon architecture, all incoming and outgoing packets from the applications and packet attributes are stored in a single buffer called the Rime buffer [67, 57, 68, 9]. All layers of Contiki's network stack including uIP, Rime and the underlying link layer operate on the same packet buffer for the buffer management. The Rime buffer has no locking mechanisms as it is a single priority level buffer. The buffer only holds the current packet.

Protocols that need to queue packets allocate the queue buffer dynamically. Queue buffer is used to hold the queued packets such as for MAC protocols that have high rate of incoming and outgoing messages before it can send or process the receiving packets; or when the radio is busy and the MAC protocol has to wait for the radio medium to be free before proceeding with transmissions. Queue buffer is used to avoid the risk of the packet overwritten by the newer packet. The Rime buffer contents are copied into the queue buffer when there is a queue buffer allocated.

4.1.3 Tunslip

Serial Line Internet Protocol (SLIP) [70] is a protocol that has a low complexity and small overhead commonly used to encapsulate IP packets for point-to-point communication between the sink (LPBR) and the device connected such as an embedded PC across the serial connections. The communication between the devices can take place on any reliable network such as the Ethernet where the LPBR can be connected to an embedded PC which contains an Ethernet interface.

Contiki provides support to communicate with devices using SLIP through its tunslip tool. Tunslip is used to bridge the IP traffic between the LPBR and the

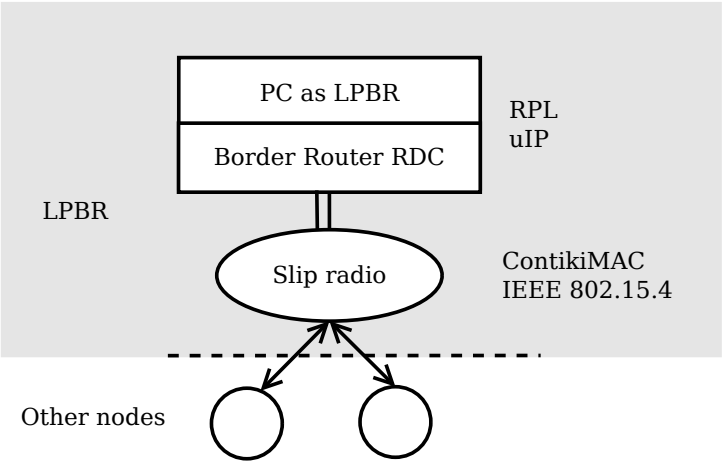


Figure 4.1: Low power border router

embedded PC over a serial line. The other side of the serial line does a similar job to bridge the embedded PC to the LPBR using the network interface. It constructs a SLIP tunnel between a virtual network interface (tun) and SLIP, the physical serial interface to encapsulate and pass the IP traffic to and from the other side of the serial line. The tun interface is used as any real network interface such as for routing and traffic forwarding [71, 72].

4.2 MCRP Implementation

MCRP is implemented in Contiki and uses ContikiMAC as the MAC protocol, RPL as the routing protocol. ContikiMAC is modified to allow multi channel where the channel selection processes take place on the upper layers and the channels are kept in the network neighbour table to ensure the correct channel.

The protocol implementation is separated into two types of nodes: i) the centralised LPBR where the bridging takes place between the border router on a PC to the nodes, and ii) the decentralised transmission nodes referred as other nodes. The implementations for both types are described below.

4.2.1 Low Power Border Router

As sensors have limited memory, most processing decisions at LPBR are transferred to a PC as it has more RAM and better processing capabilities. This enables MCRP

to have more thorough processes and to run in real time without draining the memory and battery on a sensor. LPBR is divided into two main parts as shown in Figure 4.1 where the PC is responsible as the application, transport, network and routing layers while a sensor (labelled slip radio) is set as the wireless interface to enable the PC to communicate with the other nodes via Contiki tunslip tool.

The LPBR acts as the tree root in RPL where it will initiate the creation of the RPL routing tree. LPBR is a special case as channel changes at LPBR is not as direct as other sensor nodes due to these two parts. However, it works similar ways to the other nodes.

LPBR main responsibility is to decide on the new channel selection. LPBR has no knowledge of all the channels condition at this point, thus, a channel is selected at random. LPBR keeps the results from the channel changes processes and based on it when selecting a new channel for the next node to ensure the new channel is at least two-hop away from another node using the same channel. This is done to ensure that the nearby nodes do not communicate of the same channel and risk interfering with each other.

Algorithm 1 Pseudo-code for two-hop colouring algorithm

Notations
R is a node that is a Route
N is a node Neighbour
RN is the Route's Neighbour node
currentCh is the node current listening channel
newCh is the new channel the node will change to
Pseudo-code
if *R currentCh* $\neq$ *newCh* **then**
    succeed one-hop
    check all *RN* channels
    if *RN* channel $\neq$ *newCh* **then**
        succeed two-hop
        confirm *newCh*
    end if
else
    generate a new *newCh*
    update the number of *newCh* generated for *R*
    use default channel 26 is all tries fail
end if

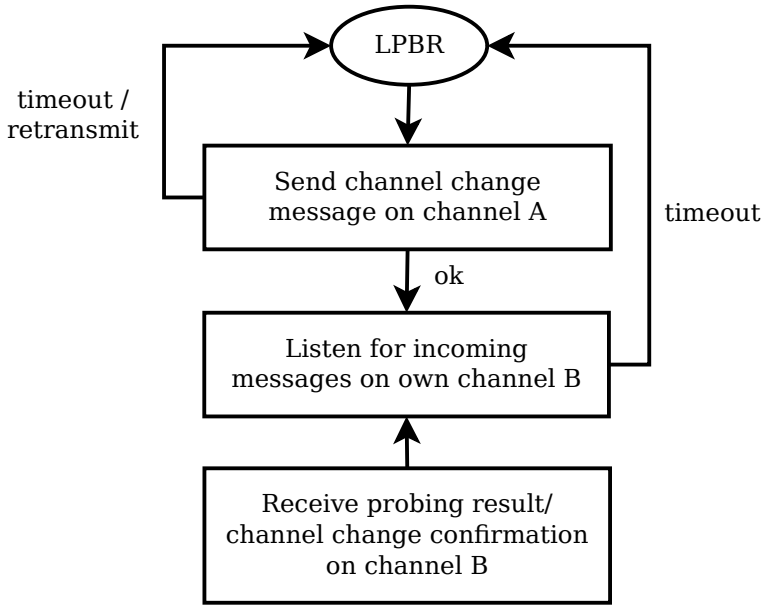


Figure 4.2: LPBR processes

The pseudo-code of the implemented two-hops colouring algorithm for new channel selection is shown in Algorithm 1. When the new channel is selected, LPBR will send the value to the intended node.

The new channel is stored in the buffer before the data is sent over SLIP to the radio-chip (slip-radio). As the slip radio is unable to access the neighbour table where the next hop node channel is stored, the channel value is passed through the buffer. LPBR keeps the updated value of all its neighbours channels in the neighbour channel. Slip radio that receives the packet buffer can access the channel value and kept the value is a simplified version of the neighbour table. This is done in order to ensure that the packet that is being queued or retransmitted is sent on the correct channel. The packets destination, which in this case, the next hop node is first check before the packet is transmitted each time. The MAC layer sets the channel accordingly before sending. ContikiMAC can access the simplified neighbour table as it is on the slip radio. The simplified neighbour table only keeps the information of the node neighbours and neighbours channels which are the critical information in order to transmit packets correctly. The other information that is related to the neighbours' conditions is monitored at the PC.

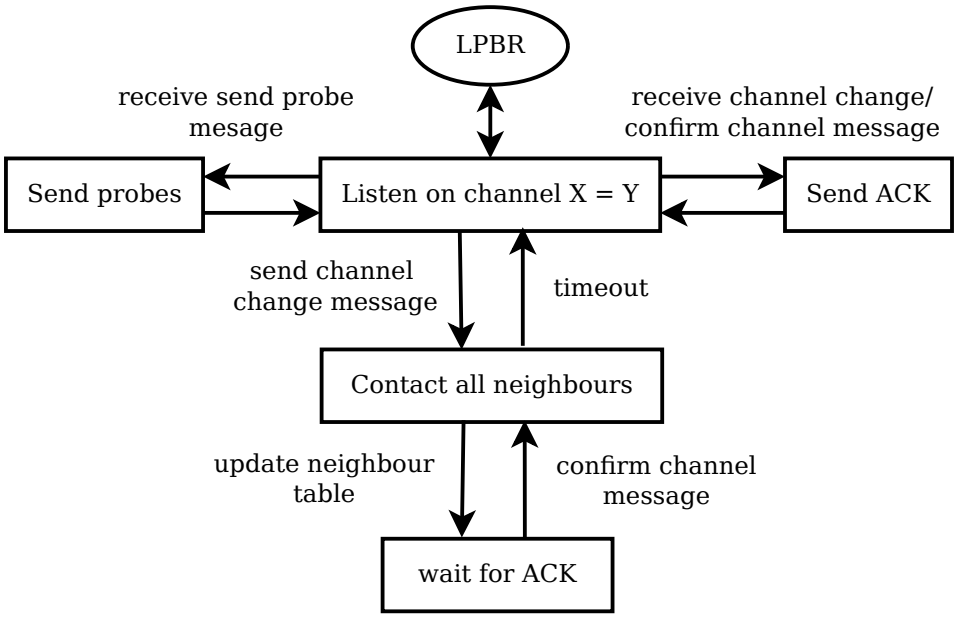


Figure 4.3: Nodes channel change processes

LPBR processes are shown in Figure 4.2. The slip-radio resets to its listening channel after the packet is transmitted. LPBR will wait and listen to any incoming packets. In the channel probing phase, LPBR does not take part in probing. However, LPBR is informed of the results of probing and kept a table of the probing results and channels to be able to use the information when deciding on a channel change based on the previous results of probing on the known channels.

4.2.2 Other Nodes

The new channel from LPBR that is received by the destination node is saved. Figure 4.3 shows the processes that the node takes in the channel change. When LPBR sends a *Channel Change* message to the destination node, the destination node will send a packet back to LPBR to acknowledge the channel change message. If LPBR does not receive the message, the channel change message is retransmitted. LPBR will then wait and listens for any incoming packets. At this point, channel changes processes will take place between the node and its neighbours. To clarify, *node* refers to the node that would like to change its listening channel and *neighbour node* is the neighbour of the node (including route node) that takes part in the channel change decision through probe message.

Unlike LPBR, other nodes have all the layers within the nodes themselves. This makes channel changes less complicated, however, the nodes are being limited by the number of RAM they have which resulted in probing values to be stored in the centralised LPBR. The nodes however, keep the probing results temporarily before the final decision of the channel is made.

The node sends the *Node New Channel* value to all of the node's neighbours. At this point, the new channel is not yet checked for its validity. However, all neighbours need to know the new channel as the node will change its listening channel to the new channel. Otherwise, packets cannot be received by the node since the listening channel is different than it was previously. The neighbours that receive the node's channel will update their *neighbour table* which is accessed from the application layer. Unlike LPBR, other nodes have all the layers within the nodes themselves. This makes channel changes less complicated as the nodes can access the information from any layer when required which in this case, accessing the neighbour table at the network layer from the application layer. In the neighbour table, a new entry is added to hold the channel value called *nbrCh*. As this is an important step in order to reduce the number of packet loss due to sending on the wrong channel, neighbours will send an acknowledgement of the new channel. Otherwise, it will be retransmitted.

The node will then send a *Start Probe* message to the neighbour that is a route node to start sending probing messages on the new channel. Not all neighbours are used as routes. The neighbours are chosen as route based on RPL OF which for this experiment is the ETX. The node will listens on the new channel and wait for the *Neighbour Probe* message. The route node starts to *Send Probe* messages every 3 seconds to allow retransmission or collision that could happen due to the busy channel. The maximum number of retransmission is configured to 3 attempts following the default value Contiki suggested. Collisions happen when the channel check keeps failing and new packets are constantly generated which could end up in a loop where no packets can be sent. The assumption that was made in this case is the channel will be cleared at some point which this loop will not happen. From

the experiments and simulations, this was proved true.

As only a small number of *Send Probe* messages are sent, the number of retransmissions and collisions that happen during the probing process are included in the channel decision process as it affects the channel reliability. As the retransmission and collision is a link layer process, the values are kept in a temporary *Retransmit Table* and is included to be sent in the next *Send Probe* message. This is because the value is only valid for that run. It gets reset each time a new packet is sent or received. The table is accessed at the application layer before the next *Send Probe* message is sent. The *Send Probe* message includes the current number of probe message and the number of tries (retransmissions and collisions) the previous packet had taken before it is successfully received. These values are used to decide if the channel is better than the current channel by giving a good probing result, meaning less retransmission.

The node keeps the value of all probing messages it receives. It sends the *Probe Result* message to LPBR. Unlike LPBR, the node has a limited RAM which resulted in past probing values to be stored in the centralised LPBR. The node however, keeps the probing results temporarily before the final decision of the channel is made. LPBR could use the information from the node's *Probe Result* to decide on a channel or blacklist bad channels.

The node then uses the values to decide whether the new channel is better than the previous channel by setting a threshold. The node then send *Confirm Channel* message to all neighbours that the node confirms to be listening on. The channel can be the new channel or the node can revert to the previous channel depending on the *Probe Result*. The neighbours will send an acknowledgement back to the node confirming the change. This is also important to ensure that all neighbours could communicate with the node on the correct channel. The neighbours will update their neighbour table of the node channel.

4.2.3 MAC Layer

As explained in Section 4.1.2, packets that have not been transmitted are queued in the buffer. The transmitting channel is set at the MAC layer as packets are not send

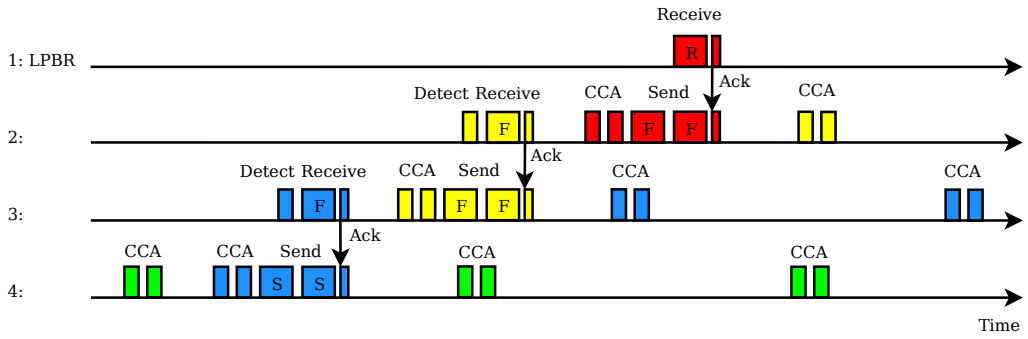


Figure 4.4: Multi channel ContikiMAC multi hop packet transmission

immediately if there are packets being queued. ContikiMAC is a single channel protocol. It is modified to support multi channel while complying to the same low power ContikiMAC principle. Each time the node goes to sleep, it will wake up on its listening channel waiting for incoming packets. If the node has a packet to send, it needs to change to the transmitting channel.

In order for the packet transmission or retransmission to be on the correct channel, the neighbour channel saved in the *neighbour table* at the network layer is accessed from the MAC layer and the channel is set to the transmitting channel. The node resets the channel to its listening channel after the transmission succeeded and goes back to sleep.

Figure 4.4 shows an example of a multi hop packet transmission in MCRP. The different colours represent different channels. Node 4's channel is represented by green, node 3 is blue, node 2 is yellow and node 1 is red. At each hop, the node changes to the next hop listening channel before forwarding the packet. In the example, node 4 is sending a packet to node 1, the LPBR through node 3 and node 2. Node 4 wakes up and checks for incoming packets on its channel. As it has a packet to be sent to node 1, it checks the next hop channel which is node 3 and changes the channel to node 3 listening channel. It checks if the channel is clear for transmission and proceed to send the packet to node 3. Node 3 detects the packet when it wakes up and receives the packet. Node 3 sends the link layer acknowledgement to node 4 so that node 4 stops sending the packet. Node 3 forwards the packet to node 2 on node 2 channel and node 2 to node 1, the LPBR which is the destination node. All

nodes reset their channel after the transmission and wake up on their own listening channel.

4.2.4 Network Layer

RPL is explained in Section 2.4.3. RPL control messages are tailored to accommodate MCRP proposal. Two main changes to RPL control messages are the DIS (which is sent by a new node to make it possible for a node to require DIO messages from a reachable neighbour) and DIO (the main source of routing control information) control messages. DIS and DIO control messages are usually sent using broadcast. However, DIS and DIO support unicast. MCRP sends RPL DIS control message in broadcast and DIO in both broadcast and unicast.

In MCRP, a new node that would like to join an existing tree needs to send the DIS control message to the reachable neighbours. However, as the reachable neighbours could be on different channel than it were initially during start up, the new node needs to send the DIS message on all channels available to be able to find the neighbours.

The neighbours that receive the DIS message will reply with a DIO message and a packet that tells the new node of its channel to communicate on. The new node updated the neighbour table and has successfully joined the tree.

If the neighbours do not receive the DIS from the new node before it is due to send the DIO message, the neighbours send a broadcast DIO on the default channel. The new node upon receiving the DIO will join the tree and updates the neighbour table. All neighbours send a DIO broadcast on the default channel and a DIO unicast for known neighbours on channels that the neighbours are listening on.

One of the main reasons for this is because broadcasting on all channels would require more energy and it would take a longer time before all reachable nodes receive the control message. This could delay changes that might happen in the tree, i.e. changing of parent node. Secondly, all nodes by default will switch on to the same default channel as that is how the nodes are being set up.

Chapter 5

Results and Discussions

This chapter presents the evaluation of MCRP. The experiment set up for Cooja simulation and FlockLab [73] testbed are explained below. In Cooja simulation, interference is introduced. MCRP is evaluated using an end-to-end packet delivery performance metric. The results from the experiments are presented and discussed.

5.1 Experimental Setup

MCRP is evaluated in Cooja simulated environment and FlockLab testbed. In Cooja simulation, an interference model is used as simulation allows full control over the test environment and the experiments are repeatable. Although the interference model does not fully mimic the behaviour of real world interference, it enables MCRP performance to be tested in various conditions when the channel performance degraded. Unlike simulation, testbed provides the ability to validate MCRP performance in the real wireless channel environments. However, the network's behaviour are complicated to examine, thus, comparing the testbed result with simulation results give a better understanding of the performance.

5.1.1 Simulation

MCRP is evaluated in the Cooja simulated environment with emulation of TMote sky nodes that feature the CC2420 transceiver, a 802.15.4 radio. The nodes run on IPv6, using UDP with standard RPL and 6LoWPAN protocols. The network consists of 31 nodes which are used to run the simulation where one node is used as the border router node, 16 interference nodes, and 14 duty cycled nodes that

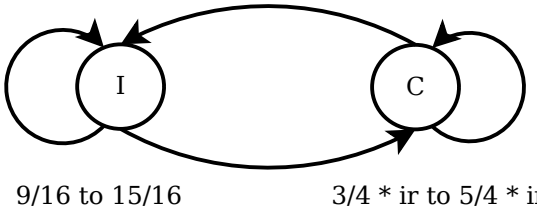


Figure 5.1: Low power border router

act as UDP clients to send packets to LPBR spanning over 20-30 metres between each node. RPL border router is used as LPBR in order to move most processing decisions on a PC as it has more RAM and better processing capabilities than a sensor. TelosB has limited RAM and ROM of 10K bytes and 48K bytes of flash memory [64]. By using a border router, this allows channel changing to be decided in real time without draining the memory and battery on a sensor. The border router also acts as the root of the tree.

A controlled interference node that generates semi-periodic bursty interference is simulated to resemble a simplified Wi-Fi or Bluetooth transmitter on several channels at random. The interference model proposed in [2] is used in the simulation. The interference has two states, a clear state and an interference state. In the interference state, the interference node generates packets for a time that is uniformly distributed between $9/16$ seconds and $15/16$ seconds. In the clear state the interferer produces no packets and stays in this state for between $3/4 * clear\_time$ and $5/4 * clear\_time$ where *clear_time* refers to the rate of interference. The model is illustrated in Figure 5.1. Multiple channels interference is used in the simulation to show the hypothesis that MCRP can help avoid interference. The scenario that is considered is where ContikiMAC with RPL system is subject to interference on its channel after set up has successfully completed so the RPL set up is allowed to complete before interference begins.

The protocol performance in loss over time in the presence of interference is observed. Two multiple channels interference scenarios are considered; (1) extreme and no interference rate on 8 channels each and (2) extreme, moderate, mild and no interference rate on 4 channels each. The interference channels are randomly cho-

sen from the available 16 channels and the same interference channels and rates are used throughout the experiments. However, channel 26 is kept clear from interference in order to ensure RPL set up is unaffected. In scenario 1, the interference rates are fixed to extreme and no interference to observe the effect it has on the channel changing decisions. In scenario 2, the interference rates are vary to observe how MCRP copes in deciding a channel when there is more interference than scenario 1 but with less interference intensity.

The simulation runs for a duration of 45-60 minutes to send 210-560 packets. When the nodes are switched on for the first time, all nodes are initialised to channel 26, the default channel for Contiki MAC layer. RPL is allowed five minutes to set up (which is ample time). RPL topology is formed in a minute. The simulation waits for another five minutes to allow trickle timer to double the interval length so that RPL control messages are not being sent frequently. The multichannel protocol is then runs for 25 minutes. In the 15 nodes simulation, the protocol takes 20-25 minutes to run the channel change set up. Another 5 minutes wait time is allowed if retransmissions happen. In a single channel simulation, all the nodes are changed to channel 22 after 5 minutes of RPL set up time. This allows RPL to have enough time to discover all nodes to form an optimised topology. The topology formation does not form completely if the interference node interferes from the beginning. The interference node starts sending packets to interfere after 3 minutes the system is switched on so that the interference channel is involve in the channel changes decision. It is proven that the protocol tries to avoid changing to the interference channel through time out and probing failures. After 30 minutes, the client nodes will send a normal packet periodically every 30-60 seconds to LPBR. This is done in order to avoid collision of the nodes sending at the same time.

5.1.2 Testbed

Similar to the simulation settings, the nodes run on IPv6, using UDP with standard RPL and 6LoWPAN protocols. The network consists of 26 nodes; 1 border router node and 25 duty cycled nodes as shown in Figure 5.2 where the grey lines represent the link qualities between nodes and yellow gradient is the noise. The indoor

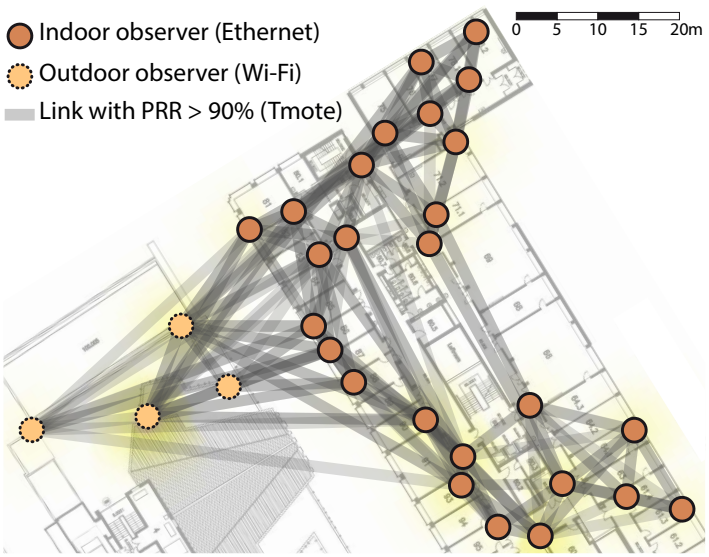


Figure 5.2: Layout of FlockLab deployment

nodes are distributed in offices, hallways and storerooms across one floor in an office building. Unlike in simulation, interference nodes are not added as the testbed has external interference from the surrounding (e.g. from the co-located Wi-Fi or limited connectivity) to see the effect on MCRP. The link qualities estimation information of the testbed is displayed on the FlockLab website. The link qualities and routing change throughout the day as the noise level on all channels and frequency bands are tested during testbed idle times.

As the channel condition in the testbed is beyond control, the experiment runs for a longer time period than the simulation. The testbed experiment is in the preliminary stage. The testbed is run for a duration of 4 hours to send up to 1250 packets depending on the availability of the nodes. The nodes are not switched on at the same time, thus RPL is allowed ten minutes to set up. MCRP is then run for 2 hours where the channel changes are done much slower than in the simulation as the channel condition is unknown, thus the number of retransmissions and collisions increases. The nodes wait for the normal incoming packet if the channel change processes finish early.

5.2 Evaluation

The performance of MCRP is compared against the standard ContikiMAC with RPL. MCRP is analysed using an end-to-end packet delivery performance metric. The transmission success rate is calculated from the sender to the receiver over multiple hops.

The simulations are repeated ten times. In all plots, the mean value of the ten simulations is plotted with error bars corresponding to one standard deviation in either deviation to give a measure of repeatability. The plots are of the proportion of received packets (from 0% to 100%) against time where the loss is measured over the previous time period. The x-value is shifted slightly left and right to prevent error bars overlapping.

5.2.1 Packet Loss Rates

The performance obtained in ContikiMAC with RPL (single channel) is compared with MCRP in terms of packet loss rate. As described previously, levels of interference used (referred to as *clear_time* in [2]) vary between 100% (no interference), 75% (mild), 50% (moderate) and 25% (extreme) where the percentage is the ratio of the time the channel is clear for transmission. All of the tests have a common format: the RPL procedure is allowed to set up without interference in order not to bias subsequent tests. Then the interferers begin to operate with a constant level (none, mild, moderate or extreme).

Figure 5.3 shows the results in simulation for ContikiMAC with RPL protocol. It can be seen that the level of packet loss varies considerably between experiments (the error bars are always large). It can also be seen that even for mild interference there is considerable loss and this gets worse as time proceeds. In the extreme interference case the loss always goes up until no packets are received. For mild interference the system evolves until it is losing around 20% of packets but this can increase.

The results from the single channel with interference is compared with the multichannel with the same interference rate of 75% (mild), 50% (moderate) and 25% (extreme). The test is done to evaluate MCRP behaviour in different interference

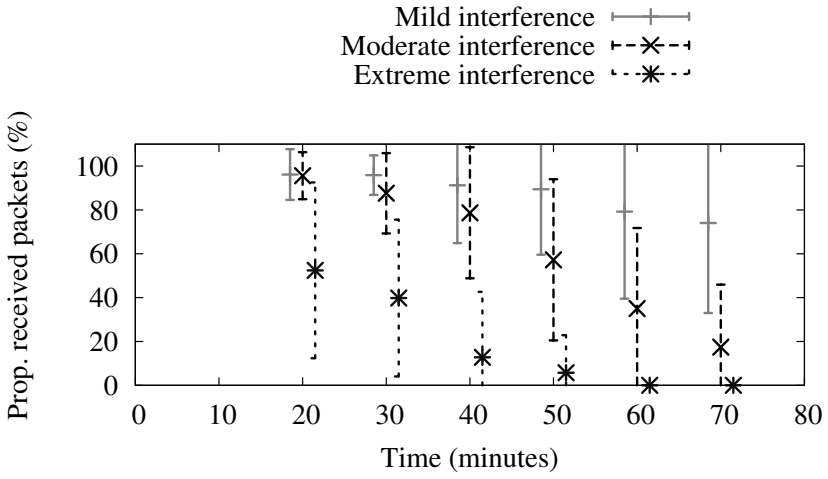
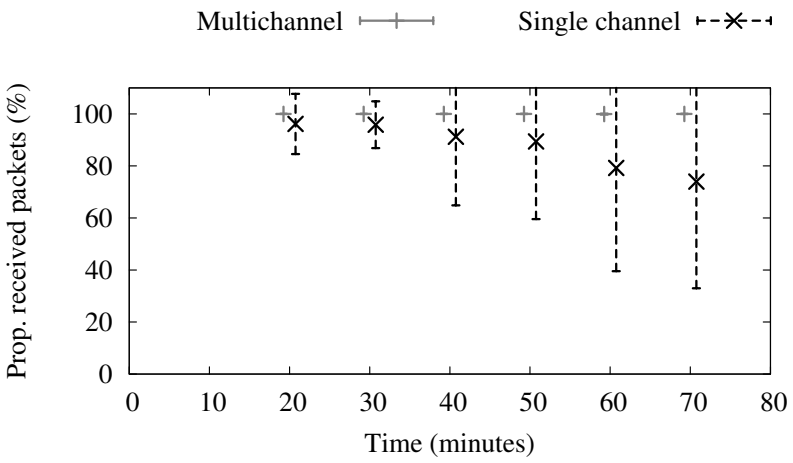


Figure 5.3: Level of packet loss for mild, moderate and extreme interference levels using single channel

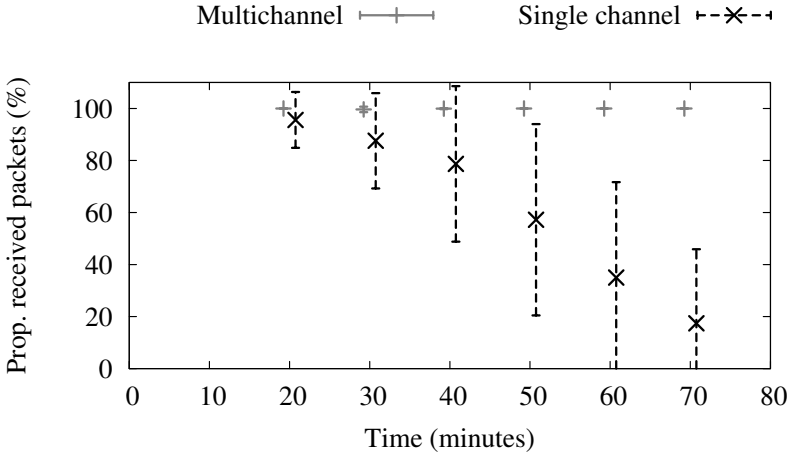
rate and to compare the result with a single channel case.

Figure 5.4 shows the averaged results from ten runs that were done. It can be observed that during high and moderate interference, if LPBR tries to send a channel change value that is the same channel as the interference, the request will either timed out or if it succeeds, the probing messages received are less than a threshold that allows for the node to change its listening channel to the new channel. This is as expected as MCRP checks the channel each time before deciding on the new channel to avoid interference channel. By doing this, it ensure that the node's listening channel is a good channel. This enable the use of all available channels without blacklisting any channel until it is sure that it is a bad channel through the probing process. The channel quality table is built at the LPBR that over time can be used to learn good and bad channels based on several probing processes. MCRP avoids the interference channel which as a result, resulted in less loss than in a single channel case.

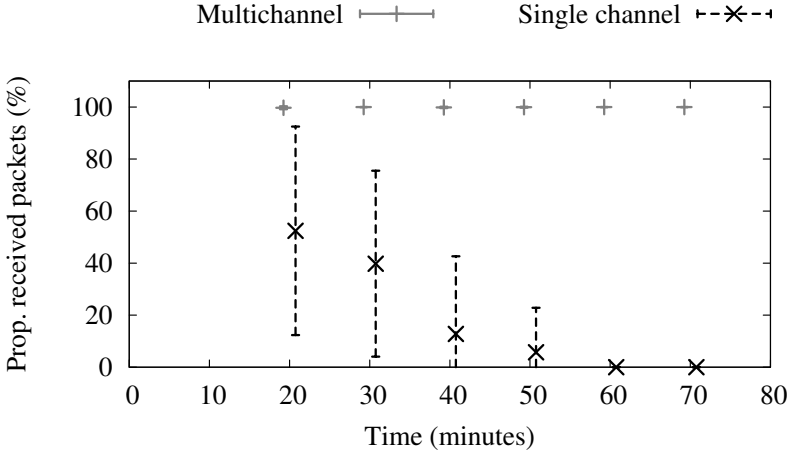
In the single channel, the node does not have enough time to recover from the interference to retransmit and drops all packets. Figure 5.4(c) shows that there are more packets drop over time and it stops receiving packets as it doesn't have enough buffer to store the incoming packet and the channel becomes congested. However, as the interference rate increases (less interference), the single channel performance improves as it has more time to recover.



(a) Mild Interference



(b) Moderate Interference



(c) Extreme Interference

Figure 5.4: Results of Multichannel RPL and a single channel RPL on different interference rate.

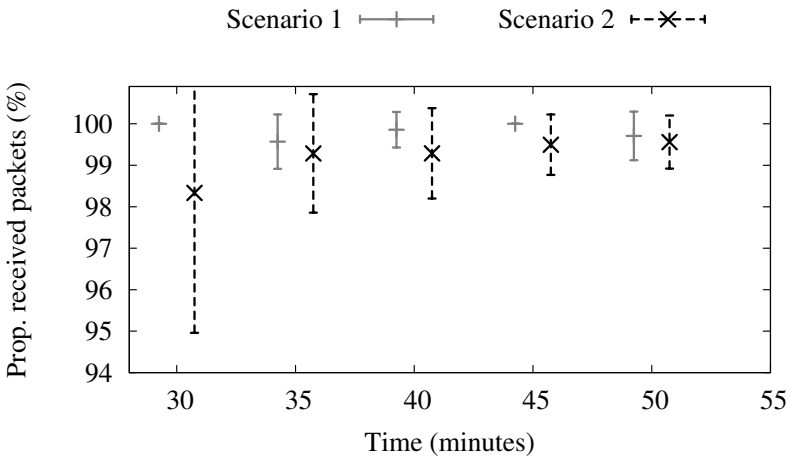


Figure 5.5: Level of packet loss for scenario 1 and scenario 2 using multi channel

In the mild interference case, all probing messages are received even though there is interference in that channel. This means that the channel can be used for transmission. As the interference rate is mild, all packets are received. This is also the case with a single channel. The interference does not affect the transmissions as the interference is not frequent enough. The node has enough time to recover from the interference through retransmissions. However, the interference would slightly effect the packet transmission over time. The channel change processes should run periodically to avoid this from happening.

To further evaluate MCRP capabilities to cope with interference from many sources, thus channels, two interference scenarios are considered. In scenario 1 half the channels (including the original channel) have no interference at all and half the channels have extreme interference. In scenario 2, four channels (including the original channel) have no interference, four have mild, four moderate and four extreme interference. Figure 5.5 shows multi channel results for these two scenarios. In scenario 1 the protocol performs extremely well, the packet loss is near zero and the protocol successfully detects channels with interference. Scenario 2 has similar results as in scenario 1. The protocol does well at reducing the effects of interference and could detect moderate and mild interference.

In the testbed case, the conditions for all channels are looked into to have a better idea of the variation in interference. The channels are tested on all channels

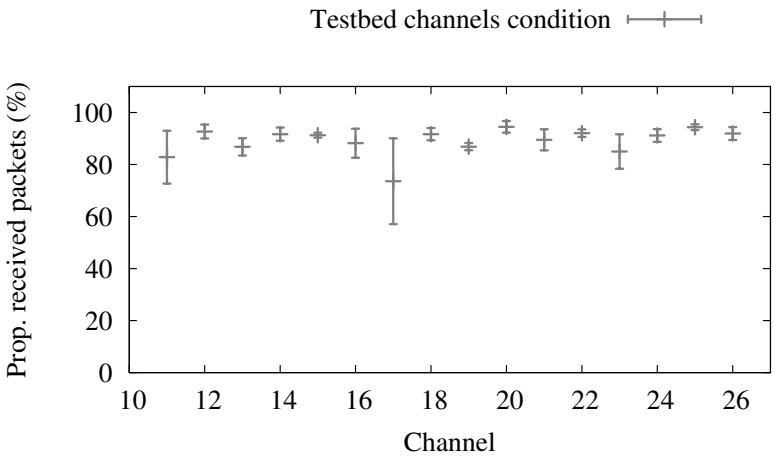


Figure 5.6: Level of packet loss on testbed for different channels

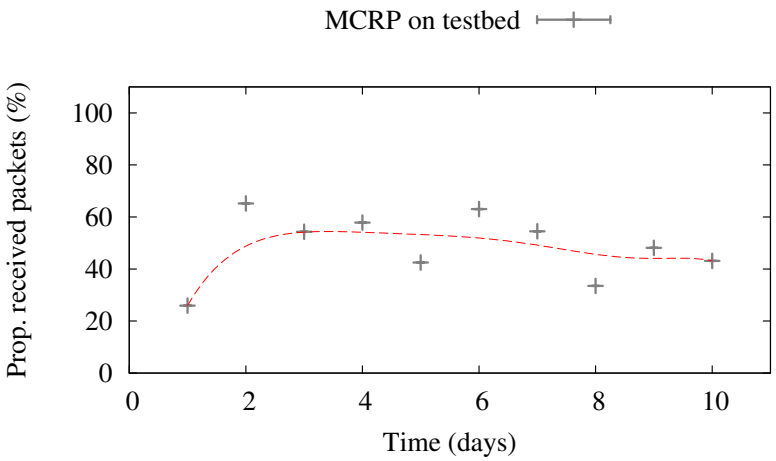


Figure 5.7: Level of packet loss on testbed using multi channel

for three different days. Figure 5.6 shows that the interference on the testbed varies for different channels on different days. However, it shows high receiving rate of 50%-90% in general.

Figure 5.7 shows the results from ten runs and the average that were done on different days. It can be observed that the results vary and multi channel give unstable results. On average, 50%-60% of packets are received at the LPBR. There are a few reasons for this low receiving packet values. As explained in Section 4.1.2, the incoming and outgoing packets are kept in the same buffer. When the channel is busy the outgoing and incoming packets will fill up the buffer which results in new packet to be dropped. The dropped packets might be the important packets that decide or confirm the new channel. This results in sending in incorrect channel.

The node will tries to send until it receives the link layer acknowledgement or time out which as a result, blocks the channel during its sending period by making the channel busy to be used by other node of the same channel. Another option to overcome this problem is by enabling RPL DIO on all channel so that the lost nodes can be recovered and updated with the correct channel each time the control message is sent. RPL DIO control message is sent to the neighbouring nodes according to the Trickle timer as explained in Section 2.4.3.

As the environment vary, it is not possible to replicate the same experiment as the topology might have changed depending on the RPL ETX value at each run. The nodes are possibly to take different routes to send the packet than in previous experiment.

In MiCMAC [8], it is stated that MiCMAC has a transmission success rate of 99% when using four channels. However, when more than four channels are used (8 or 16 channels), MiCMAC performance degrades to approximately 88% (16 channels) due to interference channels. The interference model that MiCMAC uses is different than in this experiment. They compare the result with Chrysso where Chrysso has a transmission success rate of approximately 88% for 4 and 8 channels and suffers greatly in the case of 16 channels with 60% success rate. MCRP on the other hand, shows greatly reduced loss rate with any number of channels at approximately 99%.

5.2.2 Setup Overhead

Obviously the system of changing channels and probing to see if a channel is free of interference introduces a certain amount of overhead into the protocol. This takes the form of (a) extra messages passed and (b) extra time taken to set up. Default RPL on ContikiMAC for the topology considered in these experiments completed its set up using 276 packets. MCRP, the multi-channel protocol completed its set up in 716 packets, that is an overhead of 440 packets on top of RPL. This overhead comes from the channel changing messages to nodes and neighbours, probing messages, channel confirmation messages and acknowledgement packets which are required to ensure a thorough channel change decision. However, it is worth mentioning that

this is a one-off cost. This represents (in this experimental set up) approximately one hour of extra packets in the situation of a deployment that is meant to work for weeks or months. In terms of set up time, the protocol begins to change channels only when the RPL set up process is complete (or at least stabilises). The set up time is 1154 seconds beyond the RPL set up time of 286 seconds. However, it should be noted that, in fact, the system remains fully functional and capable of sending packets during the set up so this set up overhead does not matter to data transmission. Therefore it can be concluded that data sending costs (extra packets) of set up are negligible in the context of a deployment that will last more than a day. The extra set up time is also negligible within this context and furthermore does not degrade performance of the network during this set up phase.

Chapter 6

Energy vs Loss Tradeoff

////citeSOFTWARE-BASED DUNKELS. implement an on-line energy estimation technique in Contiki. These approaches measure the state of components, such as radio, sensors and LEDS, to calculate consumption./////CHECK!

-requires more energy (to do MCRP) but reduce retransmissions in the long term -how much energy than usual? -improvement in loss when using MCRP?

Three are three main ways that have been exploited for an energy-efficient WSNs which are through MAC protocols, transmission range and routing protocols.

Energy consumption of a node depends on the interference patterns. Developed a generic method for capturing interference characteristics and predicting a node's energy consumption. (energy consumption prediction based on interference measurement). WSN radios use nearly the same energy in all active modes of operation (send, receive, listen). Capture an interference pattern at a deployment site and use this pattern to estimate the energy consumption of a node deployed at this location in the future. [22]

[74] is an objective function for RPL that used node remaining energy as metric in the parent selection process. Energy-based OF characteristics in terms of node battery level estimation, path cost and node rank computation. A node selects the neighbor that advertises the greatest path cost value as parent. Compute the path cost (from node i to the sink) as the minimum node energy level (between parent path cost and its own energy). Uses the node's remaining energy as the main routing metric. The implementation makes use of a well-known battery theoretical model

which they estimate at runtime the node battery lifetime for routing. Energy aware routing aims to use nodes with higher remaining power level. The network should be reorganized to find more interesting nodes for routing thereby a balancing on all nodes battery level should occur. Use the rank notion to avoid routing loops. Rank, record its relative position to other nodes with regard to DODAG root.

Chapter 7

Future Work

-include gantt chart

7.1 Conclusions

7.2 Future Works

Bibliography

[1] IEEE. IEEE standard for local and metropolitan area networks–part 15.4: Low-rate wireless personal area networks (LR-WPANs) amendment 1: MAC sublayer. *IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011)*, pages 1–225, April 2012.

[2] Carlo Alberto Boano, Thiemo Voigt, Nicolas Tsiftes, Luca Mottola, Kay Römer, and Marco Antonio Zúñiga. Making sensornet MAC protocols robust against interference. In *Proceedings of the 7th European Conference on Wireless Sensor Networks*, EWSN'10, pages 272–288, 2010.

[3] M. Petrova, Lili Wu, P. Mahonen, and J. Riihijarvi. Interference measurements on performance degradation between colocated IEEE 802.11g/n and IEEE 802.15.4 networks. In *Networking, 2007. ICN '07. Sixth International Conference on*, pages 93–93, April 2007.

[4] IEEE. IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11. *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pages 1–2793, March 2012.

[5] Yafeng Wu, J.A. Stankovic, Tian He, and Shan Lin. Realistic and efficient multi-channel communications in wireless sensor networks. In *IEEE INFOCOM 2008. The 27th Conference on Computer Communications*, April 2008.

[6] Thomas Watteyne, Ankur Mehta, and Kris Pister. Reliability through frequency diversity: Why channel hopping makes sense. In *Proceedings of the*

6th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks, pages 116–123, 2009.

[7] V. Iyer, M. Woehrle, and K. Langendoen. Chrysso - a multi-channel approach to mitigate external interference. In *2011 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 449–457, June 2011.

[8] B. Al Nahas, S. Duquennoy, V. Iyer, and T. Voigt. Low-power listening goes multi-channel. In *2014 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 2–9, May 2014.

[9] 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.

[10] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level sensor network simulation with COOJA. In *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, pages 641–648, Nov 2006.

[11] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Comput. Netw.*, 52(12):2292–2330, August 2008.

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

[13] JeongGil Ko, Joakim Eriksson, Nicolas Tsiftes, Stephen Dawson-Haggerty, Jean-Philippe Vasseur, Mathilde Durvy, Andreas Terzis, Adam Dunkels, and David Culler. Beyond Interoperability: Pushing the Performance of Sensor Network IP Stacks. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 1–11, New York, NY, USA, 2011. ACM.

[14] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Comput. Netw.*, 38(4):393–422, March 2002.

[15] Alberto Cerpa, Jeremy Elson, Deborah Estrin, Lewis Girod, Michael Hamilton, and Jerry Zhao. Habitat monitoring: Application driver for wireless communications technology. *SIGCOMM Comput. Commun. Rev.*, 31(2 supplement):20–41, April 2001.

[16] 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(2):18–25, March 2006.

[17] N. Noury, T. Herve, V. Rialle, G. Virone, E. Mercier, G. Morey, A. Moro, and T. Porcheron. Monitoring behavior in home using a smart fall sensor and position sensors. In *Microtechnologies in Medicine and Biology, 1st Annual International, Conference On. 2000*, pages 607–610, 2000.

[18] Sibbald Barbara. Use computerized systems to cut adverse drug events: report. *CMAJ*, 164(13):1878, June 2001.

[19] E.M. Petriu, Nicolas D. Georganas, D.C. Petriu, D. Makrakis, and V.Z. Groza. Sensor-based information appliances. *Instrumentation Measurement Magazine, IEEE*, 3(4):31–35, Dec 2000.

[20] Gyula Simon, Miklós Maróti, Ákos Lédeczi, György Balogh, Branislav Kusy, András Nádas, Gábor Pap, János Sallai, and Ken Frampton. Sensor network-based countersniper system. In *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, pages 1–12, New York, NY, USA, 2004. ACM.

[21] Lakshman Krishnamurthy, Robert Adler, Phil Buonadonna, Jasmeet Chhabra, Mick Flanigan, Nandakishore Kushalnagar, Lama Nachman, and Mark Yarvis. Design and deployment of industrial sensor networks: Experiences from a

semiconductor plant and the north sea. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, SenSys '05, pages 64–75, New York, NY, USA, 2005. ACM.

[22] Alex King, James Brown, John Vidler, and Utz Roedig. Estimating node lifetime in interference environments. In *Local Computer Networks Conference Workshops (LCN Workshops), 2015 IEEE 40th*, pages 796–803, Oct 2015.

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

[24] Yi-hua Zhu, Wan-deng Wu, Jian Pan, and Yi-ping Tang. An energy-efficient data gathering algorithm to prolong lifetime of wireless sensor networks. *Comput. Commun.*, 33(5):639–647, March 2010.

[25] Jie Wu, Ming Gao, and I. Stojmenovic. On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks. In *Parallel Processing, 2001. International Conference on*, pages 346–354, Sept 2001.

[26] 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*, pages 10 pp. vol.2–, Jan 2000.

[27] Li Li and J.Y. Halpern. Minimum-energy mobile wireless networks revisited. In *Communications, 2001. ICC 2001. IEEE International Conference on*, volume 1, pages 278–283 vol.1, Jun 2001.

[28] Ya Xu, John Heidemann, and Deborah Estrin. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom '01, pages 70–84, New York, NY, USA, 2001. ACM.

[29] A. Bachir, M. Dohler, T. Watteyne, and K.K. Leung. MAC essentials for wireless sensor networks. *Communications Surveys Tutorials, IEEE*, 12(2):222–248, Second 2010.

[30] Lei Tang, Yanjun Sun, O. Gurewitz, and D.B. Johnson. PW-MAC: An energy-efficient predictive-wakeup mac protocol for wireless sensor networks. In *INFOCOM, 2011 Proceedings IEEE*, pages 1305–1313, April 2011.

[31] Youngmin Kim, Hyojeong Shin, and Hojung Cha. Y-MAC: An energy-efficient multi-channel MAC protocol for dense wireless sensor networks. In *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, pages 53–63, April 2008.

[32] N.A. Pantazis, S.A. Nikolidakis, and D.D. Vergados. Energy-efficient routing protocols in wireless sensor networks: A survey. *Communications Surveys Tutorials, IEEE*, 15(2):551–591, Second 2013.

[33] Omprakash Gnawali. The minimum rank with hysteresis objective function, RFC6719. `https://tools.ietf.org/html/rfc6719`, 2012.

[34] A.A. Aziz, Y.A. Sekercioglu, P. Fitzpatrick, and M. Ivanovich. A survey on distributed topology control techniques for extending the lifetime of battery powered wireless sensor networks. *Communications Surveys Tutorials, IEEE*, 15(1):121–144, First 2013.

[35] Paolo Santi. Topology control in wireless ad hoc and sensor networks. *ACM Comput. Surv.*, 37(2):164–194, June 2005.

[36] Luigi Alfredo Grieco Thomas Watteyne, Maria Rita Palattella. Using IEEE802.15.4e time-slotted channel hopping (TSCH) in an internet of things (IoT): Problem statement. `https://tools.ietf.org/html/rfc7554`, May 2015.

[37] L.F.W. van Hoesel and P.J.M. Havinga. A lightweight medium access protocol (LMAC) for wireless sensor networks: Reducing preamble transmissions

and transceiver state switches. In *1st International Workshop on Networked Sensing Systems, INSS 2004*, pages 205–208, Tokyo, Japan, 2004. Society of Instrument and Control Engineers (SICE).

[38] Ozlem Durmaz Incel, Lodewijk van Hoesel, Pierre Jansen, and Paul Havinga. MC-LMAC: A multi-channel MAC protocol for wireless sensor networks. *Ad Hoc Netw.*, 9(1):73–94, January 2011.

[39] Adam Dunkels. The ContikiMAC radio duty cycling protocol. Technical Report T2011:13. ISSN 1100-3154 `http://dunkels.com/adam/dunkels11contikimac.pdf`, 2011.

[40] Contiki. Contiki 2.6. http://contiki.sourceforge.net/docs/2.6/, Jul 2012.

[41] Contiki. Contiki 2.6 The uIP TCP/IP stack. http://contiki.sourceforge.net/docs/2.6/a01793.html, Jul 2012.

[42] Kemal Akkaya and Mohamed Younis. A survey on routing protocols for wireless sensor networks. *Ad hoc networks*, 3(3):325–349, 2005.

[43] 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*, pages 174–185. ACM, 1999.

[44] 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*, pages 56–67. ACM, 2000.

[45] David Braginsky and Deborah Estrin. Rumor routing algorthim for sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 22–31. ACM, 2002.

[46] Curt Schurgers and Mani 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. IEEE, 2001.

[47] Maurice Chu, Horst Haussecker, and Feng Zhao. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *International Journal of High Performance Computing Applications*, 16(3):293–313, 2002.

[48] Arati 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.

[49] Narayanan Sadagopan, Bhaskar Krishnamachari, and Ahmed Helmy. The ACQUIRE mechanism for efficient querying in sensor networks. In *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, pages 149–155. IEEE, 2003.

[50] 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.

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

[52] K. Sohrabi, J. Gao, V. Ailawadhi, and G.J. Pottie. Protocols for self-organization of a wireless sensor network. *Personal Communications, IEEE*, 7(5):16–27, Oct 2000.

[53] A. Manjeshwar and D.P. Agrawal. Apteen: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless. In *Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM*, pages 8 pp–, April 2002.

[54] 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.

[55] S. Lindsey, C. Raghavendra, and Krishna Sivalingam. Data gathering in sensor networks using the energy*delay metric. In *Parallel and Distributed Processing Symposium., Proceedings 15th International*, pages 2001–2008, April 2001.

[56] O. Younis and Sonia Fahmy. Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *Mobile Computing, IEEE Transactions on*, 3(4):366–379, Oct 2004.

[57] Adam Dunkels, Fredrik Österlind, and Zhitao He. An adaptive communication architecture for wireless sensor networks. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, SenSys '07, pages 335–349, New York, NY, USA, 2007. ACM.

[58] Tsvetko Tsvetkov. RPL: IPv6 routing protocol for low power and lossy networks. *Sensor Nodes–Operation, Network and Application (SN)*, 59:2, 2011.

[59] Nicolas Tsiftes, Joakim Eriksson, Niclas Finne, Fredrik Osterlind, Joel Hglund, and Adam Dunkels. A framework for low-power IPv6 routing simulation, experimentation, and evaluation. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, pages 479–480, New York, NY, USA, 2010.

[60] T Winter, P Thubert, T Clausen, J Hui, R Kelsey, P Levis, K Pister, R Struik, and J Vasseur. RPL: IPv6 routing protocol for low power and lossy networks, RFC 6550. `https://tools.ietf.org/html/rfc6550`, 2012.

[61] Simon Duquennoy, Olaf Landsiedel, and Thiemo Voigt. Let the tree bloom: Scalable opportunistic routing with ORPL. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, pages 2:1–2:14, 2013.

[62] Philip Levis, T Clausen, Jonathan Hui, Omprakash Gnawali, and J Ko. RFC6206: The trickle algorithm. https://tools.ietf.org/html/rfc6206, 2011.

[63] T.R. Jensen and B. Toft. *Graph Coloring Problems*. Wiley Series in Discrete Mathematics and Optimization. Wiley, 2011.

[64] Crossbow Technology. TelosB - TelosB mote platform. Document Part Number: 6020-0094-01 Rev B.

[65] J Vasseur, M Kim, K Pister, N Dejean, and D Barthel. Routing metrics used for path calculation in low power and lossy networks. https://tools.ietf.org/html/rfc6551, 2012.

[66] J Hui and P Thubert. Compression format for IPv6 datagrams over IEEE 802.15.4-based networks, RFC 6282. `https://tools.ietf.org/html/rfc6282`, 2011.

[67] Adam Dunkels. Full tcp/ip for 8-bit architectures. In *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, MobiSys '03, pages 85–98, New York, NY, USA, 2003. ACM.

[68] Adam Dunkels. Rime - a lightweight layered communication stack for sensor networks. In *Proceedings of the European Conference on Wireless Sensor Networks (EWSN), Poster/Demo session, Delft, The Netherlands*, January 2007.

[69] Adam Dunkels. Contiki Crash Course, October 2008.

[70] J Romkey. A nonstandard for transmission of IP datagrams over serial lines: SLIP, RFC 1055. `https://tools.ietf.org/html/rfc1055`, 1988.

[71] FIT IoT-LAB. Building Contiki's tunslip6. https://www.iot-lab.info/tutorials/build-tunslip6/.

[72] David Carels, Niels Derdaele, EliDe Poorter, Wim Vandenberghe, Ingrid Moerman, and Piet Demeester. Support of multiple sinks via a virtual root for

the rpl routing protocol. *EURASIP Journal on Wireless Communications and Networking*, 2014(1), 2014.

[73] Roman Lim, Federico Ferrari, Marco Zimmerling, Christoph Walser, Philipp Sommer, and Jan Beutel. Flocklab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks*, IPSN '13, pages 153–166, New York, NY, USA, 2013. ACM.

[74] Patrick Olivier Kamgueu, Emmanuel Nataf, Thomas Djotio, and Olivier Festor. Energy-based metric for the routing protocol in low-power and lossy network. In *SENSORNETS*, pages 145–148, 2013.

[75] Thang Vu Chien, Hung Nguyen Chan, and Thanh Nguyen Huu. A comparative study on operating system for wireless sensor networks. In *2011 International Conference on Advanced Computer Science and Information System (ICACSIS)*, pages 73–78, December 2011.

[76] Lanny Sitanayah, Cormac J. Sreenan, and Szymon Fedor. A cooja-based tool for maintaining sensor network coverage requirements in a building. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, pages 70:1–70:2, 2013.

[77] Pascal Thubert. Objective function zero for the routing protocol for low-power and lossy networks (RPL), RFC6552. `https://tools.ietf.org/html/rfc6552`, 2012.

[78] A. Sivanantha, B. Hamdaoui, M. Guizani, Xiuzhen Cheng, and T. Znati. EM-MAC: An energy-aware multi-channel MAC protocol for multi-hop wireless networks. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2012 8th International*, pages 1159–1164, Aug 2012.

[79] Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. Collection tree protocol. In *Proceedings of the 7th ACM Con-*

ference on Embedded Networked Sensor Systems, SenSys '09, pages 1–14, 2009.

[80] Asaduzzaman and Hyung Yun Kong. Energy efficient cooperative LEACH protocol for wireless sensor networks. *Communications and Networks, Journal of*, 12(4):358–365, Aug 2010.

[81] Joris Borms, Kris Steenhaut, and Bart Lemmens. Low-overhead dynamic multi-channel MAC for wireless sensor networks. In *Proceedings of the 7th European Conference on Wireless Sensor Networks*, EWSN'10, pages 81–96, 2010.

[82] R. Fonseca, O. Gnawali, K. Jamieson, P. Levis S. Kim, and A. Woo. TEP 123: The Collection Tree Protocol, Aug 2006.

[83] An-Feng Liu, Peng-Hui Zhang, and Zhi-Gang Chen. Theoretical analysis of the lifetime and energy hole in cluster based wireless sensor networks. *Journal of Parallel and Distributed Computing*, 71(10):1327 – 1355, 2011.

[84] An-Feng Liu, Peng-Hui Zhang, and Zhi-Gang Chen. Theoretical analysis of the lifetime and energy hole in cluster based wireless sensor networks. *Journal of Parallel and Distributed Computing*, 71(10):1327 – 1355, 2011.

[85] Hongbo Jiang, Shudong Jin, and Chonggang Wang. Prediction or not? an energy-efficient framework for clustering-based data collection in wireless sensor networks. *Parallel and Distributed Systems, IEEE Transactions on*, 22(6):1064–1071, June 2011.

[86] Jalel Ben-Othman and Bashir Yahya. Energy efficient and QoS based routing protocol for wireless sensor networks. *J. Parallel Distrib. Comput.*, 70(8):849–857, August 2010.

[87] I.A. Essa. Ubiquitous sensing for smart and aware environments. *Personal Communications, IEEE*, 7(5):47–49, Oct 2000.

[88] Shio Kumar Singh, MP Singh, DK Singh, et al. Routing protocols in wireless sensor networks–a survey. *International Journal of Computer Science & Engineering Survey (IJCSES) Vol*, 1:63–83, 2010.

[89] W.K.G. Seah, Zhi Ang Eu, and H. Tan. Wireless sensor networks powered by ambient energy harvesting (wsn-heap) - survey and challenges. In *Wireless Communication, Vehicular Technology, Information Theory and Aerospace Electronic Systems Technology, 2009. Wireless VITAE 2009. 1st International Conference on*, pages 1–5, May 2009.

[90] Z.G. Wan, Y.K. Tan, and C. Yuen. Review on energy harvesting and energy management for sustainable wireless sensor networks. In *Communication Technology (ICCT), 2011 IEEE 13th International Conference on*, pages 362–367, Sept 2011.

[91] James M Gilbert and Farooq Balouchi. Comparison of energy harvesting systems for wireless sensor networks. *international journal of automation and computing*, 5(4):334–347, 2008.