Improvements to End-to-End Performance of Low-Power Wireless Networks

August Betzler

Advisor: Carles Gomez Montenegro
Co-Advisor: Ilker Demirkol

Departament d'Enginyeria Telemàtica

Universitat Politècnica de Catalunya, BarcelonaTech

June 2015

Abstract

Over the last decades, wireless technologies have become an important part of our daily lives. A plentitude of new types of networks based on wireless technologies have emerged, often replacing wired solutions. In this development, not only the number and the types of devices equipped with wireless transceivers have significantly increased, also the variety of wireless technologies has grown considerably. Moreover, Internet access for wireless devices has paved the way for a large variety of new private, business, and research applications. Great efforts have been made by the research community and the industry to develop standards, specifications, and communication protocols for networks of constrained devices, we refer to as Wireless Sensor Networks (WSNs). The Institute of Electrical and Electronics Engineers (IEEE) defined the 802.15.4 standard for Wireless Personal Area Networks (WPANs). With the introduction of an adaptation layer which makes IEEE 802.15.4 networks IPv6-capable, interconnecting billions of constrained devices has become possible and is expected to become a reality in the near future. The vision that embraces the idea of interweaving Internet technology with any type of smart objects, such as wearable devices or sensors of a WSN, is called the Internet of Things (IoT).

The main goal of this thesis is the improvement of the performance of low-power wireless networks. Given the wide scope of application scenarios and networking solutions proposed for such networks, the development and optimization of communication protocols for wireless low-power devices is a challenging task: The hardware restrictions of constrained devices, specific application scenarios that may vary from one network to another, and the integration of WSNs into the IoT require new approaches to the design and evaluation of communication protocols. To face these challenges and to find solutions for them, research needs to be carried out. Mechanisms and parameter settings of communication protocol stacks for WSNs that are crucial to the network performance need to be identified, optimized, and complemented by adding new ones.

The first contribution of this thesis is the improvement of end-to-end performance for IEEE 802.15.4-based WPANs, where default parameter settings of common communication protocols are analyzed and evaluated with regard to their impact on the network performance. Physical evaluations are

carried out in a large testbed, addressing the important question of whether the default and allowed range settings defined for common communication protocols are efficient or whether alternative settings may yield a better performance.

The second contribution of this thesis is the improvement of end-to-end performance for ZigBee wireless HA networks. ZigBee is an important standard for low-power wireless networks and the investigations carried out address the crucial lack of investigation the ZigBee HA performance evaluations through physical experiments and potential ways to improve the network performance based on these experiments. Eventually, this thesis focuses on the improvement of the congestion control (CC) mechanism applied by the Constrained Application Protocol (CoAP) used in IoT communications. For the handling of the possible congestion in the IoT produced by the plethora of the devices and/or link errors innate to low-power radio communications, the default CC mechanism it lacks an advanced CC algorithm. Given CoAP's high relevance for IoT communications, an advanced CC algorithm should be capable of adapting to these particularities of IoT communications. This thesis contributes to this topic with the design and optimization of the CoAP Advanced Congestion Control/Simple (CoCoA) protocol, an advanced CC mechanism for CoAP.The investigations of advanced CC mechanisms for CoAP involve extensive performance evaluations in simulated networks and physical experiments in real testbeds using different communication technologies.

Contents

List of Figures

List of Tables

LIST OF TABLES

Glossary

6LoWPAN IPv6 over Low-power Wireless Personal Area Network

ACK Acknowledgement

AODV Ad-hoc On demand Distance Vector

AODVv2 Dynamic MANET On-demand

API Application Programming Interface

APL Application Layer

ASK Amplitude Shift Keying

BDP Bandwidth Delay Product

BLE Bluetooth Low Energy

BPSK Binary Phase Shift Keying

BU Backoff Unit

CAP Contention Access Period

CC Congestion Control

CCA Clear Channel Asessment

CCDF Complimentary Cumulative Density Function

CFP Contention Free Period

CoAP Constrained Application Protocol

CoCoA CoAP Simple Congestion Control/Advanced

CODA Congestion Detection and Avoidance in sensor networks

CON Confirmable

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance

CWL Congestion Window Limit

DAG Directed Acyclic Graph

DAO Destination Advertisement Object

dBm Decibel-Milliwatt

DIO DODAG Information Objects

DODAG Destination Oriented Directed Acyclic Graphs

DSSS Direct Sequence Spread Spectrum

DTC Distributed TCP Caching

ECN Explicit Congestion Notification

Entel Enginyeria Telemàtica

ESRT Event-to-Sink Reliable Transport

ETX Expected Transmission Count

FFD Full Function Device

FHSS Frequency Hop Spread Spectrum

FTP File Transfer Protocol

GFSK Gaussian Frequency Shift Keying

HA Home Automation

HC Hop-Count

HTTP Hypertext Transfer Protocol

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

IoT Internet of Things

IPv6 Internet Protocol version 6

kbps Kilobit per second

LAN Local Area Network

LDR Link Delivery Ratio

LLN Low-power and lossy Network

LOADng LLN On-demand Ad-hoc Distance-vector Routing Protocol Next Generation

GLOSSARY

LQI Link Quality Indication

M2M Machine-to-Machine

MAC Medium Access Control

MHz Megaherz

MPR Multipoint Relay

MQTT Message Queue Telemetry Transport

MQTT-SN MQTT for Sensor Networks

MTU Maximum Transmission Unit

NAT Network Address Translation

NON Non-confirmable

nst-AODV Not so tiny AODV

NWK Network

O-QPSK Offset-Quadrature Phase Shift Keying

OF Objective Function

OLSR Optimized Link State Routing

OSI Open Systems Interconnection

OTA Over The Air programming

Path-DR Path Delivery Ratio

PDR Packet Delivery Ratio

PHY Physical

PLF Piecewise Linear Function

PSSS Parallel Sequence Spread Spectrum

QoS Quality of Service

RAM Random Access Memory

RDC Radio Duty Cycling

RERR Route Error

REST Representational State Transfer

RFC Request For Comments

RFD Reduced Function Device

RFID Radio-Frequency Identification

ROLL Routing Over Low-power Lossy networks

ROM Read-only Memory

RPL IPv6 Routing Protocol of Low-power and lossy networks

RREP Route Reply

RREQ Route Request

RSSI Received Signal Strength Indicator

RST Reset

RTO Retransmission Timeout

RTT Round-Trip Time

RTTVAR Round-Trip Time Variance

SACK Selective Acknowledgement

SD Symbol Duration

SE Smart Energy

SNR Signal-to-Noise Ratio

SRTT Smoothed Round-Trip Time

TC Topology Control

TCP Transmission Control Protocol

TDMA Time Division Multiple Access

TLS Transport Layer Security

TLV Type-Length-Value

TSCH Time-Slotted Channel Hopping

TSS TCP Support for Sensor nodes

TTL Time To Live

UDP User Datagram Protocol

WHAN Wireless Home Automation Network

WNG Wireless Network Group

WPAN Wireless Personal Area Network

WSN Wireless Sensor Network

WWW World Wide Web

XML Extensible Markup Language

ZC ZigBee Coordinator

ZED ZigBee End Device

ZR ZigBee Router

1

Introduction

Over the last decades, wireless technologies have become an important part of our daily lives. A plentitude of new types of networks based on wireless technologies have emerged, often replacing wired solutions. In this development, not only the number and the types of devices equipped with wireless transceivers have significantly increased, also the variety of wireless technologies has grown considerably. Moreover, Internet access for wireless devices has paved the way for a large variety of new private and business applications, as well as research areas. The evolution of hardware technology, in particular the miniaturization of components, as well as the reduction of manufacturing costs have led to the development of the subclass of wireless networks of low-power, constrained devices we refer to as Wireless Sensor Networks (WSNs). Such networks are mainly composed of sensors or actuators with low-power wireless transceivers, either in the form of discrete units (wireless motes) or as embedded hardware. These devices are very constrained with regard to their memory, processing, and communication capacities.

Great efforts have been made by the research community and the industry to develop standards, specifications, and communication protocols for such networks of constrained devices. The Institute of Electrical and Electronics Engineers (IEEE) defined the 802.15.4 standard for Wireless Personal Area Networks (WPANs) [5] of wireless constrained devices and manufacturers have been providing software and hardware for a large variety of WSN-applications based on this important standard. Typical application scenarios for IEEE 802.15.4 networks are Home Automation (HA), agricultural monitoring, Smart Energy (SE) metering, and many other types of use-cases [6] that involve sensing, data collection, and remote controlling of devices. In the industry, a large variety of specifications for such networks have emerged, including ZigBee [7], which has an important share of the market [8]. With the introduction of the Internet Protocol version 6 (IPv6) [9] Low-Power Wireless Personal Area Networks (6LoWPAN) adaptation layer, which makes IEEE 802.15.4 networks IPv6-capable, interconnecting billions of constrained devices has become possible and is expected to become a reality

in the near future [10][11]. The vision that embraces the idea of interweaving Internet technology with any type of smart objects, such as wearable devices or sensors of a WSN, is called the Internet of Things (IoT). The IoT aims connecting everything with everything and it targets an automated exchange of information between devices to improve existing services or to offer new types of services.

The main goal of this thesis is the improvement of the performance of WSNs. Given the wide scope of application scenarios and networking solutions proposed for such networks, the development and optimization of communication protocols for wireless low-power devices is a challenging task: The hardware restrictions of constrained devices, specific application scenarios that may vary from one network to another, and the integration of WSNs into the IoT require new approaches to the design and evaluation of communication protocols. This thesis carries out the investigations that are necessary to face these challenges and to finds solutions for them. Mechanisms and parameter settings of communication protocol stacks for WSNs that are crucial to the network performance need to be identified, optimized, and complemented by adding new ones. The investigations carried out in this thesis cover a wide range of network types and application cases, ranging from the evaluation of generic IEEE 802.15.4-based WPANs, up to the evaluation of IoT network scenarios. In the following, the main contributions of this thesis in the field of end-to-end performance improvement for low-power wireless networks are presented.

1.1 Scientific Contributions

Fig. 1.1 gives an overview of the different types of networks and network communications investigated throughout this thesis, along with the resulting contributions. For scientific publications that resulted from this thesis are a letter-based citation style is used, while external sources are cited in a number-based style.

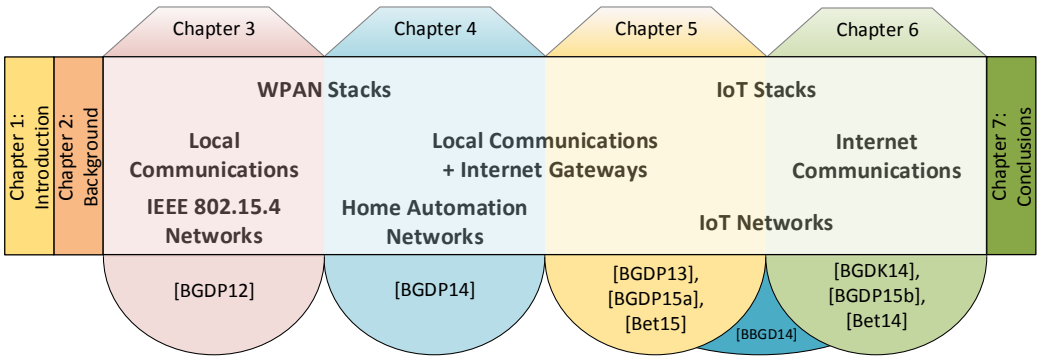


Figure 1.1: Protocol stacks, communication patterns, and types of networks considered in each of the chapters for the investigation on end-to-end performance improvement in low-power wireless networks, along with the resulting contributions.

The first contribution of this thesis is the improvement of end-to-end performance for IEEE 802.15.4-based PANs. The default parameter settings of common communication protocols are analyzed and evaluated with regard to their impact on the performance of end-to-end communications in low-power wireless networks. Evaluations are carried out in a physical testbed with 60 nodes in a multitude of different experiments that address the important question of whether the default and allowed parameter settings defined for common communication protocols are efficient or whether alternative settings may yield a better performance. The results of these evaluations have led to the publication of a conference paper [BGDP12].

The second contribution of this thesis is the improvement of end-to-end performance for ZigBee [12] wireless HA networks. ZigBee is an important standard for low-power wireless networks and the investigations carried out address the crucial lack of investigation of the ZigBee HA performance through physical experiments and by showing potential ways to improve the network performance based on these experiments. In a holistic investigation that involves an exhaustive experimental evaluations in a complex testbed setup and that covers multiple layers of the ZigBee communication protocol stack, improved mechanism and parameter settings for ZigBee HA networks, but also for IEEE 802.15.4-based networks in general, are identified. The results of this investigation, including two improved communications protocol stack settings that are derived from the evaluations, have been published in a journal paper [BGDP14].

The third and last area of contributions evolves around the improvement of end-to-end communications for the Constrained Application Protocol (CoAP) used in IoT communications. CoAP defines a simple congestion control (CC) algorithm, designed for a conservative and safe operation of the protocol in the Internet. However, for the handling of the possible congestion in the IoT produced by the plethora of devices and/or link errors innate to low-power radio communications, it lacks an advanced CC algorithm. Given CoAP's high relevance for IoT communications, an advanced CC algorithm should be capable of adapting to these particularities of IoT communications. This thesis contributes to this topic with the design and optimization of the *CoAP Advanced Congestion Control/Simple* (*CoCoA*) protocol, an advanced CC mechanism for CoAP. The investigations of advanced CC mechanisms for CoAP involve extensive performance evaluations in simulated networks and physical experiments in real testbeds using different communication technologies. Two open-source implementations of CoCoA for constrained and unconstrained devices, respectively, have been developed in the course of this thesis [Bet15, Bet14]. Further, the results of the evaluations of CC for CoAP have been published in two conference papers [BGDP13, BGDK14] and have been submitted for publication to two journals [BGDPed, BGDPew]. Above that, the work on CoCoA has resulted in the co-authorship of the CoCoA Internet-Draft [BBGD14].

1.2 Organization

This thesis is organized in a total of 7 chapters. The current chapter and Chapter 2 serve as introduction into the investigations carried out in this thesis, which are covered in Chapters 3 to 6. These chapters constitute the core of this thesis, as shown in Fig. 1.1. The concluding Chapter 7 recapitulates the findings of this thesis and provides ideas for future work. In the following, the details of the content presented in each of the chapters is given.

Chapter 2 introduces WSNs, as well as the state-of-the-art technologies and protocols used for end-to-end communications in WSNs. The information contained in this chapter is necessary to understand the basic concept of WSNs and how communications take place in such networks. Apart from giving an extensive state-of-the-art for standards and protocols used in WSNs, Chapter 2 also introduces the operating systems of constrained devices that have been used to carry out performance evaluations.

In **Chapter 3**, a detailed experimental evaluation of the end-to-end performance of an IEEE 802.15.4-based communication protocols stack is carried out in a real indoor WSN. The implementation of the communication protocol stack and its detailed analysis give insights into how the most relevant features of the stack affect performance and how performance can be improved by adjusting crucial mechanisms and parameters. Evaluations of the Medium Access (MAC), Network (NWK), and Transport layers are covered in this chapter.

Chapter 4 extends the evaluations presented in Chapter 3. A holistic analysis of performance improvement for WSNs is carried out in a complex HA network scenario that implements a ZigBee communication protocol stack. The experimental evaluation of the stack reveals that there exist parameter and mechanism configurations that have a high impact on the end-to-end network performance. The insights gained from these evaluations are used to determine two improved configurations of the ZigBee communication protocol stack that lead to important performance improvements. The evaluations cover the MAC, NWK and Application layers.

Chapter 5 widens out the field of investigation to end-to-end performance improvements for full IPv6 capable IoT networks. The investigation in this chapter focuses on the design and optimization of the CoCoA protocol, an advanced CC mechanism for CoAP. The simulation-based evaluations of the default CC mechanism and CoCoA provide detailed insights in their strengths and weaknesses and the results are used to propose an improved version of CoCoA. In the subsequent evaluations of the improved CoCoA, *CoCoA+*, it is shown that CoCoA+ meets the design criteria of an advanced CC for CoAP by always performing better than or at least similar to default CoAP.

In **Chapter 6** the improvements of the updated version of CoCoA on the performance in IoT communications are validated in experiments carried out in real IoT network setups. It is shown that CoCoA improves the performance when it is used for Internet cloud services. The design choices for CoCoA made in Chapter 5 are verified for the IoT by comparing how CoCoA performs in comparison to other well known CC

mechanisms in experiments that rely on a variety of hardware and communication technologies. The collaboration with Matthias Kovatsch from the Institute for Pervasive Computing at the ETH, Zürich, resulted in an implementation of CoCoA for Java.

Chapter 7 presents the conclusions of this thesis, summing up the results obtained during the investigation of end-to-end performance improvements of low- power wireless networks. Moreover, several proposals for future investigations are made to complement and extend the work presented in this thesis.

2

Background: Low-Power Wireless Networks

When comparing traditional, *unconstrained* wireless networks, such as Wi-Fi [13], with WSNs, differences can be found in many aspects. On the one hand, processing and communication capacities of WSNs are very limited due to the very restricted hardware capacities of constrained devices and the fact that they are communicating over lossy radio channels with small bandwidths. On the other hand, WSNs can consist of several thousands of nodes. In the IoT it has been forecast that there will be billions of devices communicating one with each other [11]. This imposes a series of challenges for WSNs that ranges from hardware design choices, over the design of protocols and communication standards, up to the deployment and layout of networks.

The chapter is structured as follows: An in-depth introduction to constrained wireless devices and WSNs is given in Section 2.1. Section 2.2 provides a general overview of the communication protocol stack for WSNs and in Sections 2.3 to 2.8, commonly used standards and protocols are introduced. Lastly, the software used for experimental and simulation based evaluations is presented in Section 2.9.

2.1 Constrained Wireless Devices

Constrained wireless devices are compactly designed, mote-like devices that are equipped with low-power radios and processors, often offering a large variety of sensors and/or actuators. A typical unit within this subclass of wireless devices disposes of relatively small amounts of memory, in terms of Read Only Memory (ROM) and Random Access Memory (RAM), and it has very limited processing capacities. The Internet Engineering Task Force (IETF) categorizes constrained devices into three different classes [14]. The device capabilities of Class 0, Class 1, and Class 2 devices are shown in Table 2.1.

Name	Data Size (e.g., RAM)	Code Size (e.g., Flash)
Class 0, C0	<< 10 KiB	<< 100 KiB
Class 1, C1	10 KiB	100 KiB
Class 2, C2	50 KiB	250 KiB

Table 2.1: Classes of constrained devices.

Class 0 devices are very constrained in their hardware and processing capacities and fulfill very basic roles as sensors or actuators. The tasks of Class 0 devices involve simple activities, such as temperature monitoring or light switching. They may be able to communicate with the Internet, however, not in a secure manner [14]. They rely on devices with higher capacities that act as gateways or proxies for Internet communication.

Class 1 devices offer around 10 KiB of RAM and up to 100 KiB of ROM. This allows them to implement communication protocol stacks for constrained devices that are designed to enable full Internet communication using protocols such as CoAP [15] without the need of additional gateways or proxies. The protocols for constrained devices that are designed by the IETF have to run on Class 1 devices.

More complex protocols or ones with large memory footprints, such as Hypertext Transfer Protocol (HTTP), Transport Layer Security (TLS) and also Extensible Markup Language (XML)-like data representations, exceed the capacities of Class 1 devices, but can be implemented on Class 2 devices that offer around 50 KiB of RAM and 250 KiB of ROM.

The investigation carried out in this thesis focuses on the improvement of protocols designed for communications in networks of Class 1 devices. The main advantage of using Class 1 devices over the other two types of classes is the capability for unrestricted (Internet) communications while operating with low memory and processing capacities, resulting in low energy consumption and low production costs.

WSNs have some major benefits, when compared to wired solutions: a simple and flexible deployment, where devices can be set up dynamically and an easy maintenance, especially considering that Class 1 devices can have lifetimes of up to several years [16], while being only powered by battery. WSNs are also often capable of building ad-hoc network topologies, thus adapting to specific networks conditions and being resilient against outages of relay devices.

However, WSNs have to cope with a series of important restrictions that affect their performance. The hardware design of low-power devices does not permit high memory capacities and powerful processing units. Main reasons for constraining devices is to achieve a low energy consumption rates and to avoid high economical costs, which is important for large scale deployments of devices. Above that, the tasks of sensing and actuating do not require powerful hardware to be carried out.

The hardware limitations not only affect the processing and memory capacities, but apply to the radio transceivers used in constrained devices, affecting the radio communications. Radio transceivers used by constrained wireless devices normally operate with limited bandwidth and at low transmission powers to reduce the energy consumption during active radio communications. These limitations mainly affect the transmission quality and range.

For battery operated devices, it is essential to maximize the battery lifetime. An efficient way to do so is to turn off their radio during idle periods. This can be achieved by synchronizing the devices in a network, determining intervals during which the devices of the network can communicate over the radio, while turning off the radio during the rest of time. Other solutions may apply desynchronized methods to save energy, where nodes enter sleeping modes and active modes independently from each other. Beacon-synchronization or radio duty cycling (RDC) are commonly used mechanisms in synchronized or desynchronized networks, respectively.

Another notable issue of low-power wireless networks is that network communications often have to deal with a high degree of packet losses. Common reasons for a high degree of losses in low-power signal transmissions are a low signal to noise ratio, signal distortions and internal or external radio interference. The radio channels in WSNs are susceptible to different physical phenomena that may increase the probability of bit errors. Such effects, for example multipath propagation and signal fading, can deteriorate the quality of a link significantly. Indoor environments, like private homes, office buildings, or factories are prone to these types of phenomena. Also, other networks sharing the same frequency band as WSNs may cause interference. As a result, the quality of links may drop at the frequencies that are affected by interference, up to the point where communications are no longer possible, depending on the degree of interference. Given the relatively high degree of losses when compared to traditional wireless networks, WSNs of constrained devices are often referred to as 'Low-power and Lossy Networks (LLNs)' [6].

The limitations of the radio capacities of constrained devices can partially be avoided by choosing a sufficiently high transmission power or increasing the time a device spends with the radio interface turned on. However, this comes at the cost of a higher energy consumption, reducing the overall battery lifetime of devices. Yet, there are some unavoidable limitations of the radio communication capacity as a consequence of the compact designs of constrained wireless devices. To remain compact, constrained devices often are equipped with integrated onboard antennas of small size. Such antennas have limited transmission capacities when compared to external antennas [17]. As a consequence of aforementioned limitations, radio transceivers used by constrained wireless devices normally do not achieve the same radio bandwidth and transmission ranges like transceivers used by non-constrained devices.

WSNs use communication technologies that often support the formation of wireless multihop topologies to cover wide distances, which is necessary to interconnect devices given the limited transmission ranges of low-power radios. In multihop networks, data

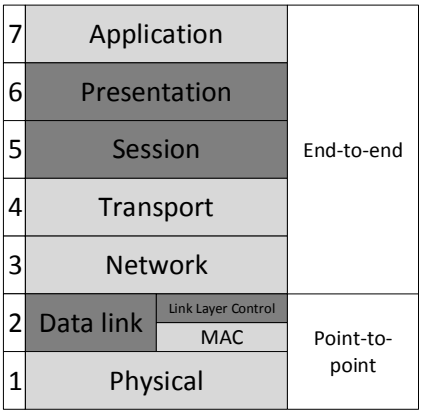


Figure 2.1: Visualization of OSI-model, highlighting the layers that are covered in this thesis.

transmissions from one node to another ('*end-to-end*') involve the collaboration of multiple protocol layers. Each layer participates in the transmission by carrying out specific tasks, such as accessing the radio channel, routing packets, or providing reliability. How these tasks are carried out depends on the choice of protocols for each layer and their settings. Each protocol defines a set of rules and mechanisms that determine its behavior during the communication process. In the WSNs, for each layer of the communication protocol stack, the number of available protocols is large, which is the result of designing and adapting protocols to different types of application scenarios, networks, or hardware.

As a result of the large variety of protocols there are many possibilities to configure a communication protocol stack in WSNs. Since the research carried out in this thesis involves the analysis and improvement of several protocol stack layers, an understanding of the basic mechanisms that are implemented at each layer is necessary. The following section introduces the layers of the communication protocol stacks that are relevant to this thesis and presents a selection of important protocols and standards for WSNs.

2.2 Wireless Sensor Networks (WSN) Communication Protocol Stacks

According to the Open Systems Interconnection (OSI) reference model [18], as depicted in Fig. 2.1, a network protocol stack is composed of up to 7 different layers. Each layer implements one or several of the aspects required for the communication process between two devices, ranging from radio channel access methods to multihop end-to-end reliability mechanisms. For each layer, a large variety of protocols and/or standards are available, suited for different network topologies, types of hardware, and application scenarios.

As mentioned in the introduction to WSNs in the previous section, protocols and standards defined for networks of non-constrained devices are not likely to match the requirements of WSNs of constrained devices. Protocols designed for non-constrained devices may require large amounts of memory in terms of ROM and RAM that are not available to Class 1 or Class 2 devices. Moreover, they may not be suited for typical communication technologies and data traffic patterns of WSNs. Further, the peculiarities of WSN communications, including high bit error rates, limited radio bandwidth, and varying link qualities, are often not taken into account in the design of protocols for networks of unconstrained or wired devices.

To fit the specific requirements of constrained networks, standards like IEEE 802.15.4 [5] have been designed. This important standard in particular, defines a physical layer (PHY) and a MAC layer for WSNs. Similarly, on higher layers of the communication protocol stack, there exist a variety of protocols designed for WSNs, such as the novel CoAP. There also exist lightweight versions of protocols originally designed for unconstrained devices (for example the not so tiny Ad-Hoc On-Demand Distance Vector (nst-AODV) [3]). Protocols for WSNs are designed to be memory and processing efficient, providing mechanisms that are tailored to the special characteristics in WSN communications.

In each of the following sections a different layer of the communication protocol stack is covered. The layers are covered from bottom to top, introducing protocols and standards for each layer that are well established and relevant for the research carried out in this thesis (Sections 2.3 to 2.7). Following these sections dedicated to single layer protocols, multi-layer protocols and specifications like ZigBee that cover several layers of the protocol stack are introduced (Section 2.8).

2.3 Physical Layer and Medium Access Control Layer: IEEE 802.15.4

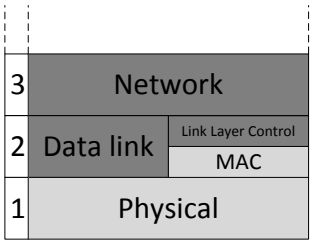


Figure 2.2: Section 2.3 focuses on the PHY and MAC layers of the OSI-model.

The evaluations carried out in this thesis mostly focus on communications that rely on the IEEE 802.15.4 standard, which is the primary communication standard for networks of constrained devices. Alternatives to IEEE 802.15.4, such as Z-Wave [19],

BLE [20], or Wi-Fi (IEEE 802.11) [13], are introduced later, including a comparison of their key features (see Section 2.3.5).

The IEEE has defined the IEEE 802.15.4 standard for low-power wireless networks, which includes a physical and medium access control layer, located at the bottom layers of the OSI reference model (Fig. 2.2). The IEEE 802.15 specifications are designed for Wireless Personal Area Networks (WPANs). The physical layer for wireless devices defines physical and electrical processes for generating and interpreting modulated radio signals. The IEEE 802.15.4 standard for low-rate WPAN defines which modulation schemes and frequencies may be used by the devices forming the network [5]. Table 2.2 shows possible configurations.

Table 2.2: Overview of the operational modes of the IEEE 802.15.4 PHY layer. The 868 MHz band is used in Europe, while the 915 MHz band is used in the United States of America.

Freq. Band (MHz)	Modulation	Max. Bitrate (kbit/s)	Spread. Method
868	BPSK	20	Binary DSSS
915	BPSK	40	Binary DSSS
868	ASK	250	20-bit PSSS
915	ASK	250	5-bit PSSS
868	O-QPSK	100	16-array orthogonal
915	O-QPSK	250	16-array orthogonal
2400	O-QPSK	250	16-array orthogonal

The worldwide commonly used modulation scheme for IEEE 802.15.4 is the Offset-Quadrature Phase Shift Keying (O-QPSK) modulation at 2400 MHz. Direct Sequence Spectrum Spread (DSSS) is additionally applied to reduce the susceptibility to bit errors. Other modulation schemes at 868 MHz (EUR) and 915 MHz (USA) represent alternative configurations with other data rates. At 2400 MHz, IEEE 802.15.4 allows operating on 16 different channels. They are arranged symmetrically around 2442.5 MHz, with a width of 2 MHz each and an interspacing of 5 MHz and be calculated as detailed in Equation 2.1.

$$F_c = 2405 + 5(k - 11) \text{ in MHz, for } k = 11, 12, ..., 26 \text{ (see Fig. 2.3)} \tag{2.1}$$

Transmitting at the given frequencies and with the given modulation allows data rates up to a theoretical maximum of 250 kbit/s. However, IEEE 802.15.4 shares the frequency band with other widely used standards such as IEEE 802.11, Bluetooth and also operational frequencies of devices such as microwaves and wireless telephones. Sharing the frequency spectrum may be the cause for interferences that may lead to a drop of the real achievable performance.

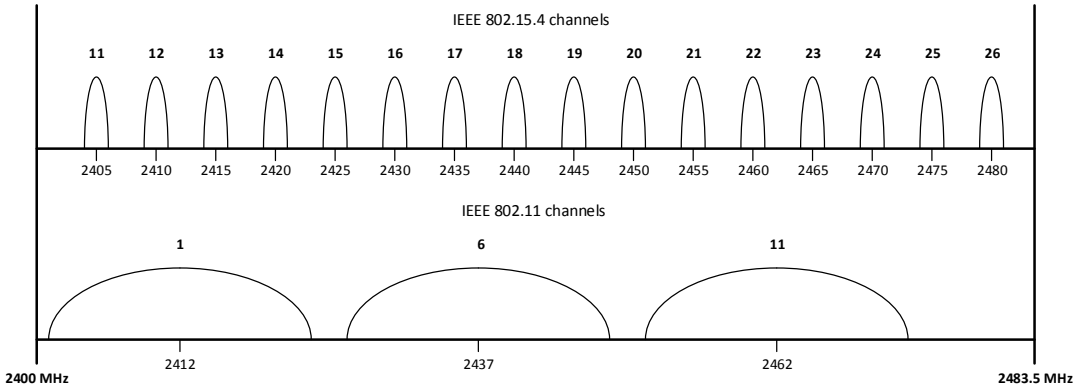


Figure 2.3: A visualization of the IEEE 802.11 and IEEE 802.15.4 channels at 2.4 GHz.

In a WSN, all devices need to use the same PHY configuration to be able to communicate. While this means that all nodes of such WSNs are configured equally in terms of operating frequency and modulation schemes, IEEE 802.15.4 devices may be split into two main types, which are Full Function Devices (FFDs) and Reduced Function Devices (RFDs). FFDs may be PAN coordinators, simple coordinators or normal devices. RFDs are only allowed to be normal devices. Full functional devices can communicate with all other devices in the network, while RFDs only may communicate with FFDs. Further, RFDs are only assigned basic tasks and represent simple entities, as switches or sensors, which allows them to be more restricted in terms of hardware capacities and processing power. Due to the limitations of RFDs, they are predetermined to be battery-powered devices.

The IEEE 802.15.4 MAC sublayer was introduced by IEEE to control the concurrent access to the channel by several devices in local networks. In the following the mechanisms of the IEEE 802.15.4 MAC layer are presented.

2.3.1 Beacon-enabled/Beacon-less Transmission Mode

In IEEE 802.15.4 there exist two possible configurations that define when nodes may access the channel to transmit data. The first is the unslotted (beacon-less) transmission method, where devices inside a WSN are not synchronized and a node may initiate a transmission at any time. In this unslotted mode of transmission, nodes compete with each other for access to the radio medium.

The other method of transmission is the slotted (beacon-enabled) transmission, where time is divided into so called 'super frames' by the PAN coordinator, the device that establishes the network and coordinates it. Devices that are interested in transmitting may either transmit in the contention access period (CAP) where devices are competing for the channel or the contention free period (CFP), where time slots for transmissions are specifically assigned to certain devices. Coordinating the nodes in

such a way allows the devices to transmit without contention and moreover, devices may sleep during inactive time periods to save energy and extend battery lifetimes.

2.3.2 Channel Access Control

To manage the channel access in the beacon-less transmission and during the CAP, the IEEE 802.15.4 MAC layer implements a Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) mechanism. In the following the functional descriptions are restricted to the beacon-less transmissions.

Before sending data, a device senses the wireless channel for any other transmitting devices. This process is called Clear Channel Assessment (CCA). The sender only proceeds to the transmission phase, if the channel is found to be free. In IEEE 802.15.4, there are three different modes for CCA to decide when the channel is considered to be busy:

1. 'Energy above threshold' – CCA reports a busy medium, if energy is measured above a certain threshold. For the CC2420 Radio used in TelosB motes this would be by default -77 dBm [21].

2. 'Carrier sense only'– CCA reports a busy medium, if it detects a signal compliant with the IEEE 802.15.4 standard, independent from the measured energy.

3. 'Carrier sense with energy above threshold' - CCA reports a busy medium if an IEEE 802.15.4 signal is detected and an energy level above the threshold is measured.

If the channel is busy, the MAC layer initiates a backoff-timer and starts sensing the carrier again after its expiration. This backoff is by default a random time interval between 0 and 7 backoff units (BUs), each lasting 20 symbol durations (SD), corresponding to 320 μs at the default frequency of 2.4 GHz. Each consecutive time the CCA mechanism detects a busy channel state, the backoff interval is doubled. After several attempts (default = 4 [5]), the transmission is considered to be not possible and the packet is dropped. If the carrier sensing finds the channel to be available, the frame is transmitted.

The standard CSMA/CA mechanism used in IEEE 802.15.4 tries to avoid collisions in form of packet collisions by sensing the channel before sending and only allowing a node to send if the channel is found to be clear. However this is not a definitive solution, as there may be multiple nodes that find the channel idle and decide to transmit at the same time. If two nodes that are not in direct radio range (that means more than one hop away), sense the channel, for instance they will never find the channel busy if one of them is transmitting. Seeing the channel free, even though another node is transmitting could lead to a situation where transmissions collide at a third node which is trying to receive one of the transmitted frames. This is called hidden terminal problem and is shown in Fig. 2.4 and is one of the main causes for packet collisions

in IEEE 802.15.4 networks. Nodes A and C may be transmitting at the same time to node B, as they both are not in each other's transmission range and therefore will not find the channel busy, even if one of them is transmitting.

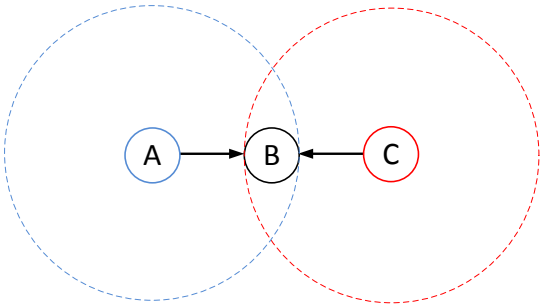


Figure 2.4: Hidden terminal effect, where three nodes are involved.

On the contrary, the IEEE 802.15.4 CSMA/CA mechanisms may prevent nodes from sending data that would not collide in the exposed terminal situation. Figure 2.5 illustrates a setup for the exposed terminal problem to happen. Nodes B and C could prevent each other from sending data to A and D correspondingly, if they sense the channel busy, as one of them is transmitting. However, in this case it would not matter, if they transmit simultaneously, as the destinations A and D are lying outside of the transmission range of C and B, respectively.

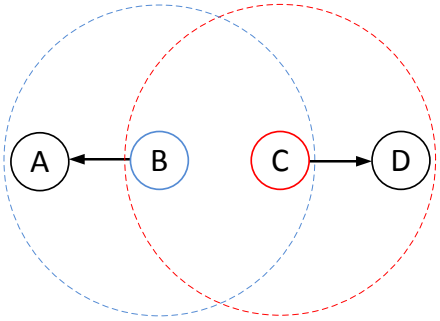


Figure 2.5: Exposed terminal effect, where four nodes are involved.

In Enhanced-CSMA [22], a modified version of the CSMA mechanism used in IEEE 802.15.4, the hidden terminal effect is taken into account. As previously stated, the results of a CCA-scan at a sending node do not have to coincide with the results a CCA channel scan would provide at a receiving node. E-CSMA empirically generates success probability distributions per receiver and correlates it with the locally observable channel conditions at the transmitter. With this mechanism a balance between success probability with local queue drop probability is achieved, increasing the overall performance.

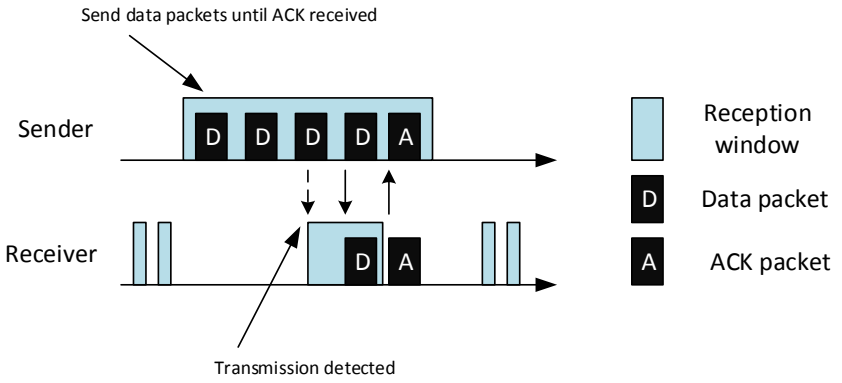


Figure 2.6: RDC mechanism in ContikiMAC [1].

2.3.2.1 Radio Duty Cycling (RDC)

The beacon-enabled transmission mode coordinates devices in such a way that they can turn off their radio during phases of radio inactivity in order to save valuable energy, while assigning them slots during which they can transmit their data.

To reduce the energy consumption in beacon-less networks, a mechanism that is not specified in the IEEE 802.15.4 standard can be applied: RDC. Generally, in this operational mode, devices turn off their radio transceivers during large periods of time (for example 99% of the time, which equals a 1% duty cycle ratio). The transceivers are turned on periodically to check for a possible ongoing transmission. A sending device may transmit data at any time and requires mechanisms like a preamble or the repeated transmissions of the data packets to indicate that it wants to transmit data to other devices. The data packet then can be delivered to a receiver once it detects the preamble or the ongoing data transmissions in one of its duty cycles.

An example of this behavior is given in Fig. 2.6. It shows the basic RDC behavior used in the ContikiMAC [1] that is applied in some of evaluations carried out in this thesis. In ContikiMAC a sender repeatedly transmits a data packet until the receiver detects the ongoing transmission of the data packet during the periodically repeating listen periods. Once the ongoing transmission is detected, the radio transceiver of the receiver node is kept on until it receives the data packet and replies with an ACK. With the reception of an ACK, the sender node stops strobing the data packet and turns off the radio.

RDC is not compatible with the IEEE 802.15.4 MAC layer specification, yet it can be applied in networks with IEEE 802.15.4 PHY layer using an alternative MAC implementation. Apart from the ContikiMAC used as a MAC layer implementation that applies RDC, there exists a large variety of other MAC layer implementations for IEEE 802.15.4 networks, covering different approaches to RDC. A survey of possible approaches is done in [23] and methods for energy efficient implementations of protocols for constrained devices are summarized in [24].

2.3.3 One-hop Reliability

IEEE 802.15.4 gives the option to enable reliable transmissions, in which case the transmitter of a data frame waits for a MAC layer Acknowledgement (ACK) after the transmission. An ACK is sent immediately after the reception of the data frame and a turnaround time T_{ACK} of 12 SDs, during which the transceiver changes from the listen to send mode.

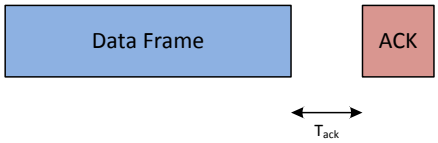


Figure 2.7: Acknowledgement procedure as defined in IEEE 802.15.4 .

A MAC layer ACK contains the sequence number used in the data frame to make it associable to the data frame. It does however not use a source and destination address. If the transmitter of the data frame does not receive an ACK after 54 SDs, it assumes the data frame was not received correctly. In this case, if it still may use a retransmission, it sends the data frame again after going through the entire CSMA/CA procedure. If all available retransmissions are already spent, the node gives up sending the frame. The default maximum amount of allowed retransmissions is 3. If no ACKs are enabled or if the destination address is the Broadcast address 0xffff, no ACK is expected and the transmitting device does not know if a transmission was successful (unless it uses a sniffing mechanism to detect if the frame was forwarded by the next node).

2.3.4 IEEE 802.15.4 Versions

There exist different amendments to the IEEE 802.15.4 standard that modify or extend certain mechanisms from the base specification, in some cases adding new ones. The alternative versions of IEEE 802.15.4 are identified by a letter attached to the end of the name of the standard (IEEE 802.15.4x, where x is the letter determining the amendment). Support for Radio-Frequency Identification (RFID) is added in IEEE 802.15.4f, an extension for low-data-rate, wireless, smart metering utility networks is added in IEEE 802.15.4g, and the Time-Slotted Channel Hopping (TSCH) replaces the original IEEE 802.15.4 MAC layer in IEEE 802.15.4e, to name some of the versions of IEEE 802.15.4 that are available. The standard that is used for the investigation carried out in this thesis is the widely used IEEE 802.15.4-2006, the revised version of the first specification of the basic standard (IEEE 802.15.4-2003).

2.3.5 Alternatives to IEEE 802.15.4

While the IEEE 802.15.4 is the default standard used in WSNs, there exist alternatives, like Z-Wave, Bluetooth Low Energy (BLE), and IEEE 802.11 adaptations for low-power networks. While these alternatives are also suitable to WSNs, this thesis only focuses on the work with the IEEE 802.15.4 standard. The choice of communication technologies for WSNs depends on the specific requirements, like the intended network topology and the use case. In the following, the main application scenarios for the alternative communication technologies are given, along with technical details.

Commercial standards normally are meant to be used for a specific class of devices with non-IEEE 802.15.4 compliant physical layer and are designed for particular types of environments. An example would be the PHY and MAC layers used by the Z-Wave nodes. The standard defined in Z-Wave only is applied on Z-Wave devices, which are basically remote controlled HA devices. Other low-power technologies that can be used for communications in networks of constrained devices are BLE and Wi-Fi [13, 20]. BLE uses the same 2.4 GHz radio channels as Bluetooth and is designed for energy efficient one-hop connections between two devices, for example a connection between a smartphone and a health-measuring gadget. In contrast to normal Bluetooth, BLE transmits very short data packets and it applies duty cycling in order to achieve very low energy consumption. Wi-Fi is mainly used for high-bandwidth communications in networks of unconstrained devices, such as Local Area Networks (LANs). While this technology per se is not energy efficient and running it in networks of constrained devices would deplete the batteries of the participators quickly, it can be tweaked to reduce its energy consumption drastically. The modified Wi-Fi ('low-power Wi-Fi') minimizes energy consumption during idle periods by introducing RDC, while being compatible with 'normal' IEEE 802.11 devices. A main advantage of low-power Wi-Fi is the ease of incorporation of new devices into existing Wi-Fi infrastructure. An overview of the crucial features each of the different communication technologies is given in Table 2.3.

Table 2.3: Key performance feature comparison of different wireless communication technologies used in the ambit of low-power wireless networks.

	IEEE 802.15.4	Z-Wave	BLE	IEEE 802.11
RF Band (MHz)	868/915/2400	868/908/2400	2400	2400
Bitrate (kbps)	20/40/250	9.6/40/100	1000	$\leq 54 \times 10^3$
Receiver sensitivity (dBm)	$\leq$ –85 at 2.4 GHz $\leq$ –92 at 868/915 MHz	–101 (at 40 kbps)	$\leq$ –70 (required) –87 to –93 (typical)	–68
Max. msg. size (bytes)	127	64	47	2346
Hop limit	30/10 (mesh/tree)	4	1	1

2.4 Adaptation Layer: IPv6 Over Low-Power Wireless Personal Area Network (6LoWPAN)

Since IPv6 requires a Maximum Transmission Unit (MTU) of at least 1280 bytes and this surpasses the available size of IEEE 802.15.4 frames (127 bytes), fragmentation of packets might be necessary. In the worst case, an uncompressed header of an IPv6 packet, including transport and IEEE 802.15.4 security overheads, leaves 33 bytes from 127 bytes for actual application layer data. The 6LoWPAN adaptation layer has been specified by the IETF to allow the transmission of IPv6 packets over networks of constrained devices using IEEE 802.15.4 (and other technologies like BLE [25]). 6LoWPAN defines frame formats and mechanisms that make the transmission of IPv6 packets possible. This includes header compression to reduce the overhead caused by the relatively large IPv6 headers. Moreover, packets are fragmented and reassembled if the IEEE 802.15.4 frame sizes are exceeded.

Two methods of routing can be applied in 6LoWPAN: Route Over and Mesh Under. In Mesh Under, frames are treated all equally, independent from their IPv6 headers, by using an additional header that allows the nodes to locally forward packets through the PAN. The nodes are therefore not aware of the actual IPv6 packet header and routing decisions are taken below IP. On the other hand, in Route Over (IP-routing), the IPv6 packets are reconstructed at each node by reassembling all the fragments belonging to a single packet.

2.5 Network Layer: IPv6 and Routing Protocols

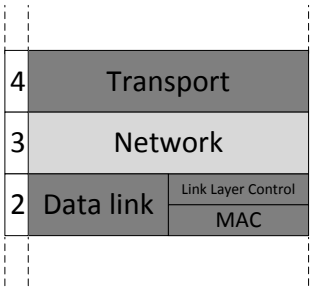


Figure 2.8: Section 2.5 focuses on the NWK layer of the OSI-model.

The NWK layer is responsible for network routing and for the logical addressing inside a WSN. In networks where the two endpoints of a transmission are more than one hop away from each other, the routing mechanism is responsible to find at least one multihop-path between them. Intermediate nodes take on the role of packet forwarders, also called *relays*. When forwarding a packet, it is not passed on to layers above the NWK layer. Instead, a forwarding node determines the next node on the way to the

destination and hands the packets over. The way of determining a route, storing the routing information and maintaining it depends on the routing protocol used.

Examples for routing protocols designed for WSNs are IPv6 Routing Protocol for Low-power and lossy networks (RPL) [26], SPIN [27], COUGAR [28], CADR [29] or GAF [30], to name just a few. However, since for the work carried out in this thesis one-to-one connections need to be established, most of the listed routing protocols are not suitable since they are mainly designed for many-to-one data connections. In such scenarios, sinks collect data from different nodes of the WSN. A survey of these protocols is done in [31]. From the routing protocols listed, RPL and AODV play an important role for this thesis, as they are the IETF standard protocol designed for the IoT and the protocol used in ZigBee, respectively.

The NWK layer also takes care of how nodes are addressed, since a node inside a network needs to be uniquely identifiable. This can be a two byte short address in a local PAN or an IPv6 address for networks with Internet connectivity, to name two examples. The NWK is also responsible of fragmentation and reassembly of packets that may not fit into one single frame. The reconstruction of the packet may happen at the destination node or at intermediate nodes depending on the protocol used. The common network layer protocol used to interconnect devices over the Internet is IP (this includes IPv4 and IPv6).

In the following, an introduction to IPv6 and routing metrics is done, followed by an overview of commonly used routing protocols that are suited for one-to-one and many-to-one connections.

2.5.1 IPv6

The significant growth of devices connected to the Internet over the last decades caused the address space of IPv4 [32] to almost reach its depletion. In IPv4, the maximum number of addresses is determined by 32 bit address fields, thus providing up to 2^{32} unique IPs. As a solution to the predictable issue of address depletion, IPv6 [9] was designed by the IETF. It increases the address space to 2^{128} addresses and provides several other features, such as stateless address autoconfiguration, where devices entering an IPv6 network automatically request IP configuration parameters from a router in the network. IPv4 and IPv6 are not interoperable, thus requiring mechanisms to translate from one protocol to the other. Possible methods to achieve interoperability are packet tunneling or network address translation (NAT).

The large address space provided by IPv6 is a basic requirement for the IoT, where up to billions of devices will be interconnected using this standard. However, to enable IPv6 for constrained devices in IEEE 802.15.4 networks, the 6LoWPAN adaptation layer is necessary (as explained in Subsection 2.4). First, the header compression applied by 6LoWPAN reduces the default IPv6 header size (see Fig. 2.9) from 40 bytes to a minimum of 7 bytes. This leaves more space for payload in IEEE 802.15.4 frames. Second, IPv6 requires an MTU of 1280 bytes, while an IEEE 802.15.4 frame has a size

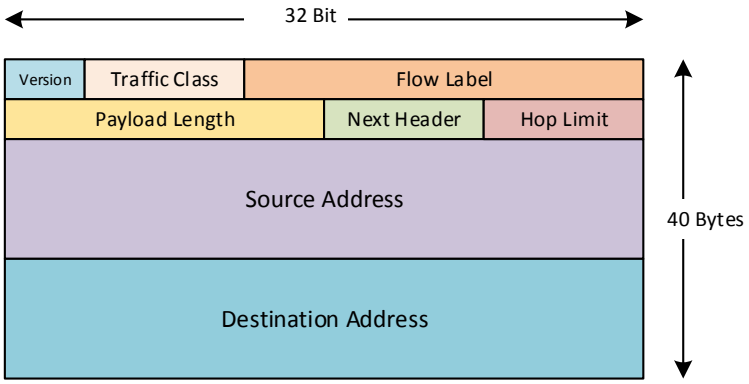


Figure 2.9: Structure of an IPv6 header.

of 127 byte. To still be able to transmit packets up to the MTU size, 6LoWPAN applies fragmentation.

2.5.2 Routing Metrics

The routing metric is an essential part of any routing mechanism. It defines a criterion that determines how the cost of a path is calculated. A 'good' path with a low cost is preferred over a 'bad' path with a higher cost. When choosing a metric to be used for routing, it is necessary to analyze the needs of the user application and which criterion is intended to be optimized. Possible criteria include a small path-delay, a high reliability (or both), low energy consumption, etc. Depending on the chosen metric, the paths chosen by the routing protocol may change significantly, even in one and the same network. A summary of commonly used metrics can be found in [33]. From these metrics, two are used by a large variety of protocols and are thus of major relevance for this thesis: the hop-count (HC) and the Expected Transmission Count (ETX) metrics.

The HC metric is an easily implementable, one-dimensional metric, where each hop on the path between source and destination is added to the total cost for the path. This metric chooses the route with the lowest number of hops, assuming that a lower amount of hops benefits the overall path delay or may even provide the lowest energy consumption since the lowest number of nodes is involved in the transmission.

However, a short path is of no use, if the Link Delivery Ratio (LDR) of its links is rather low. A low LDR results in a higher amount of losses and therefore requires more retransmissions at the MAC layer or even at higher layers, if end-to-end reliability is used. Therefore it may be necessary to estimate the LDR of the links during the route establishment process. A popular metric that optimizes the routes according to this criterion is the ETX metric. It uses the LDR of each hop along a path to calculate the expected amount of transmissions necessary to successfully transmit data over one hop

[34]:

$$\text{ETX} = \frac{1}{d_f * d_r} \tag{2.2}$$

The expected LDRs are represented by forward d_f and d_r reverse delivery ratios. The formula assumes that all transmissions are independent from each other and that each transmission can be seen as a Bernoulli-trial. To be able to calculate the ETX, knowledge of a link in both directions is necessary. This is a difficult task, as the topology of the network and therefore the link properties may be unknown at the time of route establishment. The calculation of the LDR requires knowledge about successful frame transmissions. The exchange of numbered packets or another mechanism is therefore necessary. If such statistics are not available at the time of route discovery, alternative methods need to be applied. One method is to use the Received Signal Strength Indication (RSSI) or the Link Quality Indication (LQI) to do an estimation of the LDR. In [35] a formula is proposed to map a single LQI measurement to LDR. The cited paper also shows that using the LQI metric called Path Delivery Ratio (Path-DR) yields better results than simply using a HC metric.

In Chapters 3 and 4 the effect on the performance of routing protocols using HC, ETX and Path-DR metrics are evaluated and compared. In the following, a selection of routing protocols used in WSNs is presented.

2.5.3 Ad-Hoc On-Demand Distance Vector (AODV)

The Ad-hoc On-demand Distance Vector (AODV) routing protocol was designed for mobile nodes in ad-hoc networks, but also found its applications in static ad-hoc networks [36]. ZigBee, as an important representative of protocols stacks designed for WSNs, bases its routing on AODV in mesh mode. Since AODV is a reactive routing protocol, it is able to find routes between two arbitrary nodes of the network in an ad-hoc manner, i.e., at any point of time when a transmission of data from one node to another is necessary. To establish and maintain routes, AODV defines several types of control messages: Route Requests (RREQs), Route Replies (RREPs) and Route Errors (RERRs). While RREQs and RREPs are used to find routes, RERRs are used to mark the invalid routes.

The basic operation of AODV for a route creation consists in the source node broadcasting a RREQ message that is re-broadcast by nodes that receive the RREQ. At each node, the previous hop taken by the RREQ-packet is stored in a routing table. Information about the RREQs is stored in a cache to avoid that already forwarded RREQs may be forwarded again. The process of forwarding RREQs repeats, until the destination node receives one of the RREQs or if the ring search counter (TTL_START[1]) expires [36]. The TTL_START is determined at the source node and indicates the maximum number of hops that are allowed between source and destination. If no routes are found with the initial allowed distance of 1 hop, the value is increased by TTL_INCREMENT

[1] Time To Live

(the default is 2) and the route discovery is repeated up to RREQ_RETRIES times (the default is 2). Upon reception of a RREQ, the destination node uses a certain metric to evaluate if the route found is adequate. The default metric used by AODV is the HC metric that assigns a cost to the route equal to the number of hops between the two endpoints of the transmission.

If the route found is accepted by the destination node, it prepares a RREP, which is then propagated in unicast-mode back to the source node by using the information about the reverse route stored at each node. Also, in this procedure each node stores the address of the previous hop of the RREP, adding another routing table entry. The routing entries added at each node of the path for both directions may then be used to exchange data between the two endpoints. As soon as the RREP arrives at the source node, the route is established. To avoid loops in the route creation process, sequence numbers are used to differentiate older routes from the newer ones. When forwarding a RREQ, it may happen that a node along the way already knows a valid route to the destination. In this case, the intermediate node may respond directly to the request and transmit a RREP in the direction of the node that is requesting the route.

If a link break along active routes is detected a route repair is done either by the source node or optionally by an intermediate node. Route repairs are used if nodes lose their connection with neighboring nodes due to environmental changes or physical movements. During a route repair the same process used to find a new route is applied. If a route repair is not successful, a RERR is sent, which alerts precursor nodes of the unreachable destination about the link break. If only one precursor is affected, a unicast RERR may be sent, else a broadcast is used.

2.5.3.1 Not So Tiny AODV (nst-AODV)

The Wireless Network Group (WNG) in the 'Network Engineering Department' (Entel) developed *Not So Tiny AODV* (nst-AODV) to better fit the requirements of constrained devices by reducing the amount of ROM and RAM required [3]. To achieve this, some of the basic AODV functionality has been eliminated or reduced in its scope. nst-AODV gets rid of the precursor lists and detects link breaks after a certain amount of retries have been spent, instead of using 'Hello-messages' to inform periodically about the state of links. However, nst-AODV is 'not so tiny', since it was developed on the basis of TinyAODV [2], by adding functionality to it, which provided better performance at the expense of greater implementation complexity.

2.5.4 Dynamic Mobile Ad-Hoc Network On-demand (AODVv2)

The Dynamic Mobile Ad-Hoc Network On-demand (AODVv2, previously called DYMO) routing protocol is AODV-based and maintains its basic operation principles, adding a few changes [37]. It uses similar route discovery and route maintenance procedures as

[2]Several implementations available online.

in AODV. During route discovery, each intermediate node receiving a RREQ adds the usual routing information for the path to the source of the RREQ, but it also adds its own information to the RREQ and forwards it. This feature is called path accumulation. Other nodes may acquire and use this information by sniffing for routing packets, to discover routes to other nodes without doing route discoveries. A route's lifetime in a nodes routing table is extended each time a packet is forwarded along the route. If there is no valid routing entry in a node's routing table, then a RERR notifies the source of the data transmission about this incident and it deletes its routing entry. If then data needs to be sent over the deleted route, a route discovery is started. This means that no local route repair is possible in AODVv2.

2.5.5 LLN On-demand Ad hoc Distance-vector Routing Protocol (LOADng)

Another AODV-based routing protocol is introduced with the LLN On-demand Ad hoc Distance-vector Routing Protocol Next Generation (LOADng)[38]. It differs in some points from AODV, which are:

- When discovering a route and forwarding RREQs, no intermediate node may answer with a RREP. Only the destination node is allowed to answer to the RREQs.
- No precursor list is maintained; when a RERR is sent, it is directly sent only to the source node.
- Optimized flooding is supported, an option that allows to use alternative flooding mechanisms, like the one proposed in the Request for Comments (RFC) 6621 [39].
- LOADng supports different address lengths, ranging from 1 and 2 octet addresses up to full 16 octet IPv6 addresses, as long as all devices in the routing domain use the same type of address.
- The packet format of control messages is as described in RFC 5444 [40], which can include Type-Length-Value (TLV) elements.

The latter two features are not available in nst-AODV.

2.5.6 Optimized Link State Routing (OLSR)

The Optimized Link State Routing (OLSR) [41] is a proactive IP routing protocol, meaning that route information is gathered and maintained periodically by the nodes so that upon requiring a route they already are available. OLSR uses *hello* and *topology control*(TC) messages. Hello messages are used to determine link status information and neighbor nodes. Based upon the information gathered with hello messages, a set of Multipoint Relay (MPR) Selectors is built that determine a set of MPRs.

TC messages are broadcasted periodically across the network to spread the information about the advertised neighbors of a node. Instead of all nodes participating in the flooding of broadcast control messages all across the network as done in other proactive protocols, only the MPRs forward the broadcast messages to reduce message overhead. The link state information required to determine routes according to the HC metric is only generated by MPRs and the minimal set of link state information generated for the TC messages needs to declare the links to the MPR selectors. OLSR is designed for large and dense networks, where the use of MPRs increases the efficiency of the routing protocol.

2.5.7 IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL)

RPL is the routing protocol to be used in 6LoWPAN-based WSNs that has been designed and developed by the IETF *Routing Over Low-power Lossy networks* (ROLL) working group. The ROLL working group analyzed several protocols for their suitability in 6LoWPAN networks. Since, in their current state, none of the analyzed protocols (AODV, OLSR, etc.) seemed to fit the requirements of a routing protocol for WSNs, RPL was developed [42].

RPL constructs Destination Oriented Directed Acyclic Graphs (DODAGs). In contrary to a Directed Acyclic Graph (DAG), a DODAG may only have one destination node, which would act as a sink node and/or a gateway in other networks. Each node in the WSN periodically informs about its state by sending DODAG Information Objects (DIOs). A DIO contains information such as the rank of the node inside the WSN, a DODAG-ID, etc. Each node in the network may use this information to calculate its best parent node by using a distance vector algorithm. The rank of a node indicates the distance from a node to the destination node. Loop-freedom is obtained by assuring that any node has a higher rank than its parents. It is possible for nodes to have multiple parents. The time between two transmissions of a DIO is calculated with the Trickle algorithm [43].

Since nodes may not be updating their information regularly as they may have low duty cycles or use a large update period, a DODAG Information Solicitation (DIS) may be used to explicitly request a DIO from a node. This way all the nodes maintain and update their information and are able to reach the destination node. Since only the information about parent nodes is updated with DIOs, a data transmission in the other direction (from a parent node to one of its child nodes) requires another type of information message. This information is disseminated by nodes in a Destination Advertisement Object (DAO). A DAO contains the information about the nodes that can be reached through the node sending the DAO. A parent node that gets this information forwards it and includes itself in the list of reachable nodes. This procedure is repeated, until the destination node is reached. The cost of a route may be calculated with different metrics, in RPL referred to as Objective Function (OF), for example the

HC metric (see [44], [45], and [46] for details). One-to-one communications with RPL are not optimal, that is why P2P-RPL has been designed [47].

2.6 Transport Layer Protocols

For end-to-end data transmissions, where the lossless delivery of data packets from the source to the destination nodes is essential, end-to-end reliability is necessary. Above that, in some networks and for certain applications, flow control (or congestion control) may be necessary to avoid exceeding the radio bandwidth or storage capacity of nodes. Such features are normally provided by the transport layer protocols, located between the Session and NWK layers (see Fig. 2.10)[3]. This section explains how end-to-end reliability as well as congestion control can be implemented and provides an overview of transport layer protocols.

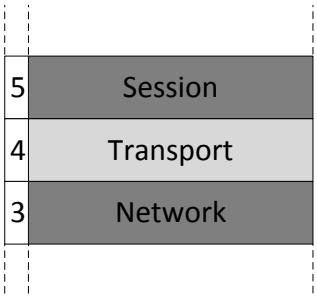


Figure 2.10: Section 2.6 focuses on the Transport Layer of the OSI-model.

End-to-end Reliability

When transmitting critical data, such as an alarm or a heartbeat signal, a high degree of reliability may be required. On the other hand, periodically collected sensor data may not require reliability at all. Transport layer protocols like Transmission Control Protocol (TCP) implement end-to-end reliability, but it can also be provided by application layer protocols like CoAP (see 2.7.3). End-to-end reliability is commonly achieved by using end-to-end ACKs. This implies that the source node expects an ACK from the destination node, confirming the reception of a data packet. If the source node does not get an ACK after a certain amount of time, it assumes that the corresponding data packet was not received correctly by the destination and the data is retransmitted. The time a node waits for an ACK before considering a transmission to have failed is called retransmission timeout (RTO). If the RTO expires and no ACK was received, then the source assumes its data packet seems to not have been correctly received by the destination and the data packet is sent again. This procedure may be

[3]Please note that the Session layer is not covered in this thesis.

repeated several times, until a limit of retries is reached. There are different methods to implement the end-to-end ACKs, for example as cumulative ACKs or positive ACKs. Cumulative ACKs allow the destination to confirm several data packets at once (up to the last valid received), while positive ACKs confirm only individual packets. Another option is the use of Selective ACKs (SACKs), that may confirm parts out of a group of acknowledgeable packets.

The RTO used in end-to-end reliability may be a fix value, but it also can be dynamic and adjusted to the observed network parameters. A thorough evaluation of RTO settings as part of the investigations of end-to-end performance improvement is carried out in Chapters 3, 4, and 5.

Congestion Control

Congestion can be the cause of different malfunctions inside a WSN and is an important issue that can lead to a degradation of the network's performance. Congestion mainly happens in two ways:

1. If the incoming packet rate of a node is higher than the rate of outgoing packets, the buffers eventually get filled, as the internal buffers of a node are limited in their size. Upon receiving further packets, they either replace already buffered packets or they are dropped, depending on the applied policy.

2. If several nodes that are in transmission or interference range are transmitting simultaneously, radio collisions can happen, invalidating the affected packets. This causes the throughput to drop and at the same time retransmissions become necessary. This adds delay to transmissions and decreases the throughput. If the rate of data-creation surpasses the achievable throughput, nodes get congested.

Congestion control at the transport layer usually is a part of flow control that is carried out by the two endpoints of a data transmission. A mechanism relevant to the investigations carried out in this thesis that belongs into the category of flow control is congestion control (CC). CC can be generally split into 3 separate phases:

1. Congestion Detection

2. Congestion Notification

3. Congestion Avoidance

By monitoring the network traffic or internal node parameters, congestion detection may be applied. Examples for congestion detection mechanisms are the monitoring of internal buffer-states or the observation of channel usage. If congestion is detected, a congestion notification mechanism may need to inform other nodes about the state of the network. This can be achieved by disseminating control messages inside the

network. In multihop connections, intermediate nodes may also collaborate in the CC with mechanisms like the Explicit Congestion Notification (ECN) mechanism [48]. In the last step, congestion avoidance, mechanisms react if a congested network state is detected or is about to be entered.

An important protocol in Internet communications that implements end-to-end reliability and CC mechanisms is TCP. While TCP implements both mechanisms at the transport layer, there also exist separate solutions for both mechanisms implemented by other protocols at different layers of the communication protocol stack. CoAP is an example for an application layer protocols that implements both end-to-end reliability and CC. Protocol stacks that cover several layers, like ZigBee [7], also may offer end-to-end reliability and CC mechanisms.

In the following chapters it will be shown that end-to-end reliability and CC are important mechanisms that have a crucial effect on the end-to-end performance of WSNs. In Internet communications, TCP is commonly used for the reliable exchange of data between two Internet endpoints that implements both mechanisms. The methods applied by TCP to offer end-to-end reliability and CC are used as a reference for several of the evaluations carried out in this thesis and compared to the ones applied by CoAP, among others. In the following, the main features of TCP are presented and an introduction to other important transport layer protocols that implement end-to-end reliability and/or CC is done. The large amount of transport protocols available cannot be covered in this thesis, therefore only a subset is presented. An extensive survey of transport protocols for WSNs is done in [49].

2.6.1 TCP

When a reliable data transmission between two endpoints is necessary, a commonly used protocol is TCP. Many Internet applications like the World Wide Web (WWW), File Transfer Protocol (FTP), and E-Mail rely on TCP. In contrast to the User Datagram Protocol (UDP), TCP requires a session to be established between the two endpoints of the connection. After a session is initiated via a three-way handshake procedure, the reliable data transmission may begin. TCP assures that the packets arrive in order via sequence numbers and it uses end-to-end ACKs to confirm the reception of data packets.

Additionally to the end-to-end reliability, TCP uses flow control mechanisms. Flow control adjusts the transmission rate of data at the source to use the available bandwidth of the network in an efficient way. It takes care of adjusting to the reception window of the receiver, avoiding overflows, and assuring fairness among different data flows. This is achieved by adjusting the sliding window which determines how many bytes (or packets) may consecutively be transmitted by the source of the data transmission before it stops and waits for one or multiple ACKs. The maximum size of the sliding window depends on the available buffer space at the endpoints of the transmission and normally starts with a small initial value [50]. As ACKs keep confirming the

reception of data at the destination, the source knows that no packets have been lost and assumes the network supports the bandwidth consumed by the data flow. Under these circumstances it increases the window size and again waits for the data to be confirmed. As long as all packets are confirmed, the source may increase the window size.

The so called slow start mechanism is used to adjust the Congestion Window Limit (CWL) to avoid congestion. During the slow start phase, each ACK increases the window limit, until the slow start threshold is reached. Upon reaching this threshold, the window limit is only incremented after all packets from the current window are acknowledged, meaning that the size of the window increases linearly while no losses are detected. This phase is called congestion avoidance phase. When a loss is detected in any of these two phases, the window size is reduced not linearly but exponentially. A loss is assumed, as soon as an RTO happens.

The RTO that is used to determine how long to wait for ACKs is based on the RTT which is estimated by Karn's algorithm in form of a Smoothed RTT (SRTT) [51]. The RTO is calculated as follows [52]:

When the first RTT measurement R is made, the host MUST set

$$SRTT = R \tag{2.3}$$

$$RTTVAR = \frac{R}{2} \tag{2.4}$$

$$RTO = SRTT + max(G, K * RTTVAR) \text{ ,where } K = 4 \tag{2.5}$$

The granularity G of the timers used for the calculations depends on the machine this algorithm runs on. The RTT Variance (RTTVAR) indicates how much subsequent RTT measurements differ. When a subsequent RTT measurement R' is made, a host must set

$$RTTVAR = (1 - \beta) \times RTTVAR + \beta \times |SRTT - R'| \tag{2.6}$$

$$SRTT = (1 - \alpha) \times SRTT + \alpha \times R' \tag{2.7}$$

The variables α and β can be chosen by the user, their default values being $\frac{1}{8}$ and $\frac{1}{4}$. The factor G indicates the granularity of the timers used for the timeout calculation. Since TCP was developed for wired networks, TCP assumes congestion upon not receiving an ACK in the time period defined by the RTO. However, this may not always be the case for wireless networks. Since wireless links may vary over time and may suffer from sporadic losses due to fading and other physical phenomena, the cause for a packet loss may not be congestion at all [53]. Still, TCP considers the network to be congested and reduces the window size, which may lead to an underutilization of the channel.

After the basic version of TCP, a series of modifications have been developed, each improving certain aspects of TCP, but mostly focusing on the flow and CC algorithms. The results are different versions of TCP like TCP Reno and Vegas [54] and TCP Westwood [55]. A comparison of these versions is done in [56]. The basic TCP version is often referred to as 'vanilla TCP'. Since TCP implements end-to-end reliability and CC, which are two important mechanisms for end-to-end communications in WSNs. Thus, TCP is considered for some of the evaluations done in this thesis.

2.6.2 TCP Support for Sensor Nodes (TSS)

Instead of relying on end-to-end ACKs, TSS implements a per-hop reliability mechanism [57]. TSS proclaims that end-to-end ACKs may cost a lot of energy in the case of losses, which are assumed to be around 5-10% per hop. For long routes this signifies an increased probability of packet losses along the path which could lead to an increased amount of transport layer retransmissions. TSS therefore caches TCP segments at intermediate nodes and uses local retransmissions from hop to hop. A packet is cached locally at a node, until it knows that the packet was successfully forwarded by the next node via sniffing. Also the source node may retransmit the data packet, if an overall RTO of $1.5 \times$ RTT is surpassed.

TSS offers a backpressure CC mechanism for the nodes that are forwarding TCP segments and notice that follow up nodes cannot transmit all previously forwarded packets. In this case a node waits for the next node to forward all packets or for a TCP-ACK for the already forwarded packets, before it continues transmitting packets to the next hop. The Distributed TCP Caching (DTC) [58] differs mainly from TSS as it does not implement the backpressure mechanism and allows some TCP options like selective ACKs.

2.6.3 Congestion Detection and Avoidance in Sensor Networks (CODA)

By applying hop-by hop pressure and multi-source regulation, Congestion Detection and Avoidance in Sensor Networks (CODA) [59] tries to reduce congestion. The first mechanism requires the nodes to monitor the channel load and the internal buffer state. If a certain channel load threshold is exceeded and congestion is about to happen, a backpressure message is propagated to the source node. The nodes on the path that are receiving this message throttle their sending rate and also decide whether to propagate the notification message further towards the source node. The Multi-source regulation takes place at the source: if the source exceeds a certain message dissemination threshold, it marks its messages with a regulation bit. If the sink receives such a packet, it sends ACKs to tell the packet generating nodes to reduce their sending rate. As soon as the congestion is cleared (data packets are no longer marked with the regulation bit), the sink may send ACKs that tell the data sources to increase their data rate.

2.6.4 Event-to-Sink Reliable Transport (ESRT)

Event-to-Sink Reliable Transport (ESRT) is pretended for networks where most of the nodes are keeping track of events and report them to a sink node [60]. ESRT provides end-to-end reliability and CC. Packets that are sent to the sink include bit notifications, upon which the sink calculates if the dissemination rate of the packet generating nodes may cause congestion. If the rate is too high and congestion is imminent, the sink node informs the data generating nodes to reduce the frequency of packet generation. This means that intermediate and generating nodes do not play an active role in the CC mechanism, they rely completely on the sink to give them an adequate feedback.

2.7 Application Layer Protocols

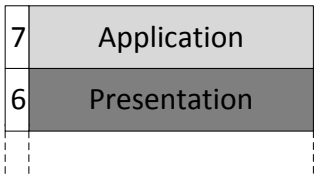


Figure 2.11: Section 2.7 focuses on the application layer of the OSI-model.

The top layer of the OSI-model is the application layer, which allows the user defined software application to access the network. Even though the name of the application layer suggests it, it does not represent the user application running on top of the communication protocol stack. Instead it acts as an interface between user programs and the network. Its task is to interpret the requests given by user application that wants to use the network and to interpret data that is reached up from lower layers, making it accessible for the on-top user application. The HTTP, Message Queue Telemetry Transport (MQTT), and CoAP protocols are introduced in the following as well-known application layer protocols used in Internet communications.

2.7.1 Hypertext Transfer Protocol (HTTP)

The Hypertext Transfer Protocol (HTTP) is designed for distributed, collaborative, hypermedia information systems [61]. This Representational State Transfer (REST) [62] protocol is used for hypertext, but can also be used for name servers and distributed object management systems. HTTP is designed to be used for the typing and negotiation of data representation in such a way that systems can be built without taking into account the type of data transmitted.

Transactions in HTTP are request/response-based and support a set of messages types to obtain or manipulate data representations between clients and servers. GET, PUT, POST, and DELETE build the base of HTTP request messages but are only a

subset of all available request message types. Commonly, TCP is used as underlying transport protocol, sometimes replaced by UDP. While the current version of HTTP (v. 1.1) establishes a connection once between client and server, in prior versions for every exchange of request/response pair a session had to be established.

2.7.2 Message Queue Telemetry Transport (MQTT)

MQTT is a publish-subscribe messaging protocol designed for Machine-to-Machine (M2M) communications. It uses message brokers to distribute messages from the publishing devices to interested subscribers. It offers three levels of Quality of Service (QoS) that determine how reliable data is delivered from publishers to subscribers. MQTT is considered to be a lightweight implementation of publish-subscribe protocol and therefore is suited for networks of constrained devices.

MQTT-SN (previously MQTT-S) [63] introduces several adaptations of the protocol to work better in constrained networks. For example, topic names are replaced by shorter topic Identifiers, certain messages become optional, or mechanisms like message buffering for data delivery to sleeping clients is added.

2.7.3 Constrained Application Protocol (CoAP)

The RESTful protocol CoAP offers a request/response interaction model between endpoints similar to HTTP, but adapted to the limitations of constrained nodes and networks and IoT specific communications [15].

CoAP is a newly designed protocol built on top of UDP and intended to form a subset of REST common with HTTP. CoAP meets many important requirements given by IoT communications, offering features such as multicast, while maintaining low overhead and simplicity for constrained devices.

CoAP defines four types of messages: Confirmable (CON), Non-Confirmable (NON), ACK and Reset (RST). The way these messages interact, are similar to the interactions of such message types known from HTTP. Above these interaction patterns, an observe mechanism is defined [64] that allows CoAP clients to subscribe to resources provided by CoAP servers.

CoAP implements reliability at the application layer, since the underlying UDP transport protocol does not provide reliability. To achieve this, messages may be marked as confirmable, signaling to the receiver of such a message, that a confirmation with the same message-ID is expected. CoAP uses an exponential backoff behavior in the case that an ACK is not received during the RTO period to avoid congestion. Requests may be carried in CON and NON messages, while responses may also be carried in CON and NON messages, as well as piggybacked in ACKs. The support of Datagram Transport Layer Security (DTLS) [65] is mandatory for CoAP.

CoAP implements a very basic CC mechanism for reliable communications. At the time of writing this thesis, very limited effort has been put into optimizing the CC for CoAP. Considering the high impact CoAP has in IoT communications, there is a

strong necessity of evaluating whether CoAP's CC mechanism is capable of dealing with congestion in IoT networks, with large amount of devices and specific data patterns. Moreover, as stated in CoAP's base specification, advanced CC mechanisms need to be defined and evaluated. This crucial area of investigation is approached in Chapters 5 and 6 of this thesis.

2.8 Multiple Layer Specifications

Some manufacturers use fixed sets of protocols for the communication protocol stack used by their devices in WSNs. These stacks normally include a proprietary application layer, like in the stacks defined by ZigBee [12] or Z-Wave [19]. In other cases, they only include a predefined application programming interface (API) to access the communication functionality provided by the stack. The rest of the stack, including NWK, MAC, and PHY layers of such multi-layer specifications are adapted to the specific needs of the network environment in which they are used.

In the ambit of IEEE 802.15.4 WSNs, ZigBee plays a very important role. In order to provide a detailed understanding of ZigBee, in the following the main features of the ZigBee protocol stack layers are introduced, along with alternative stacks (see Fig. 2.12).

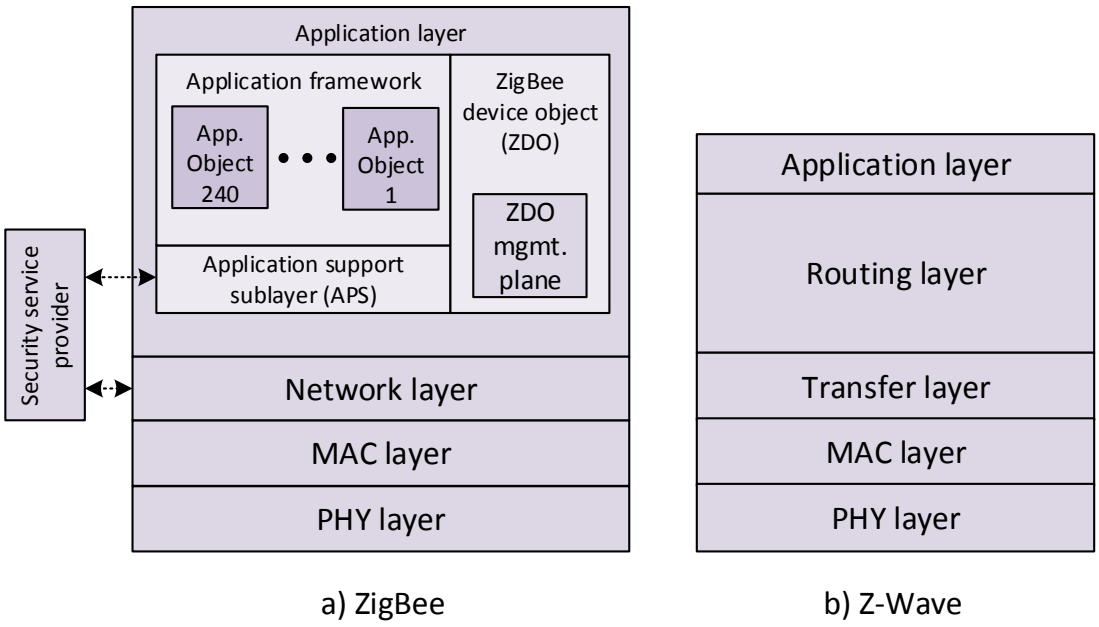


Figure 2.12: ZigBee and Z-Wave stacks in comparison.

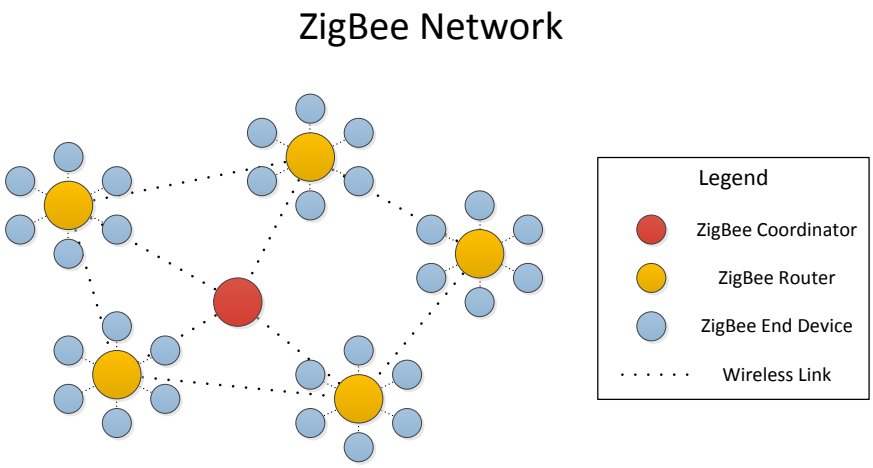


Figure 2.13: An example for a ZigBee network with mesh and star topologies. The Mesh topology is built by the ZC and the ZRs, while the Star topologies are built by the ZEDs connected to the ZRs.

2.8.1 ZigBee

ZigBee is a complete communication protocol stack for WSNs. ZigBee has been designed for a large variety of applications, such as SE, HA, or Health Care. For many of these applications a specific application profile is provided, defining which parameter settings and features from the ZigBee protocol stack need to be implemented and how they should be configured. The ZigBee stack consists of four layers: the PHY, the MAC, the NWK layers, and the application layer (called APL in ZigBee). The MAC and PHY layers are based on the common IEEE 802.15.4 (Section 2.3). The ZigBee specification defines standard mechanisms and configuration parameters, so that devices from different manufacturers can be compatible with each other, as long as they are using the ZigBee standard, operating with the same configurations and using the same application profiles.

In each ZigBee PAN, a unique ZigBee Coordinator (ZC) is needed to establish the network and to manage it. It serves as the root of the network and other ZigBee devices need to contact it, to join or leave the network. Together with ZigBee Routers (ZRs), which are able to fulfill regular functions from the application point of view, and also serve as forwarders for data packets, the ZC builds a mesh or a tree network. ZigBee End Devices (ZEDs) connect directly to ZRs or the ZC, forming local star topologies. An example for a ZigBee network topology is presented in Fig. 2.13.

ZEDs are not able to route packets. ZCs and ZRs correspond to FFDs of the IEEE 802.15.4 specification, while ZEDs correspond to RFDs. An important aspect of ZigBee networks is the fact that neither the ZC, nor ZRs may switch into sleep mode, as they need to be able to receive messages from other motes at any time. On the other hand,

it is elementary that ZEDs have short duty cycles to reduce their power consumption, as they are supposed to be powered by batteries.

A ZigBee network may operate in the beacon-enabled or beacon-less mode of IEEE 802.15.4, depending on the necessities of the application and the network topology. By default, MAC layer ACKs are enabled. According to the IEEE 802.15.4 specification, possible values for the *macMaxFrameRetries* parameter are defined to be between 0 and 7, with a default value of three.

The ZigBee NWK layer is responsible for, among other functions, discovering and maintaining routes between devices and relaying messages. It specifies a hierarchical routing scheme for the hierarchical (tree) topology and a peer-to-peer routing protocol based on AODV [36] (see Section 2.5.3 for details) for the mesh topology. While the hierarchical topology is mainly designed for many-to-one traffic, the mesh topology facilitates the communication between any two nodes of the network, *i.e.*, it supports point-to-point traffic. The route selection is based on the path cost calculation metric used by ZigBee, which computes a link cost $C\{l\}$ for every link on a route and calculates the cost of a path as the summation of the costs of all of the links along the path. When multiple RREQ messages reach the destination, the route corresponding to the one with the minimum total path cost is selected. $C\{l\}$ is defined to range between one and seven and is calculated as:

$$C\{l\} = \begin{cases} min\left(7, \text{round}\left(\frac{1}{p_l^4}\right)\right) & \text{(default)}, \quad (2.8) \\ 7 & \text{(optional)}, \quad (2.9) \end{cases}$$

where p_l is the one hop LDR [7]. The LDR of a link can be estimated from the LQI calculated by the receiving node on the reception of a packet. The LQI may be obtained from measurements of the signal-to-noise ratio (SNR), the RSSI or a combination of both. The conversion of LQI to LDR is not specified in the specification and is up to the implementation. Setting the link cost to a fixed value, as in Equation (2.9), is also an option provided by the ZigBee specification. This option is equivalent to applying the HC metric that chooses the shortest available route.

The APL is the top layer of the ZigBee protocol stack, which provides an optional end-to-end reliability mechanism to assure the successful delivery of data between two end nodes. End-to-end ACKs can be used to confirm the reception of the data at the destination node. The loss of a packet is assumed if after a certain amount of time, no application layer acknowledgment (APL ACK) has been received from the destination node. Then, a retransmission of the data packet is initiated. The duration of the time interval during which a response is expected is specified by the RTO value. ZigBee uses blockwise acknowledgments, which are used to confirm the successful reception of several packets at once, similar to the cumulative acknowledgment method of TCP [66]. If the block size is set to one, the end-to-end reliability follows the simple stop-and-wait policy with positive ACKs.

If the data to be transmitted to a destination node exceeds a single data frame's capacity, the fragmentation option allows to split the data into several frames. The fragments are called 'Blocks' and up to *apsMaxWindowSize* of these blocks may be transmitted consecutively. ZigBee has no congestion window adaptation policy, since the window size is always a fixed amount of blocks. The receiver answers with an ACK that contains a bit field, indicating if any packets of a block are missing. Then the source node may retransmit missing packets from the window, until all of them have been acknowledged. After all packets have been acknowledged, the transmission window is moved, so the following blocks may be transmitted. To make fragmented transmissions possible, it is required that the maximum window size is the same for all nodes in the network.

2.8.2 Z-Wave

Z-Wave offers a protocol for reliable wireless communication, with the main purpose to communicate short control messages from a control unit to one or more nodes in the WSN using end-to-end reliability. This section only focuses on the basic aspects of Z-Wave.

Z-Wave defines 2 basic kinds of devices, which are controlling devices and slaves. Controlling nodes are the ones that initiate commands and distribute the commands to different destinations, while slave nodes reply to and execute these commands. Both, controlling devices and slave devices, may act as relays when it comes to forwarding messages.

Z-Wave is composed of 5 layers:

- PHY: Z-Wave nodes operate at 868 MHz (EUR) and 908 MHz (US) bands, as well as in the 2.4 GHz band, allowing bitrates of 9.6 kbit/s, 40 kbit/s and 100 kbit/s, respectively.
- MAC: Z-Wave uses a CSMA/CA mechanism to access the medium.
- Transfer: This layer controls the transfer of data between two nodes. It includes retransmissions, checksum checks and ACKs.
- Routing: In Z-Wave, source routing is used. This requires the nodes to know the topology of the network. The routing information is included in each packet, so each node along the path acquires the next hop information from the packet. The maximum HC is 4, which is supposed to be enough for HA scenarios.
- Application: The topmost layer of the Z-Wave protocol stack is responsible for decoding and executing commands of the Z-Wave network. It also specifies what kind of messages needs to be interchanged between two nodes of the WSN. End-to-end reliability is provided at this layer too.

Z-Wave is designed to be used in HA environments, where a plentitude of devices needs to be controlled by using short control messages.

2.9 Research Methodology and Tools Used for Performance Evaluation

To conduct the research and carry out evaluations in this thesis, two common methods of performance analysis are applied: experiments in real WSNs and network simulations. In experiments, the nodes need to be programmed with operating systems that run the communication protocol stacks. In the ambit of constrained devices two important software stacks are used: The Tiny Operating System (TinyOS) [67] and Contiki [68]. Two testbeds are used for experimental performance evaluations, namely the UPC testbed in Castelldefels and the ETH FlockLab testbed in Zürich. These are introduced in the Sections where the correspondent experiments are carried out. To program the nodes for experiments in the UPC testbed, TinyOS is used, while in the FlockLab testbed Contiki is used to program the devices. Contiki is also used for the simulation-based evaluations in this thesis, since it can not only be used to program real nodes, but it also can be used to simulates devices in the Cooja network simulator [69], which is part of the Contiki toolset.

2.9.1 Advantages and Drawbacks of Experimental and Simulation Environments

Experiments require a real physical setup, such as a testbed of sensor nodes. The main advantage of experiments is that a real physical medium is used for radio transmissions, therefore being more realistic than simulations, which often suffer from limited radio channel models. Further, in testbed experiments real data processing takes place: messages travel along the communication protocol stack and require processing at each layer introducing delays that are often not taken into account in simulations. Overall, experiments produce most realistic results.

However, there are several drawbacks to the experimental approach: For instance, it is difficult and costly to set up an appropriate testbed. Also, the gathering and understanding of measurements is more complicated than in a completely controlled environment (for example simulations). Unpredictable physical phenomena (signal scattering, multipath propagation, external/internal interference, etc.) may require a very detailed logging and analysis of events that occur during an experiment. Above that, real physical restrictions, such as memory capacity and processing power apply, requiring efficient engineering of the mechanisms and algorithms that are to be tested. Eventually, testbed experiments generally require much longer than simulations, since communications happen in real time and cannot be parallelized in multiple instances.

As alternative to experiments in real testbeds, network simulators can be used for performance evaluation. There exist several free/open-source network simulators that allow the simulation of WSNs, including the NS series [70], Omnet [71] and Cooja [69]. For a typical network simulator, a variety of protocols for different layers of

the stack are implemented and often several types of radios are available to chose from. When setting up a simulation, practically any network topology is possible, including scenarios with mobility. Providing tools to test many different topologies and parameters quickly by changing the simulation configurations is a strong point of network simulators. Moreover, the parallel execution of simulations allows to gather results quickly and incorporated logging tools help to evaluate simulation results. In contrast to real testbed experiments, simulations can be paused or their execution speed may be altered, facilitating the analysis of the network's behavior.

However, as a major drawback, simulated radio channels suffer often from inaccuracy and that therefore are not capable of reproducing the characteristics of complex in- and outdoor environments, such as buildings or cities, respectively.

2.9.2 TinyOS

TinyOS [67] is an operating system designed for low-power wireless sensor nodes. It provides drivers for several microcontroller and radios, including the MSP430 MCU [72] and CC2420 radio transceiver [21]. Supported sensor nodes are the TelosB, MicaZ, and IRIS nodes, amongst others.

TinyOS is programmed in NesC [73] and is divided into modules that provide different aspects of the protocol stack functionality. Modules can be wired (linked) with each other for data handovers and event signaling via interfaces. TinyOS executes its code synchronously in a non-preemptive (blocking) way, but also allows to spawn tasks that are executed as soon as the scheduler is able to assign a free time slot.

The default TinyOS library includes a variety of modules providing the basic functionality for end-to-end connectivity between nodes. In this thesis, TinyOS is used to program the nodes of the sensor grid of the UPC in Castelldefels (introduced in the next chapter), where a significant part of the evaluations are carried out.

2.9.3 Contiki

Contiki is an open source operating system designed for constrained devices and the IoT and is the choice of many sensor motes. Since Contiki is an open source project, it is constantly evolving and changing its structure. Contiki benefits from an increasing number of supported motes and the development of currently available and evolving protocol. Fig. 2.14 shows the IETF protocols over IEEE 802.15.4 and how they are implemented in the full IPv6-capable Contiki, along with other Contiki specific layers.

2.9.4 Cooja

Cooja is a network simulator that can execute Contiki-based code, designed to be a rapid development platform for Contiki. Cooja supports simulation of sensor networks at three levels: the application level, the operating system, and the machine code instruction set [69]. As an outstanding feature, the binary image of compiled Contiki

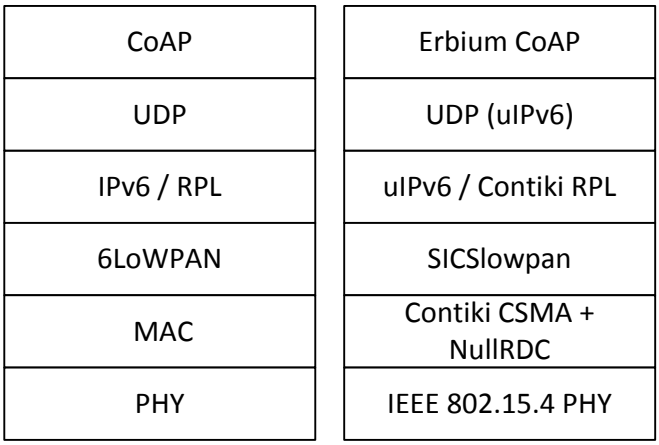


Figure 2.14: A comparison between the IETF communication protocol stack (left) and its implementation in Contiki (right).

code designed for real motes can be uploaded to be used for simulations. Cooja also is capable of emulating the real hardware for several types of nodes. All together a high degree of simulation realism is achieved, since memory restrictions, the real node periphery and internal processing of the motes are simulated as well.

Cooja allows three-dimensional deploying of nodes inside a simulated environment and offers several radio models. The radio models determine how radio signals propagate in the network. The common radio channel model used is the unit disk graph radio model (UDGM), which defines the LDR within the transmission range to 100%. The interference range, set normally to be twice the transmission range, indicates up to which distance the radio signal is capable of interfering the radio of other nodes. If a node receives two packets at the same time (being in transmission or interference range), the simulator assumes a packet collision, rendering both packets useless for the receiving node. More complex models like a ray-tracing-based model are under development or not yet fully implemented. Cooja is used in the evaluations presented in Chapter 5 of this thesis.

3

Performance Evaluation and Improvement of an IEEE 802.15.4-based WSN Protocol Stack

The task of improving performance for end-to-end transmissions in low-power wireless networks is not a trivial one. In fact, the performance of a low-power wireless network depends on a large variety of physical, environmental, and architectural design factors. It is determined by the communication protocol stack configuration, the network topology and node deployment, the environment in which the network is deployed, and the type of hardware and telecommunication technologies that are used. Each of these aspects can be evaluated with regard to its impact on the network performance. Improvements then can be achieved via the tuning or alteration of the configurations.

This chapter presents an in-depth analysis of end-to-end wireless low-power communications for an IEEE 802.15.4-based communication protocol stack. In this analysis, the default mechanisms and parameter settings of different protocol layers are compared to alternative settings with regards to their impact on the end-to-end performance according to a set of performance metrics. The results reveal settings that are crucial to the performance and settings that only have a minor impact on the performance. It can be shown that replacing the default settings provided by alternative settings can lead to noticeable improvements or deterioration of end-to-end communications. Further, synergetic effects between different protocols layers that can have important effects on the end-to-end performance are exposed. The content of this chapter has been published in [BGDP12].

3.1 Introduction

The number of motes in networks of low-power wireless devices, the topologies they form, and the hardware configurations vary from one deployment to another. This makes each network unique, presenting its own specific characteristics. Yet, many networks rely on the same standards and protocols to achieve interoperability. In addition, commercial solutions for low-power wireless networks, like ZigBee, provide the exactly same communication protocol stack for a wide spectrum of hardware and use cases.

Hence, independently of the setup or the function of different networks, the communication protocol stack is common denominator and the focus of the research carried out in this thesis lies on determining methods to improve its end-to-end network performance. This assures that the results of the research can be applied in general, without being restricted to certain topologies or hardware configurations.

To provide an introduction to the performance analysis of WSNs and as a starting point for the research on performance improvements for low-power wireless network, an IEEE 802.15.4 network scenario with a basic, yet representative application is chosen. In this basic scenario pairs of devices establish end-to-end connections and apply transparent communication patterns, in order to facilitate the analysis of the network performance. In Section 3.2, a IEEE 802.15.4-based communication protocol stack with a set of protocols is introduced. The investigated parameters and mechanisms of the protocol stack that are analyzed in this chapter are detailed in Section 3.3, using the performance metrics presented in Section 3.4. The test bed used for evaluations is presented in Section 3.5 and the test scenarios used to carry out the evaluations are specified in Section 3.6. The results of the evaluations are presented in Section 3.7. Section 3.8 concludes this chapter, giving a resume of the results obtained.

3.2 Protocol Stack Configuration

The large variety of available protocols at each layer results in a vast and inconceivable number of possible combinations to set up a communication protocol stack. The evaluations in this chapter only focus on a subset of the available protocols to form the analyzed protocol stack. To assure the relevance of the investigations and the contribution to the scientific community of this work, representative protocols are chosen to be included in the stack. These protocols can be used for a variety of use cases and are suitable for different types of networks.

The stack depicted in Fig. 3.1 has been designed and implemented for TinyOS and is evaluated in the Castelldefels testbed, composed of 60 TelosB motes [74]. The CC2420 radio transceivers of the TelosB motes use an IEEE 802.15.4 compliant radio. TinyOS provides a default MAC layer for TelosB nodes, to which several modifications are investigated in the experiments. The MAC layer implements CSMA/CA for channel

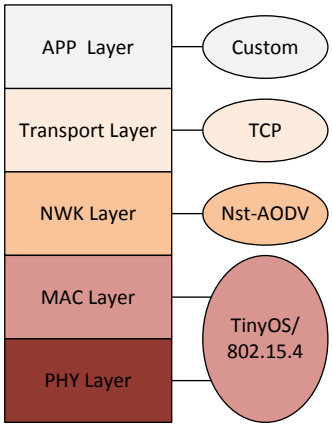


Figure 3.1: The IEEE 802.15.4-based communication protocol stack used for the evaluations.

access in the beacon-less mode of operation. At the network layer, the lightweight implementation of AODV, nst-AODV [3], is used as the routing protocol. Protocols based on AODV are used in the area of constrained networks, e.g., in ZigBee [75] and the one-to-one mechanism developed for RPL [47], to establish arbitrary one-to-one connections. nst-AODV's working principles and advantages over the standard AODV implementation are presented in [3]. To reduce complexity, local route repair is disabled.

Above the network layer, a basic version of TCP based on RFC 793 with the slow start algorithm is used for a reliable transmission of data and flow control. All core mechanisms of TCP are implemented, with certain minor variations, adapting them to the specific test conditions. Accordingly, the granularity of 1 ms is used, since TinyOS provides millisecond timers. Minimum and initial RTO values are set to 500 ms and 1 s, respectively. The default values defined by the investigated protocols and the operating system are shown in Table 3.1 for the network design criteria evaluated.

The user application resides at the top, responsible for initiating transmissions and processing received data. This IEEE 802.15.4-based stack provides all the functionality that may be necessary for a typical WSN use case, including reliability at different layers, on-demand routing, and congestion control.

3.3 Investigated Protocol Stack Parameters

The default configurations of the previously introduced protocol stack layers may not deliver the best performance under specific network conditions. Combining and evaluating all possible combinations of parameters and mechanisms inside and across the layers is difficult due to the large amount of combinations. Therefore, the default settings and certain alternative mechanisms of this IEEE 802.15.4-based stack are investigated in

Table 3.1: The default settings of the investigated design criteria.

Layer	Criterion	Default Setting
MAC	Backoff method	TinyOS backoff
	Max retries	3
Routing	Routing metric	Min. HC
Transport	CWL	8
	RTT calculation	Karn method
	ACK method	Cumulative ACK

this chapter. Further, during evaluations, configurations of individual design criteria are singled out and the settings of other criteria are fixed.

At the MAC layer, the impact of changing the maximum number of retries and altering the backoff behavior is evaluated. By default, the operating system of the nodes used in the evaluation, i.e., TinyOS, employs a backoff mechanism different from the one used in IEEE 802.15.4.. Moreover, the maximum number of retries is a crucial parameter that can affect the overall network performance. At the NWK layer, two different routing metrics are evaluated: the classic HC metric, which is the default metric of the AODV and a LQI-based metric, called Path-DR [35]. At the transport layer, the effects of different approaches for RTO calculation algorithms and end-to-end ACK-mechanisms are analyzed. Also, it is demonstrated that for constrained networks such as the one used in this work, a Congestion Window Limit (CWL) must be used and tuned to an appropriate value. The list of the investigated network parameters and their default configurations are given in Table 3.1.

Prior to evaluating the impact on the performance of aforementioned mechanisms and protocol settings, it is necessary to determine a set of performance metrics. The main metrics that are used for the evaluations carried out in this chapter are presented in the following section.

3.4 Evaluated Performance Metrics

A network's performance may be measured by different metrics. Each metric reflects a specific aspect of network's performance, such as the degree of reliability that is provided or how fast data can be delivered from one device to another. Not always one metric is enough to describe the performance of a WSN sufficiently. Often it is necessary to apply several performance metrics to obtain a comprehensive impression of how a network performs. In the following, the metrics used to determine the performance in the evaluations of this chapter are presented and their significance for WSNs is evaluated.

3.4.1 Packet Delivery Ratio (PDR)

The amount of successfully transmitted packets over the total amount of transmitted packets is indicated by the PDR. Like the throughput presented later in this section, the PDR may be measured at different layers; however, the term used at each layer may change. For example the PDR at the link layer may be called LDR. The PDR is an indicator for reliability, giving a ratio of how many packets are expected to be lost during communications. Commonly, the PDR is used to describe the delivery ratio of the application layer.

The PDR is one of the most important metrics considered for the evaluations in this thesis. In networks, where events are created by the sensor nodes, reliability is an important metric, especially if the events are of a critical type (like an alarm) and have to reach their destination with a certain probability. In a more general view, for any other type of application a high PDR is desirable, indicating that the network is working correctly and packet losses are unlikely. A high PDR is obtained if the different layers of the communication protocol stack work together coherently.

3.4.2 Link Delivery Ratio (LDR)

When the number of successfully transmitted packets over the total number of sent packet is only evaluated for a direct connection between two devices over a single link, it is referred to as LDR. If MAC layer reliability mechanisms are applied, the LDR normally is measured at the NWK layer using the feedback from the underlying layers about if the packet was successfully transmitted. This method of determining the LDR may not be a 100% accurate, since MAC layer ACK packets indicating the successful reception of the packet at the destination may get lost. If an accurate LDR is to be determined, the packets sent and received need to be counted at the source and destination nodes, respectively, and compared afterwards.

Wireless links in IEEE 802.15.4 networks are likely to be asymmetric [76], therefore the LDR for a link between two nodes can be asymmetric too. This phenomenon has a great impact on the routing mechanism, which is responsible for determining routes between two nodes. The LDR has a direct impact on routing protocols like RPL or AODV, that can use metrics like the ETX metric to determine the optimal routing path.

3.4.3 Delay / Round-Trip Time (RTT)

For some applications, the time it takes for a packet to reach its destination may be an important metric. This interval of time is often referred to as delay. When measured at the application layer, it indicates how long it takes for data to be received by the destination, after being sent by the source node and travelling over one or several hops. A per hop delay may also be considered, when measured at lower layers, for example the MAC layer.

If the successful transmission of data from the source to the destination triggers an ACK or another form of a response message, the packet returning to the source can be used to calculate the RTT. The RTT can be very important for any type of protocol or mechanism that uses RTT information to adapt its behavior during transmissions. For example, the use of RTT information in an end-to-end reliability mechanism can help to adjust the retransmission timers that determine when a packet is considered to be lost and requires a retransmission.

The delay and RTT are prone to fluctuate strongly in WSNs as the network conditions may change between different transmissions. The time packets spend on their way between source and destination node depends on various factors:

- The travel duration is affected by the length of the route, as each hop adds to the total delay. The one hop delay depends mainly on the MAC layer performance. In the research ambit of this thesis it is determined mainly by the channel access mechanisms used by the CSMA/CA algorithm and the amount of allowed retries in IEEE 802.15.4. How these mechanisms perform depends on the amount of traffic (and therefore the degree of congestion), as well as the LDR of the wireless links over which data is transmitted. In networks with high congestion and/or links with a low LDR, the amount of collisions and packet losses increases, resulting in a higher number of MAC layer retries and eventually in higher one-hop delays.

- At the network layer the delay is affected by the routing mechanisms. If processes like route discovery or route repair are necessary when a packet is intended to be forwarded and no next hop is available, the delay will increase by the amount of time it takes for such a process to finish. Also, if the network layer uses buffers to store several messages to be forwarded to other nodes, high traffic may cause the packets to stay longer in queue, which affects the delay.

- If in any case a packet is dropped or lost at any layer and reliability mechanisms are used at higher layers (application layer or TL), the delay will increase for the time it takes for the RTO to fire before the packet is retransmitted. Since RTOs at the application or transport layer normally can be quite large (several hundreds of milliseconds up to dozens of seconds), when compared to one-hop delays (normally under 100 milliseconds), lost packets may be an important cause for high delays.

The delay is relevant for time critical applications and therefore an essential metric for the analysis of end-to-end performance. The RTT per se also is an important metric that indicates response times for end-to-end transmissions, but moreover, it is relevant to the performance of some protocols that rely on RTT information to determine their behavior.

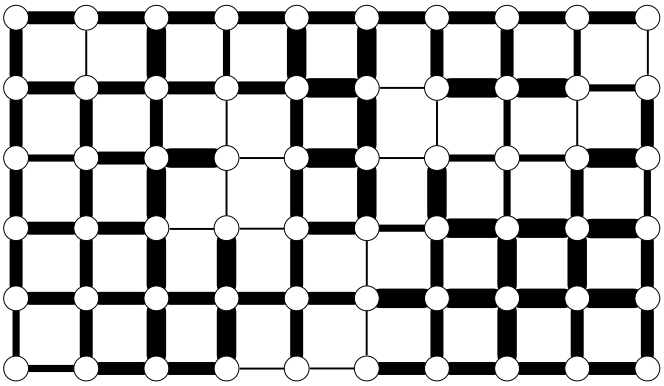


Figure 3.2: A snapshot of the one-hop connectivity among neighboring grid points in the testbed. Line thickness is proportional to the LQI value of the link.

3.4.4 Throughput/Goodput

The throughput indicates how much data is transmitted in average per second, often measured in bit/s or byte/s. The throughput may be calculated over the whole network from the simultaneous ongoing transmissions or just be observed for a link or a concrete data flow between two nodes. The throughput may be measured at different layers of the stack. If only the useful amount of data transmitted is counted, the term goodput may be used instead of throughput. The throughput or goodput mainly depends on the delay and the PDR between two devices.

When a significant amount of data is transmitted, the goodput of the application may be a good overall performance indicator. Moreover, the throughput or goodput is a general indicator of how a network performs. A high goodput indicates that all the layers of the communication protocol stack work well together and the overall performance of a network is good, since it strongly depends on the delay and the PDR. If a low goodput is observed during evaluations, there may be issues at one or more layers that affect the overall performance.

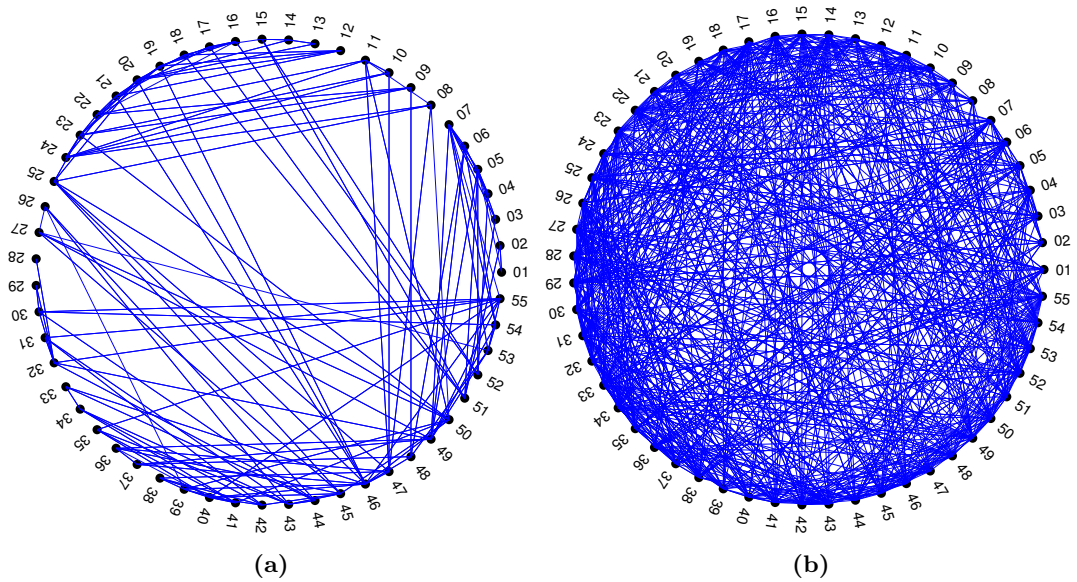
3.5 Testbed Setup: Castelldefels Sensors Grid

The UPC sensor grid in Castelldefels features an indoor grid composed of 60 TelosB nodes [74]. The nodes are located in an office environment and are attached on the bottom of a rectangular wooden grid structure that hangs 0.5 meters below the ceiling. The grid has a length of 8.1 meters and a width of 5.4 meters. The nodes in the grid are separated equidistantly from each other, building a mesh with 6 rows and 10 columns (6x10). The placement of the nodes in the grid is shown in Fig. 3.2, including a snapshot of LQI information measured during one test run.

On the upper part of the wooden structure, a USB cable-tree connects all nodes to a central computer. The nodes are powered by this connection, at the same time

allowing it to program them and to communicate with them over the serial interface. This allows for debugging programs and logging of events at the central PC.

The TelosB nodes are equipped with a TI MSP430 microcontroller [72], 48 kByte Flash, 10 kByte RAM and a TI CC2420 RF transceiver [21]. The radio transceiver operates in the 2.4 GHz frequency band with a maximum data transmission rate of 250 kbit/s. The CC2420 radio allows adjusting the transmission power in 30 steps, the lowest value corresponding to an output power of less than -25 dBm and the highest to 0 dBm. Using the lowest transmission power setting of 0 (i.e., < -25 dBm) results in barely any node being able to communicate with other nodes of the grid, as the transmission range in the majority of cases stays below the minimum distance of 90 cm between any two nodes. In rare cases communication between two nodes is possible at the lowest power setting, based on slight production differences of the hardware. Setting the transmission power to 1 (i.e., -33 dBm), results in a large variety of possible communication ranges for the nodes, as shown in Fig. 3.3a. Increasing the transmission power to any level higher than 1 increases the communication ranges drastically, allowing most of the nodes in the network to communicate with any other node over a single hop (Fig. 3.3b).



(a) **(b)**

Figure 3.3: A snapshot of the connectivity of 55 nodes in the Castelldefels grid at (**a**) a transmission power setting of 1 and (**b**) a transmission power setting of 2.

Highly complex signal propagation is experienced in the cluttered office environment of the testbed, as it is typical for indoor networks. Adding to the complexity of the network conditions, during daytime, active IEEE 802.11 networks have been detected

in the same environment. On the contrary little to none traffic has been observed during nighttime. To avoid strong interference from IEEE 802.11 networks, the nodes use channel 26 to communicate with each other, since none of the IEEE 802.11 devices operate in the corresponding frequency range. To assure similar conditions for all experiments conducted in this thesis, they are carried out during nighttime, where the general activity inside the building is the lowest.

The real radiation patterns of the omnidirectional antennas vary from node to node, causing significant differences in transmission and interference ranges. Because of that and due to the spatio-temporal effects of the environment, even if all nodes are distributed equidistantly in a symmetric pattern, the links and link qualities between nodes vary significantly. The wireless links of the testbed can be characterized by the κ- and β-factors, which indicate the inter-link reception correlation and the link burstiness, respectively [77, 78]. The κ-factor indicates to what degree the receptions of a packet on different links are correlated. The β-factor indicates how intense a link of intermediate quality (10%–90% LDR) switches between poor and good delivery. The complementary cumulative density function (ccdf) of the κ-factor of the testbed is depicted in Fig. 3.4a. Nearly 50% of the links pairs in the network have a $\kappa > 0.8$. This indicates that packet deliveries on the links of the network are mostly independent. This characteristic is similar to the one of other testbeds analyzed in [77]. On the other hand, a low β value on the majority of the links is measured, as can be seen in Fig. 3.4b. The low degree of burstiness can be explained by the fact that measurements are carried out during nighttime, where the interference level or other causes of burstiness are low.

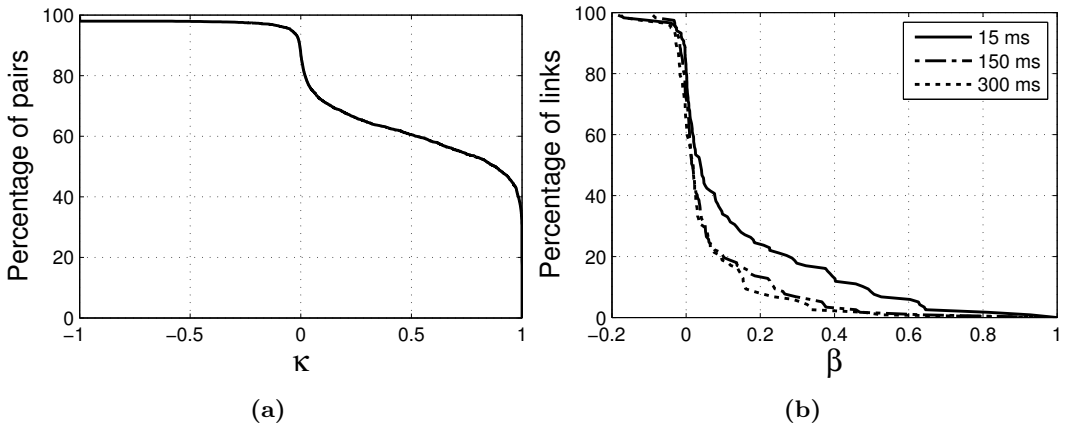


Figure 3.4: (**a**) ccdf of κ for link pairs in the UPC Castelldefels testbed for a transmission power of about –25 dBm. (**b**) The corresponding ccdf of β for different inter-packet arrival times.

3.6 Definition of Test Scenarios

In the following, the test scenarios used to evaluate the mechanisms and parameter settings listed in Section 3.3 are presented. To evaluate them, a data transfer application is defined, which consists in transmitting a fixed amount of 44 kByte from a single data source to a destination node. This amount of data may represent different types of content, including logged events, audiovisual files or a binary image. The transfer of this data is considered successful, if all data frames are delivered successfully at the destination node. During the transmission process, detailed information such as the packet RTT measured by the transport layer and the amount of failed MAC layer transmissions is logged.

Two different route setups are defined for the evaluations. In the first setup, static routes are used to make evaluations, where these static routes are built by running the nst-AODV a priori and using these routes throughout the test. In total, 6 pairs of endpoints are chosen for the evaluations with static routes. By maintaining static routes, the influence of the NWK layer on the experiments is null, thus allowing a better evaluation of the MAC layer parameters. Experiments with static routes are repeated 10 times, corresponding to the transmission of approximately 4700 frames. In the second route setup, nst-AODV is fully enabled, allowing route changes during the data transfer and the participation of all nodes of the grid in the transmission. To extract an average behavior, 31 random pairs are chosen as endpoints of the transfers. Although the dynamic route setup is more adaptive and realistic, the static route evaluations are also crucial. The reason is that they enable a more comparable setup, since the dynamic routes might change the route lengths and the values of other metrics significantly. The experiments with dynamic routing are repeated 5 times, corresponding to the transmission of approximately 2350 frames for each pair of nodes. In both setups, all experiments are repeated with different CWLs for the Transport layer, since observations show that this parameter has an essential influence on the outcome of experiments. The maximum CWL is set to be 8 and 5 for static and dynamic routing scenarios, respectively. For all tests, the nodes' buffers are chosen big enough to avoid dropped packets due to buffer overflows. The investigated performance metrics are the goodput, calculated as the useful amount of data over time received by the destination node, the RTT between the two endpoints of the transmission and the percentage of successful transfers from all the transfers of a test run. Only successful transfers are used to evaluate measured values as goodput and RTT.

The nodes are programmed with TinyOS [67] version 2.1.1. For all tests, the transmission power of the nodes is set to a low value (TXPOWER = 1), which increases the probability for routing protocols to choose routes with multiple hops to connect two devices of the network.

Figure 3.5: A graphical representation of the IEEE 802.15.4 backoff intervals.

3.7 Evaluation Results

The first part of the evaluation focuses on modifications at the MAC layer. Subsequently, the findings for the routing mechanisms at the NWK layer are presented, and ultimately design criteria for the transport layer are analyzed.

3.7.1 MAC Layer

The MAC layer is responsible for carrying out the channel access procedures and also provides one hop reliability, if enabled. The results presented in the following show that the channel access backoff behavior, the use of software or hardware ACKs, and the number of allowed retries have noticeable effects on the end-to-end performance of the analyzed scenarios.

3.7.1.1 MAC Layer Backoff

An important design criterion at the MAC layer is the type of backoff mechanism used before a transmission. TinyOS uses its own backoff mechanism as described in [79], whereas it does not employ the Binary Exponential Backoff (BEB) mechanism used in IEEE 802.15.4 (see Section 2.3.2 for details).

The one hop delay added by the CSMA/CA mechanism for a single transmission of a packet can vary between 0 BUs and in the 'worst' case 116 BUs, if CCA is repeated 4 times with the maximum backoff. 116 BUs correspond to 37.12 ms. However, this is just a theoretical value, as the channel would need to be busy over a long time and then idle after the last backoff. Also the random duration of the backoff will be lower in average, as the probability for the durations in an interval are uniformly distributed. The probability for the CCA mechanism to find the channel busy increases, as the amount of nodes and the transmission rate increase around the node that wants to transmit.

The size of the interval from which the CCA choses its random backoff value also determines how probable it is, that nodes being in each others transmission range and wanting to transmit sense the channel idle simultaneously and start a transmission in parallel. Parallel transmissions could cause all nodes that transmitted simultaneously to have to do a transmission retry, as their packets may have collided at the destination(s) of the packets. IEEE 802.15.4 increases the range of the intervals to reduce the probabilities of collisions in subsequent tries.

Figure 3.6: A graphical representation of the TinyOS backoff intervals.

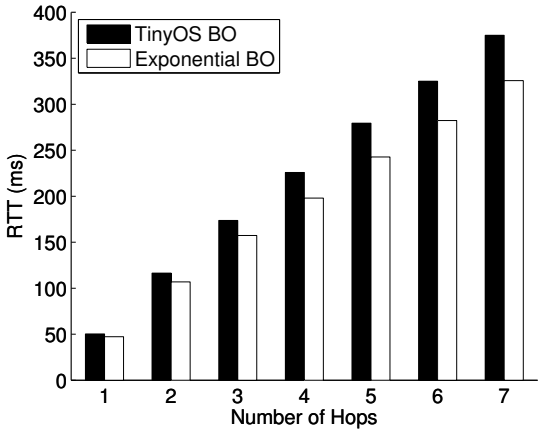
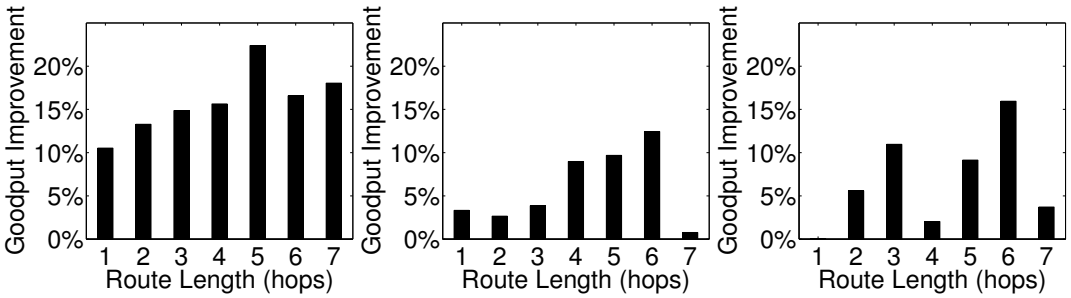


Figure 3.7: RTTs measured for the different backoff mechanisms depending on the number of hops over a static route.

The TinyOS backoff on the other hand does not apply an exponential backoff. It starts with a large backoff interval and then subsequently uses short intervals of the same length. The initial interval reaches from 1 to 32 BUs, the follow ups from 1 to 8 BUs.

The different backoff durations applied by the IEEE 802.15.4 and TinyOS backoff mechanisms have a direct impact on the RTT values that are measured for routes of different lengths. The RTT values for static routes from 1 to 7 hops are shown in Fig. 3.7. When taking a look at the performance for one hop it can be seen that the TinyOS backoff results in slightly larger RTT. The difference grows with each hop and reaches a difference of around 50 ms for larger routes. The increasing difference between the absolute RTT is a result of applying the backoff mechanisms at each hop. The slightly large one hop delay of TinyOS is added up to 14 times in the case of a 7 hop route (forward path + backward path).

From the point of view of fairness, the TinyOS backoff tries to be fairer in the sense that nodes that did not get access to the channel in their first CCA try have an increased probability to access the channel in the second try. This is due to the fact that the follow up intervals for the random backoff are shorter, meaning that the nodes are more aggressive when trying to get access to the channel repeatedly. However, if many nodes follow this procedure, the probabilities for collisions may increase, as many nodes are trying to access the channel in short intervals of time.



(a) CWL=1 **(b)** CWL=4 **(c)** CWL=8

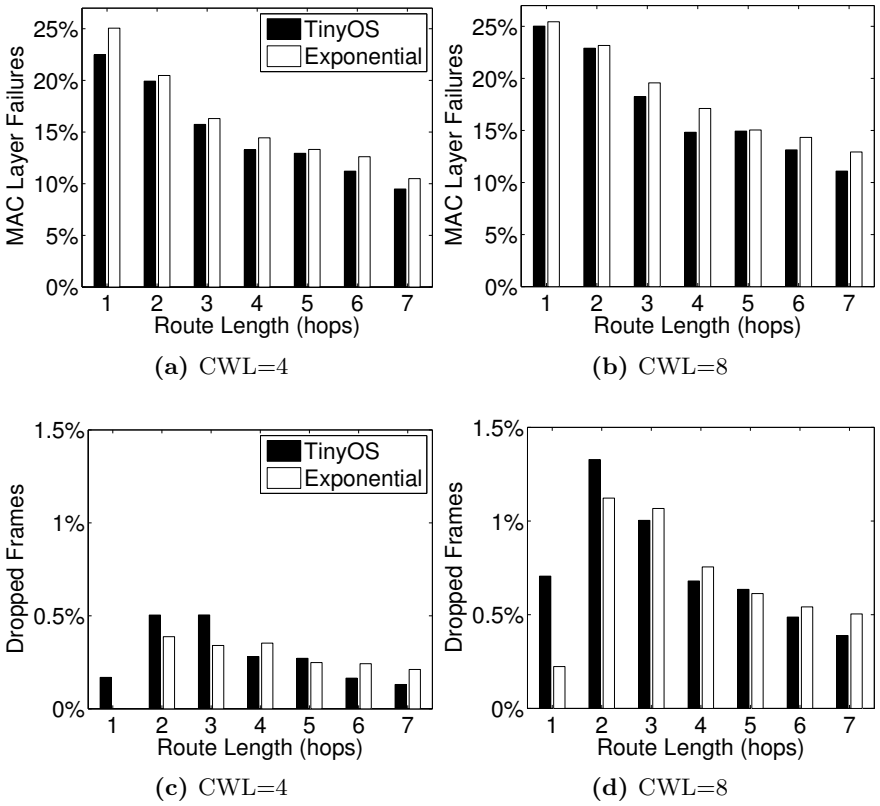
Figure 3.8: Percent improvement in goodput when using exponential backoffs compared to the default TinyOS backoff for different CWL values.

The performances of both backoff mechanisms are evaluated in the static routing scenario, and are depicted in Figs. 3.8 and 3.9. While the goodput improvement is much higher for a smaller CWL, the benefits are less predictable for a larger CWL. With a window size of 1, the benefit of using the IEEE 802.15.4 backoff increases with the number of hops. Each node on the route uses the shorter initial backoff, decreasing the average transmission delay at each hop, which results in shorter RTTs and therefore increases the goodput directly. As the CWL, and hence the contention increases, the exponential backoff leads to slightly more collisions in the medium, resulting in an increase of frame drops as shown in Fig. 3.9. This is a drawback, since the loss of packets decreases the performance of the transport layer. However, the benefit of shorter hop delays outweighs the performance loss induced by dropped packets and the higher amount of MAC layer retries.

3.7.1.2 MAC Layer Retries

The MAC layer provides one-hop reliability, where a common parameter is the maximum number of retries for a single packet transmission. TinyOS does not define a default value for this parameter. Hence, IEEE 802.15.4's default parameter value of 3 retries [5] is evaluated along with the values of 1, 5, and 7 in static scenarios.

Fig. 3.10 depicts the goodput and RTT results for static routes and for CWL values of 4 and 8. The evaluations show that for a CWL of 1, the performance differences between each configuration are negligible, and hence are not included in the results. This is due to the good average quality of the chosen route links and the fact that there is a low contention in case of CWL of 1. As seen in Fig. 3.10, the goodput increases with the number of allowed retries, even though the average RTT measured by the transport layer increases due to the delay that additional retries cause at the MAC layer. Moreover, additional retries ensure the one-hop transmission of packets, reaching a successful transfer ratio of 99% in the case of 7 retries, as seen in Fig. 3.11.



(a) CWL=4 **(b)** CWL=8

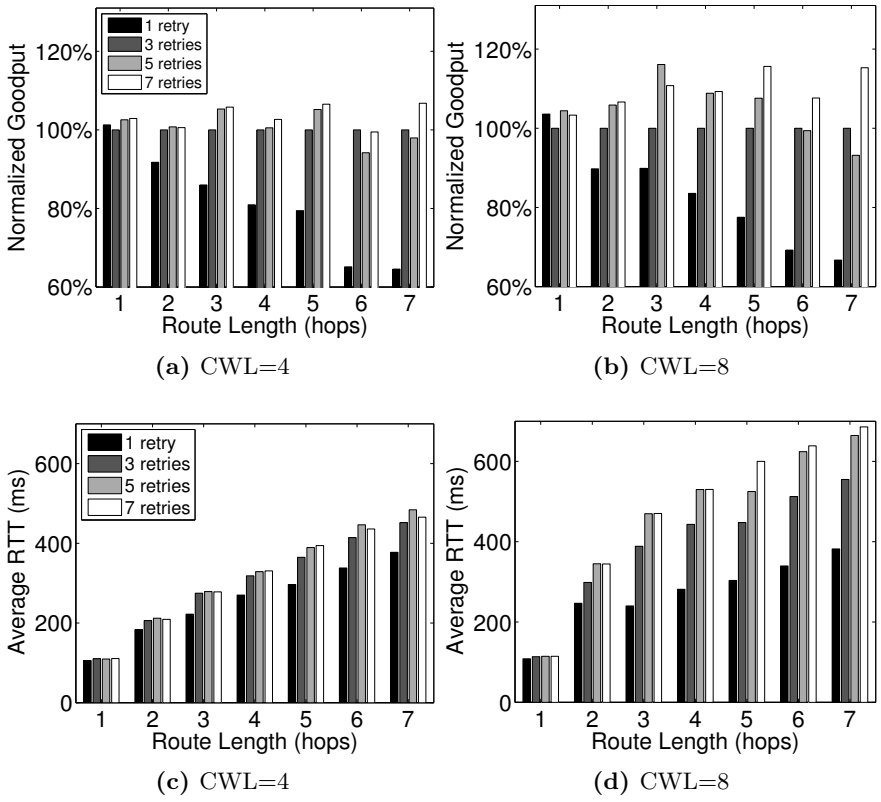
(c) CWL=4 **(d)** CWL=8

Figure 3.9: Effect of backoff mechanism choice on percentage of failed MAC layer transmissions and the overall frame drop rates for different route lengths and CWL values. The frame drops occur after the maximum number of unsuccessful MAC layer retries is exceeded.

The great increase of RTT when increasing the retry limit from 3 retries to 5 (and 7) is not only caused by the additional one-hop delays, but more importantly by the transport layer behavior. With fewer retries, the number of dropped packets increases. A packet loss will most likely provoke a RTO at the TL which resets the window size to 1. That is a reason for the smaller average RTT values observed for smaller CWL values.

3.7.2 NWK Layer

Since the NWK layer provides the routing protocol that is responsible for finding (multihop) paths between two nodes and maintaining them, it significantly affects the performance of end-to-end transmissions. Another key point when it comes to designing a NWK layer, is the routing metric used. An objective of this thesis is to analyze



(a) CWL=4 **(b)** CWL=8

(c) CWL=4 **(d)** CWL=8

Figure 3.10: Effect of maximum MAC layer retry values on normalized goodput (normalized by the goodput of 3 retries), and on the average RTT durations for different route lengths and CWL values.

how different routing protocol configurations and routing metrics affect the end-to-end performance.

The routing protocol determines what kind of mechanisms are used to find routes and to maintain them. nst-AODV operates like AODV. With the modifications described in Section 2.5.3.1, changes are introduced that reduce the memory footprint and the complexity of the protocol. Experiments reveal that there are several aspects of the routing protocol that affect the overall performance of end-to-end transmissions in WSNs. Two salient of these are:

- Message queue sizes
- Routing metric

Their details are discussed in the following subsections.

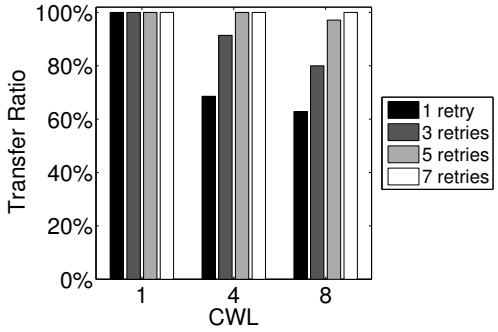


Figure 3.11: The average successful transfer ratio over all route lengths for different maximum MAC layer retry and CWL values.

3.7.2.1 Message Queue Sizes

An important aspect of nst-AODV and other routing protocols is the amount of buffer space available to store messages that need to be forwarded. A small amount of buffer space increases the probability of packets being dropped by relay nodes or nodes that generate data. Lost packets will cause higher layers to use retransmissions to recover from the losses. If large buffers are used under high traffic conditions, the delay may increase at nodes that are storing many packets at the network layer. Since the policy for the queues is normally FIFO, packets may stay a long time in the queue waiting to be processed. For CC purposes, the internal buffer states of nodes can be used to implement mechanisms like the ECN [48].

While the size of message queues has a high impact on the performance, an 'optimal' setting for the size cannot be provided. The amount of RAM available for queuing messages depends on the hardware of a device and the RAM consumption of the rest of the stack. In general, larger message queue sizes guarantee less packet drops due to full buffers (which can happen when there is congestion), which directly influences several performance metrics, like the PDR and the end-to-end delay. One important drawback when using large message queues is that the per-hop delay can be longer, since messages may remain longer in the queue before being processed. The effects on the performance of larger buffer queue and this particular phenomenon do not require any experiments, thus they are omitted in the evaluations.

3.7.2.2 Routing Metric

Since the routing metric determines which routes are used for end-to-end transmissions, it is one of the most relevant aspects of the routing protocol with regards on the performance that can be achieved in a WSN. Thus, the performances of using Path-DR [35] and HC metrics in dynamic routing scenarios are compared. Protocols like AODV in its basic version use the amount of hops between source and destination as

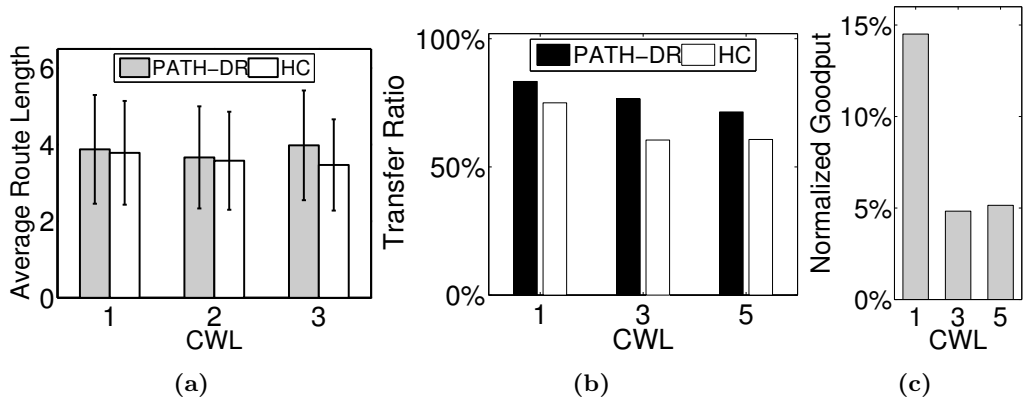


Figure 3.12: Comparison of using Path-DR versus using HC as the routing metric for (a) the average route lengths, (b) successful file transfer ratio, and (c) the percent improvement of the goodput of the Path-DR metric compared to the HC metric.

metric. The shorter a route is, the better it is considered. However, such a metric does not take into account the quality of the links on a path. Increasing the amount of traffic affects the overall performance of the route and nearby sections of the network, which is why the link qualities should be considered when choosing a route. As a metric that is aware of the link quality, the Path-DR metric is chosen for comparison. It estimates the overall PDR of a route based on the LQIs measured on each link traversed. The details of the Path-DR calculations are given in [35]. The route with the highest Path-DR is chosen for the data transmission. When using the HC metric, the shortest route is chosen, without taking the link qualities into consideration.

Dynamic routing tests are carried out with different CWL values for each routing metric, where random pars of endpoints are chosen in the network. The average route length and its standard deviation using both metrics are depicted in Fig. 3.12a. Routes chosen with a HC metric are shorter and their length has a smaller standard deviation than the routes chosen by the Path-DR metric. More importantly, the successful transfer rate is higher, as shown in Fig. 3.12b. The reason for both a higher goodput and a higher percentage of successful transfers lies in the fact that the path-DR metric chooses routes with high LQIs on the links, resulting in a low ratio of erroneous packets. In terms of goodput, the Path-DR metric delivers better results for all CWLs evaluated. Figure 3.12c shows the normalized improvement when compared to the HC metric.

Fig. 3.13 illustrates how the HC metric chooses the shortest path possible (one hop) to get to the destination. However, it does not take into account that the *natural LDR* for this link is only of 10%, which is why the link will most likely perform very bad. The term 'natural LDR' refers to the PDR for a link with losses that are introduced by bit errors originating from a lossy channel, but not from collisions or internal network

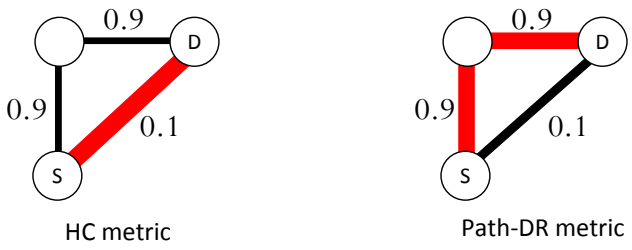


Figure 3.13: Routes chosen by HC and Path-DR metrics (red). The numbers indicate the natural LDRs for each link.

interference. One can see that in the given example the Path-DR metric chooses the path over a third node, leading to a two hop route, given the case that it estimates the LDRs correctly from the measured LQI values [1]. The LDR on the links chosen by the Path-DR metric is quite high and the total PDR is expected to be $0.9^2 = 0.81$, which is very high compared to the 0.1 of the path chosen by the HC metric. Choosing a route with a high PDR is preferable, as the total amount of packets to be sent will be lower due to the fact that there are less packet losses. It should be noted that this observation holds only for non-concurrent traffic scenarios and links with natural losses. Collisions and channel contention between several nodes can influence the performance of the routes chosen by the routing mechanisms (as shown in the evaluations of a HA scenario with concurrent traffic in Chapter 4.

For the remainder of this chapter in the dynamic routing scenarios the Path-DR metric is used to select the routes due to its superior performance.

3.7.3 Transport Layer

In this subsection the focus lies on the design criteria of the transport layer. It is common for custom transport layer protocols to adapt parts of the TCP protocol. Yet, only TCP in its entireness offers a reliable transmission of data packets, variable window sizes and a dynamic buffer allocation to support multiple data flows, which is why TCP is chosen for the evaluations in this chapter. Additionally, in the implementation used for the evaluations TCP has access to cross-layer information provided by the underlying nst-AODV, such as the route length and the RTT measured during the session initialization. This information may be used for the calculation of RTOs or the calculation of an upper bound for the CWL. Before focusing on these aspects of the transport layer, an evaluation of two end-to-end reliability mechanisms is done.

[1] The Path-DR routing mechanism maps measured one-hop LQI values to a LDR.

3.7.3.1 Acknowledgement Methods

In this subsection, the performances of two acknowledgement mechanisms are compared. These are i) Cumulative ACKs, where a single ACK can confirm the reception of several packets at once. The ACK packet is updated before its transmission, if there are new data packet arrivals; and ii) Positive ACKs, where each received packet is confirmed by a separate ACK.

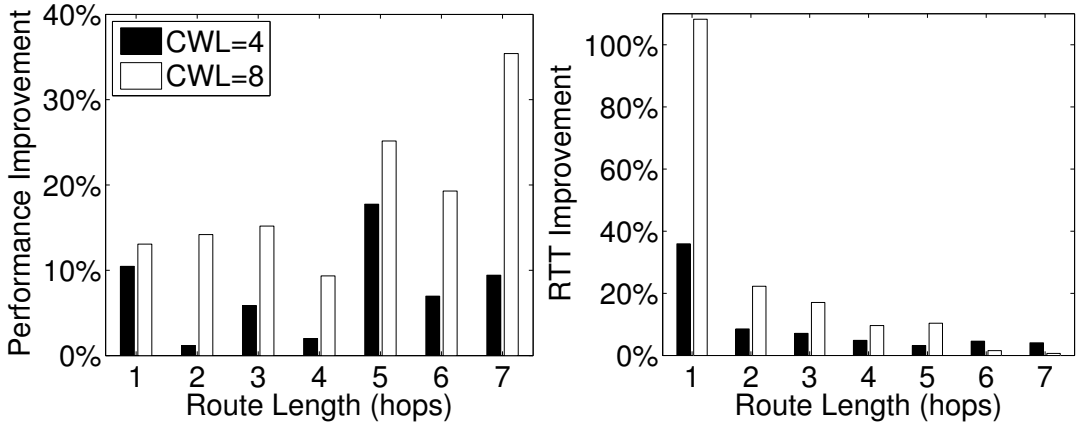
The basic version of TCP uses cumulative ACKs, which means that several data packets sent by the source are acknowledged with just one ACK. The drawback of using cumulative ACKs in this case is the loss of packets that are at the beginning of the sliding window of the sender. As the destination node expects the packets to arrive in order, the cumulative ACK just confirms the last valid packet. The destination node may have successfully transmitted a large amount of packets from the sliding window, but would still need to resend all packets beginning from the last confirmed packet. An improvement could be achieved in this case, if the destination node buffers all packets that arrive out of order. Then, after receiving the missing packet(s), the next ACK confirms all packets up to the last valid packet received. Selective ACKs offer a solution for this, as they may confirm series of packets and indicate if in the series one or multiple packets are missing. However, this case is not considered for the evaluations of this chapter, because of the simplicity of positive and cumulative ACKs.

Since the link quality based Path-DR metric is used to choose the routes in these experiments, the resulting PDR is expected to be high, and hence, the cumulative ACKs are expected to yield a better performance. Experiments on static routes are carried out to confirm this assumption. Fig. 3.14 shows how the average goodput and RTT for cumulative ACKs compares to positive ACKs for different CWLs. The results for CWL of 1 are similar for positive and cumulative ACK methods, since in this case they result in the same behavior, and hence are skipped.

As seen in the figure, cumulative ACKs always perform similar or better than the positive ACKs for both goodput and RTT. Moreover, higher CWL value results in a better improvement in these values. A linear increase is observed in the improvement of the goodput values with an increase in CWL value. The main cause for such increase is that less ACKs are sent in order to confirm the reception of data in the case of cumulative ACKs. With positive ACKs, the number of packets on both the upstream and the downstream contending for the channel increases with the ACK packets. This leads to an increment of the one way delays in both directions of the end to end transmission, resulting in a higher RTT.

3.7.3.2 RTO Calculation

In protocols like TCP, the firing of the RTO timer signals congestion under the assumption that congestion caused the packet to be dropped. For wired networks, this is a reasonable assumption, as the amount of bit errors is low and the main cause for packet losses lies in congestion. In WSNs the loss of a packet may also be the result



(a) Goodput comparison **(b)** RTT comparison

Figure 3.14: Performance improvement of cumulative ACKs on positive ACKs in terms of (a) average goodput, and (b) average RTT.

of congestion, but also due to bit errors or radio interference. Distinguishing between those causes of packet losses is not possible from the transport's/application's layer point of view. This is why the behavior of the protocols after a packet loss is the same in both cases, congestion and bit errors.

There are three common actions that protocols like TCP can take after a packet loss:

- Reducing CWL
- Retransmitting data
- Increasing the RTO

By increasing the RTO, the source node reduces the amount of data that is injected into the network over time if another loss is detected. The choice of an adequate RTO has proven to have a high impact on the overall performance of the network. The RTO determines mainly two aspects of a data transmission: If a data packet or an ACK is lost, the RTO tells the source node how long to wait before considering the packet to be lost. If the RTO is too large, the loss of a packet may be noticed with a large delay. This may decrease the goodput considerably, as time is wasted waiting for an ACK that will never arrive. On the other hand, if the RTO is too small, the source node may cause spurious retransmissions. This is the case, when the previously sent data packet or the ACK sent in reply to this data packet are still on their way through the network and would arrive at the source node eventually. Due to a small RTO, the source node may not wait for the ACK to arrive and assumes either the data packet

or the ACK has been lost. Apart from the fact that unnecessary traffic is generated with an additive increase/multiplictive decrease (AIMD) mechanism, the source node reduces the CWL, which again reduces the end-to-end data rate. Since, by default, TCP assumes congestion, the common reaction of the transport layer is to reduce its window size to 1.

The RTO may be calculated in many different ways, ranging from using static values, to advanced formulas that include several variables. In the following, the effect of three different algorithms on the Transport layer performance is compared:

- The calculation as proposed in the RFC 6298 for TCP, where a SRTT and a RTTVAR are calculated and used to compute an RTO, as detailed in Section 2.6.1. The SRTT is used to estimate the RTT between source and destination and to have a value for RTTVAR that indicates how much the RTT varies. This algorithm is supposed to adjust to changing conditions of end-to-end data transmissions. This calculation method is referred to as 'Karn's' method.

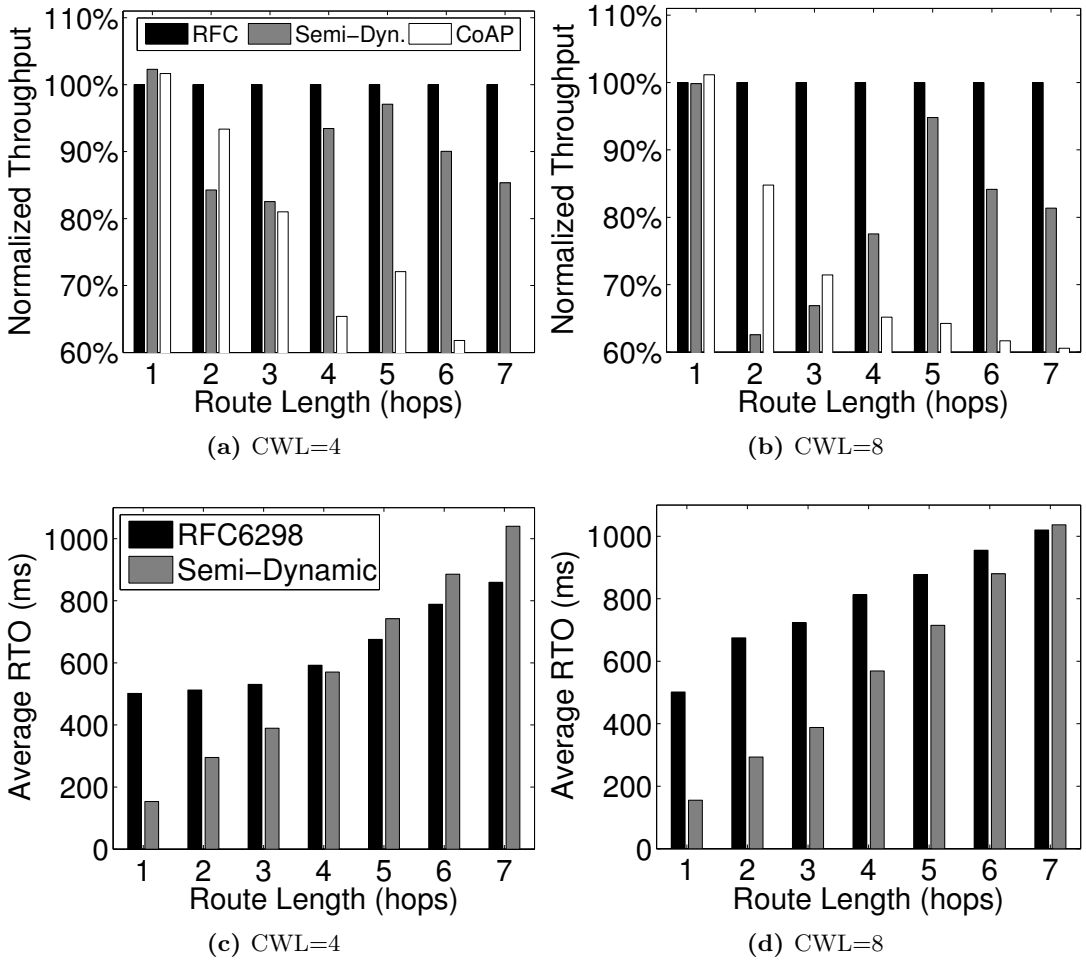
- An approach that calculates the RTT once during the TCP session establishment. If no sessions are used, this RTT value can be taken from lower layers, if a cross-layer approach is considered. This could be the time it took to find a route, as long as there was a route discovery. This calculation method is referred to as 'semi-dynamic' method.

- An initial RTO that is calculated randomly from a fixed interval and then doubled for each retransmission. This behavior is taken from the CoAP specification [15], which defines an initial RTO value interval between 2 and 3 seconds. This dithering is applied to avoid synchronization effects across different transmissions that start at the same time.

All RTO algorithms use an exponential backoff to increment the RTO timer after a loss.

Fig. 3.15 compares the performance of the semi-dynamic algorithm and CoAP relative to RFC 6298's algorithm for static routes. With a CWL of 1, neither packet losses, nor spurious retransmissions can be observed, since the nodes observe very low contention for the channel. Consequently the performance is identical for all algorithms, and are not displayed. By increasing CWL, however, differences in the performance of the three algorithms can be observed that depend highly on the number of hops. Figs. 3.15a and 3.15b show that the fully-dynamic RFC 6298 algorithm outperforms the other two algorithms in routes with multiple hops.

The cause for the difference is the way the RTO adapts to the real conditions of the network. Using an initial RTO may work well, if the value chosen is close to the real RTT. However, if it is below the RTT, it will cause spurious retransmissions and it may wait to long to react to losses. Or, if it is chosen too large, time may be wasted for an ACK that never arrives in case of a packet loss. As the network topology normally is not known to the nodes of a WSN and the available bandwidth may be very low, the



(a) CWL=4 **(b)** CWL=8

(c) CWL=4 **(d)** CWL=8

Figure 3.15: Effect of RTO algorithm choice in terms of normalized goodput (normalized by the goodput of RFC6298's algorithm) and RTOs for different CWLs.

real RTT for CoAP transmissions may be quite high. This is why the value suggested for the CoAP is relatively high with an initial value situated between 2 and 3 seconds, compared to the initial RTO of RFC 6298 of 1 second. CoAP's initial RTO needs to be considered a conservative value, as it is assumed that messages may require a long time to travel through the WSN and also may leave the local WSN via gateways and connect to any other Internet device, which could lead to long delays/RTTs. Also, the processing time required by the application on top of the communication stack may add an important delay. If during the RTO timer period no confirmation is received, the packet is sent again and the duration of the RTO timer is doubled (up to a maximum of 48 s, with the highest initial backoff and after 4 retries, which is the suggested maximum of retries).

However, the CoAP algorithm does not adapt in any way to the actual average RTT of a transmission. Thus it does not perform well when there are packet losses (larger routes and larger CWLs). The semi-dynamic RTO is only calculated once for the whole transmission, thus the overall performance of this algorithm depends strongly on the RTT value initially measured. Given the limitation of a single RTT measurement it seems coherent that the dynamic algorithm, which updates the RTO repeatedly, performs better than the CoAP and semi-dynamic algorithms for different route lengths. It can also be observed that the differences get smaller, when the average RTOs of both algorithms are close to each other. In one-hop scenarios, no packet losses could be observed, therefore the CoAP and semi-dynamic algorithms perform similar to the dynamic RFC 6298 algorithm.

3.7.3.3 Congestion Window Management

An important issue that needs to be taken into account when designing a transport layer protocol is the congestion window management. This is especially important for constrained devices, since internal buffer space and available radio bandwidth are strongly limited. Bandwidth calculation algorithms are proposed for IEEE 802.11 networks, among which some are integrated in TCP [55]. These algorithms try to estimate the available bandwidth to adjust their congestion window size correspondingly and set a CWL. The CWL defines the upper bound of the sender's window. Choosing this value affects the overall performance strongly, as it determines, if the channel gets underused or overused when using a too low or too high limit respectively. The limit of the achievable bandwidth is defined by the Bandwidth Delay Product (BDP) between two endpoints. TCP with its AIMD mechanism aims to find the upper BDP limit by increasing its window while no RTO happens, as explained in [80]. When reaching the upper limit given by the BDP, further increasing the throughput causes congestion. Since it is not feasible to calculate the BDP of a transmission in a WSN without using advanced estimation mechanisms, an easier solution is desired for constrained devices.

The authors of [80] state that the the spatial reuse factor k determines how many nodes may transmit at the same time, without interfering each other in IEEE 802.11.

The grade of spatial reuse is directly related to the BDP that indicates the bottleneck bandwidth of a path. The paper deduces that the spatial reuse and the amount of hops determine the optimal CWL for chains of nodes using the formula:

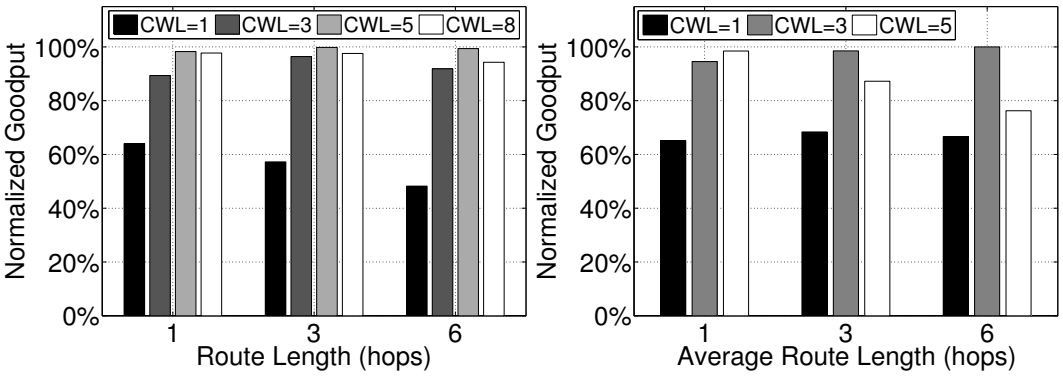
$$\text{BDP} = kN, \text{ with } k = 5 \tag{3.1}$$

In this formula N is the round-trip HC and it includes k, which is a variable that may change depending on the network topology. With an adequate value for k, the CWL can be adjusted to use the optimal bandwidth for a single-flow transmission of data. The results of [80] are based on simulations, where it is assumed that nodes are located at the edge of each other's transmission range and that the interference range is exactly two times the transmission range. Through experiments, now it needs to be determined if a factor k can also be specified for real IEEE 802.15.4 networks. Therefore the results of the experiments are used to confirm if this k exists.

In the following, a simple bandwidth estimation algorithm that tries to prevent the congestion window from overshooting is analyzed by defining an upper CWL. The main two motives to implement an upper limit for the congestion window are the following: First, a large CWL results in a higher buffer usage at intermediate nodes, increasing the probability of dropped packets. Second, a higher degree of contention is to be expected, since more data-packets and ACKs travel on the forward and backward path of a route. On the other side, using a small, static CWL may under-utilize the available bandwidth. Experiments are done, to see the effect of the CWL value on the performance achieved on static and dynamic routes.

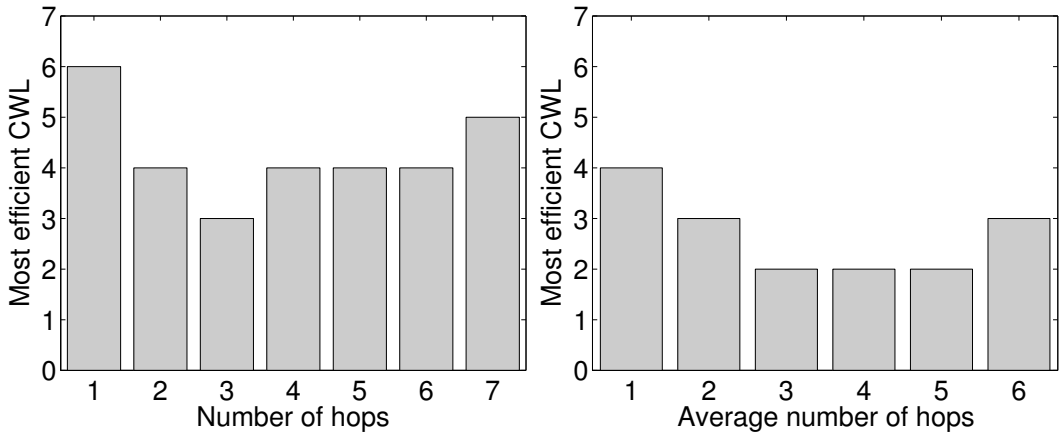
The CWL value after which the goodput does not increase more than 1% by increasing the CWL is defined as the *recommended CWL*. The goodput results for a set of CWL values are compared to the recommended CWLs in Fig. 3.16 for given (average) route lengths. The results for static routes show that independent of the route length, after reaching the recommended CWL, the goodput flattens out. Through Fig. 3.16a, it can be observed that the CWL at which the curves flatten out depends on the route length. While having a higher CWL value than the recommended CWL does not change the goodput considerably for static scenarios, it causes a higher degree of congestion along the route. This reflects in a greater amount of MAC retries and dropped packets observed in the experiments. These effects would especially be important for multiple flows scenarios.

The evaluation of the dynamic scenarios shows that a more conservative CWL policy leads to better results. Long routes require the source node to reduce the window size considerably. Fig. 3.16b shows that the goodput does no longer flatten out with increasing CWLs. Instead, a larger window size than the recommended CWL may cause the goodput to drop for routes with multiple hops. The main reason for this behavior is the mechanism with which nst-AODV reacts to failed MAC layer transmissions. A MAC layer transmission failure is interpreted as a link break, causing a route repair. This stops the transmission temporarily and requires the exchange of control messages throughout the network to find a new route between the two endpoints. Since the



(a) Static scenarios **(b)** Dynamic scenarios

Figure 3.16: Comparison of normalized goodput (normalized by the goodput of the recommended CWL) for different CWLs in (a) static and (b) dynamic scenarios.



(a) Best CWL (static) **(b)** Best CWL (dynamic)

Figure 3.17: Recommended CWLs for different route lengths in static (a) and dynamic (b) scenarios.

probability for link breaks gets higher with the increase in CWL and with it the degree of contention increases, the amount of route repairs also increases. Fig. 3.17 reflects the findings for an optimal CWL setting in the static and dynamic traffic scenario.

The results also show that a factor k for the spacial reuse cannot be found for the given indoor WSN environment. There is no linear dependency between the number of hops of a route and an optimal CWL. The main reason for these results is the fact that transmission and interference ranges are not circular and change from node to node due to differences in the hardware and differing signal propagation in the indoor environment.

As the length of the route is known by the nst-AODV, a potential performance improvement can be achieved in existing congestion window based transport layer protocols, by applying a cross-layer (CL) interaction between the transport and the routing layers to adjust the CWL. Such a CL approach is investigated in Chapter 4.

3.8 Conclusions

In this chapter, the effects of different design criteria of an IEEE 802.15.4-based protocol stack on the performance of end-to-end transmissions are evaluated in the Castelldefels sensor grid. The default settings of mechanisms and parameter from the MAC, NWK and transport layers are compared to alternative ones in a single file transmission scenario with static and dynamic routes. The stack covers a set of protocols that are of common use in WSNs. The results show that the default settings may not deliver the best performance for the protocols and scenarios considered, and that generally performance can be improved by adjusting these settings or replacing certain mechanisms completely.

At the MAC layer, the backoff mechanisms and the type of ACKs used for one-hop reliability mainly affect the per hop delay. This has a direct impact on the throughout, since data can be delivered at different speeds, depending on the used methods. The default backoff applied in IEEE 802.15.4 leads to better results when compared to the alternative mechanism implemented in the widely used TinyOS. The number of allowed MAC layer retransmissions has an important effect on the PDR and RTT values. A higher number of allowed retries than specified by default in the IEEE 802.15.4 standard can yield to important improvements of these metrics. The improvement gets larger with increasing amounts of traffic. One main drawback of increasing the number of MAC layer retries is the increase of the one-hop delay, since packets may be kept longer in the buffer before either succeeding with the transmission of the packet or dropping the packet after spending all retries. Further, it is expected that in networks with high density and in the state of traffic congestion, increasing the number of retries can lead to further congestion, which may result in congestion collapse.

The evaluations of the NWK layer show that settings like the message queue size have an important impact on the performance. Since in networks of constrained devices the memory capacities of devices are very limited, a trade-off between performance and

RAM consumption needs to be found. Another NWK layer mechanism of great impact on the performance are the routing metrics used to determine routes in nst-AODV. The comparison of the HC and Path-DR metrics reveals that the LQI-aware Path-DR metric is capable of finding better routes, yielding in noticeable PDR and goodput improvements.

The largest improvements, however, are observed at the transport layer when using cumulative ACKs instead of positive ACKs that increase with the CWL. The much lower number of end-to-end ACKs that is sent with the cumulative approach reduces the network congestion considerably, increasing the bandwidth that is available during end-to-end communications between devices. Similarly, the comparison of different RTO algorithms reveals that the adaptive RTO algorithm used by TCP (RFC 6298) yields the best results, independently from the CWL and route length. The RTO calculations described in RFC 6298 adapt to the ongoing traffic and the observed network conditions, avoiding spurious retransmissions or long idle times in case of losses. The performance of the RFC 6928 RTO algorithm is followed by the semi-dynamic RTO and the CoAP algorithms. The latter shows a poor performance in nearly all tested configurations due to its static configuration that does not adapt to the network conditions at all. This is an important finding that requires further analysis, since CoAP will be used as main application layer protocols for the IoT. More evaluations of the CoAP RTO calculation method and an in-depth analysis of CoAP's CC mechanisms can be found in Chapters 4, 5, and 6, respectively.

Overall, the evaluations carried out in this chapter provide a basic understanding of the effects on the performance of different protocol layer mechanisms and parameter settings. Moreover, the interaction of different mechanisms across the analyzed protocol stack has shown to have a relevant impact on the end-to-end performance. The knowledge gained about which are important mechanisms and parameter settings in a WSN protocol stack is applied to the ZigBee stack for wireless HA networks in the next chapter. Applying similar evaluation methodologies as the ones presented in this chapter, a thorough performance analysis of the default ZigBee stack and alternative configurations in a complex use case scenario is carried out.

3. PERFORMANCE EVALUATION AND IMPROVEMENT OF AN IEEE 802.15.4-BASED WSN PROTOCOL STACK

4

Performance Evaluation and Improvements of a ZigBee Home Automation Network

In the previous chapter, crucial mechanisms and parameters settings of an IEEE 802.15.4-based protocol stack for WSNs were investigated. Methods to improve the network performance of this stack were provided after evaluating communications in a real testbed, where a batch of data between any two nodes of a network over static and dynamic routes was transmitted. It could be shown that different mechanisms and parameter settings of the communication protocol have a direct impact on different performance metrics. The results obtained give important insights on the potential network performance gains by adjusting parameters of the protocol communication stack and serve as guidelines for its configuration. Yet, the exemplary use case that forms the basis for the evaluations does not cover the characteristics of more intricate real life networks. Thus, in this chapter, the scope of the work is extended to a ZigBee HA scenario with a higher degree of complexity. The evaluations include multiple parallel transmissions, sleepy nodes, and different traffic patterns between network devices. This contribution was published in [BGDP14].

4.1 Introduction

In our daily lives, wireless HA networks (WHANs) [81] promise an important role, as the devices in common use can easily be connected with each other to enhance user comfort and to allow efficient home management. The potential of such wireless devices increases further as the connection to the Internet becomes possible for them. WHANs, a particular form of WSNs, present a strong alternative to wired networks for HA systems, and they have been a target of several standardization efforts [82, 83].

Among these efforts, ZigBee has been considered as a strong candidate to be adopted for WHANs [8]. The ZigBee protocol stack has been the subject of several performance studies [84, 85, 86, 87], some of which focus on the use of ZigBee for WHANs [88, 89]. Various detailed evaluations have been carried out for specific ZigBee layers [90, 91, 92]. On the other hand, ZigBee has been defined on the basis of components developed and widely studied by the research community. However, these components have been considered separately. In fact, a holistic evaluation of the ZigBee protocol stack, which is vital to observe the overall performance of the specification, is not publicly available in the literature.

The generic ZigBee specification [7] defines a set of protocol mechanisms and parameters, the latter with default and allowed values. Moreover, for specific application types, ZigBee defines application profiles. One of these is the ZigBee HA profile [93], which includes specific settings for WHAN scenarios. A trivial and mainly the common way of configuring the network parameters and mechanisms in a deployment is to keep their default settings from the specifications. Hence, it is crucial to understand the potential performance gains that can be achieved by tuning the settings of the protocol stack. Therefore, an evaluation of the default and the allowed settings defined in the ZigBee specification and HA profile is needed.

In this chapter, the end-to-end network performance of ZigBee WHANs following a holistic approach is studied, which considers parameter settings and the design of mechanism from different protocol stack layers. The goal behind these evaluations is to quantify possible performance improvements through the tuning of these settings and, also, to provide a future direction for the evolution of the ZigBee specifications. Performance evaluations are done: (i) for the default protocol stack configuration; (ii) for various alternative configurations that are also compliant with the ZigBee specifications; and (iii) for alternative settings that do not disrupt the ZigBee architecture, but are however not compliant with the specifications.

For the study, various application scenarios that have been derived according to the ZigBee HA profile and IETF specifications for HA are defined [82]. Experiments are carried out in the Castelldefels grid (Section 3.5), where different network conditions are defined and evaluated. Specifically, different network topologies are defined and the effect of the transmit power setting in these is evaluated. Also, different types of traffic generation models along with varying corresponding loads are defined. Moreover, the source and destination nodes for a given traffic load are partially randomized to devise the average behavior of the evaluated approaches.

The investigated performance metrics are end-to-end PDR, end-to-end delay and energy efficiency, which are three common and crucial network performance criteria for WHANs. These metrics are evaluated for various combinations of settings of key network parameters and mechanisms from different stack layers.

Based on the findings from the experiments, two new protocol stack configurations are proposed, namely the Recommended-Compliant configuration and the Recommended-Unrestricted configuration. The two recommended configurations achieve much higher

end-to-end PDR, a lower end-to-end delay and a higher energy efficiency than the default ZigBee settings in the experiments. Numerically, the Recommended-Unrestricted settings yield up to a 33.6% relative PDR increase, up to a 66.6% delay decrease and up to a 48.7% energy efficiency improvement in comparison with the default ZigBee settings. These improvements are achieved without incurring any overhead to the network. In addition, detailed insight into the factors within and across the analyzed ZigBee layers that have a significant influence on the performance is provided. Note that the findings of this work are not limited to ZigBee networks and also provide useful guidelines for the design and performance improvement of other IEEE 802.15.4-based networks.

The rest of the chapter is structured as follows. Related work (to this specific part of the investigation) is presented in Section 4.2. A short reminder of ZigBees main features is given in 4.3. The default configuration of the ZigBee protocol stack and the investigated network parameters and mechanisms are introduced in Section 4.4. In Section 4.5, the evaluation environment is described. In Section 4.6, the results of the evaluations are presented. Based on the results gained, the two improved configurations of the ZigBee protocol stack are proposed and evaluated in Section 4.7. In Section 4.8, the recommended settings are evaluated in alternative network scenarios. Section 4.9 concludes the chapter.

4.2 Related Work

To the best knowledge of the author, no performance analysis of the mechanisms and parameters of the complete ZigBee communication protocol stack has been provided at the point of carrying out the evaluations presented in this chapter. Publicly available studies focus on a single ZigBee layer or on testing the performance of default ZigBee.

Within these studies, only a few use a real testbed environment to carry out their evaluations, while the majority backs its results on simulations. However, simulations provide a lower degree of accuracy in some aspects that are relevant to the outcome of experiments, such as the radio signal propagation in a real environment, the interference observed and the actual delay incurred by the devices in the network.

From the few papers that base their evaluations on a real testbed, ZigBee-capable hardware is designed and tested in an indoor environment using 51 nodes in [84]. The focus of the evaluation lies in determining long-term loss rates and transmission throughput. Contrary to the investigation presented in this chapter, only the capabilities of the default ZigBee protocol stack configuration are analyzed. At a smaller scale, a WHAN of four ZigBee devices is used in [88] to demonstrate that a specific WHAN architecture defined by the authors is stable and that the co-existence with IEEE 802.11 is feasible. A more realistic approach with several indoor and outdoor WHAN testbed setups that include approximately 10 real nodes per scenario is studied in [89]. Transmission failure rates and radio transmission ranges are evaluated depending on the positioning of the nodes within the scenarios. Amongst the scenarios evaluated, some

are specifically designed to represent WHANs. However, the presented analysis does not focus on the influence of different protocol stack layer configurations on the network performance, only giving results about PDRs achieved by a single configuration in each of the scenarios. The small size of this WHAN testbed and the simple performance analysis do not give insights into how the ZigBee stack performs with a multitude of devices and a larger network extent, or for different ZigBee settings.

There exist evaluations of ZigBee networks that rely on simulations, focusing mainly on a specific network mechanism or parameter and on its optimization to achieve a better network performance. The work presented in [85] analyzes the performance of two basic routing methods for ZigBee networks and presents the performance improvement of a new routing protocol with a simulator. The sole focus of improving the routing protocol independently from other layers and a simulation-based approach deliver insufficient information about the realistic ZigBee stack performance in WHANs. Furthermore, by using a simulation environment, several additions to the default ZigBee protocol stack are proposed in [86] to enable a reliable data transmission control method for situations where data congestion problems are expected. The authors of this work demonstrate the contribution of their proposal with simulated networks of up to 100 nodes and compare the achieved performance to the one obtained by the default ZigBee protocol. However, the transmission of large amounts of data to a sink node is not characteristic of a WHAN scenario, where small amounts of data are exchanged and the one-to-one traffic pattern must also be considered [82].

Kohvakka *et al.* mathematically analyze MAC layer mechanisms of a large scale IEEE 802.15.4 network that performs ZigBee network operations inside a cluster-tree topology and establish run-time operation models of the nodes [87]. The models and results are confirmed via simulations. The two performance metrics evaluated in that work are power consumption and goodput. However, the large network size used for these evaluations of up to 1560 nodes exceeds the realistic size of a WHAN [94], and the use case of data gathering (*i.e.*, many-to-one traffic directed to a sink node) is not representative for a typical WHAN scenario.

Ostermaier *et al.* show that an energy-efficient connection of devices to the IoT over devices that use low-power Wi-Fi is possible, even when using standards like HTTP and TCP/IP [95]. Wu *et al.* state that Wi-Fi technology is adequate to serve as the interface for home area networks to smart meters in smart grid environments [96]. While the market share of WHAN technologies, such as low-power Wi-Fi and also BLE, is growing steadily, ZigBee currently dominates the WHAN market [97]. Table 2.3 [98] in Section 2.3.5 shows a comparison of the key features, such as bitrates or message sizes used by several WHAN low-power technologies.

4.3 ZigBee's Main Features

A detailed summary of ZigBee's mechanisms that are part of this chapter's investigations can be found in Section 2.8.1. As a reminder, in the following a summary of the

most relevant mechanisms is given to ease the understanding of the ensuing evaluations.

At the NWK layer ZigBee uses the AODV routing protocol, introduced in Section 2.5.3. As routing metric, ZigBee uses an LQI/LDR-based algorithm that estimates the quality of a path. As long as a LDR is not available (this is the case, when no packets have been transmitted over the link), the average LQI of the link is used to approximate the LDR. After transmitting packets over the link, the real LDR can be calculated. For each link the delivery probability p_l is used to calculate the link cost, which is an integer between 0 and 7. Formula 2.8 is used to calculate the link cost $C\{l\}$. Formula 2.9 is used to implement the HC metric.

By adding up all the link costs from the source to the destination node, the path cost is calculated. The routing algorithm always chooses the path with the lowest total cost for data transmissions.

The APL layer provides reliability with end-to-end ACKs and allows fragmentation of data into several packets. While end-to-end reliability may be used depending on the application, fragmentation is normally not necessary, due to small payload sizes of ZigBee packets. If fragmentation is used, ZigBee allows using a window to send various packets at once. However, in profiles such as the HA profile, it is recommended to be set to 1. With a window limit of 1, the end-to-end reliability is implemented by a simple stop-and-wait mechanism.

4.4 Protocol Stack Configuration

Because parameters and mechanisms defined for the ZigBee communication protocol stack layers are numerous, only a specific subset that is expected to have a high impact on the network performance is evaluated. The subset is also partially chosen to confirm the results obtained from the evaluation of an IEEE 802.15.4-based protocol stack in the previous chapter. Specifically, the effect of MAC retries from the MAC layer, the routing metric from the NWK layer and the RTO algorithm from the APL are investigated. Figure 4.1 gives an overview of the parameters and mechanisms considered in the evaluations. In addition, the default and the alternative settings investigated are also shown in the figure.

4.4.1 MAC Layer

In Chapter 3, the effect of allowing different maxima for the amount of MAC layer retries has proven to be significant for the performance of applications that involve bulk data transfers between two nodes, such as file transfer applications. In this chapter, the maximum amount of allowed MAC layer retransmissions defined by the *macMaxFrameRetries* parameter from the IEEE 802.15.4 specification is analyzed with regard to its influence on the one-hop reliability and the overall network performance in the investigated WHAN scenarios. TApart from doing evaluations within the range defined in the ZigBee specification, *i.e.*, from zero to seven retransmissions, also the

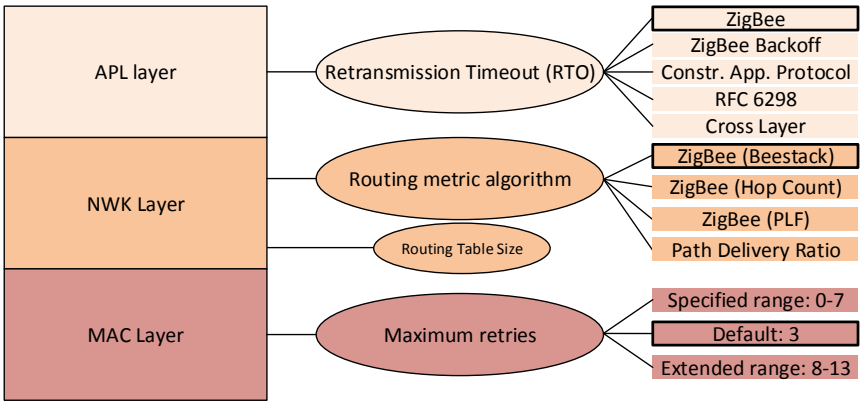


Figure 4.1: Simplified overview of the ZigBee communication protocol stack and the parameters/mechanisms that are evaluated from different layers and the default and alternative settings for these parameters/mechanisms. The default settings are highlighted with bolded boxes.

effect of using values from an extended range of up to 13 retries is analyzed, to see if one-hop reliability increases further and how the network performance changes. On the other hand, alternative backoff methods for accessing the channel have been tested in the previous chapter too, showing that the impact of this mechanism can be relevant for single end-to-end large file transfers. In this chapter the impact on the performance is determined for concurrent transmissions of small amounts of data.

4.4.2 NWK Layer

The evaluations in the previous chapter of the impact of the routing metric on the end-to-end performance revealed that in the analyzed scenario the Path-DR metric yielded better results in terms of PDR and goodput. These results are applicable for single end-to-end transmissions in WSNs, while there is no concurrent traffic. In this chapter, this evaluation is applied to a WHAN scenario, where simultaneous communications between different devices are possible.

The devices in WHAN applications have the capability to communicate with any other device in the network [82, 99], making the mesh topology the proper choice for the network topology. Thus, an important MAC layer setting in the evaluation carried out in this chapter is the use of the beacon-less mode, since the ZigBee mesh network topology only supports such a mode, as described in Section 2.8.1. Most of the devices are FFDs, except for a few RFDs that do not participate in routing. From the mechanisms provided by the NWK layer for the mesh topology, the impact of the routing metric on network performance is evaluated. The default routing metric uses Equation (2.8) to calculate the link costs. Although an LDR value is required by the Equation, that value may not be available to a ZigBee node. Alternatively, the ZigBee

specification suggests using a mapping from LQI to a link cost, the details of which are not specified and are left to the implementation. A prevalent ZigBee implementation from Freescale, BeeStack [100], calculates the LQI-based on the RSSI of a packet and converts it into a link cost, the mapping of which is shown in Fig. 4.2.

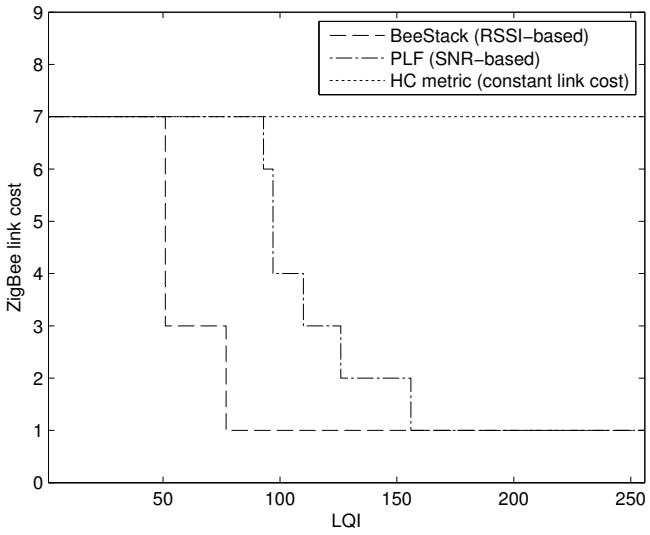


Figure 4.2: Mapping of LQI to link cost values by the BeeStack [2], HC and piecewise linear function (PLF) [3] routing metrics.

The link cost mapping used in the BeeStack implementation is assumed as the default link cost setting for the investigations carried out in this chapter, the performance of which is compared to the performance achieved with two other interpretations of the ZigBee metric. The first consists in using constant link costs resulting in the HC metric, *i.e.*, using Equation (2.9) as the link cost. The second is a piecewise linear function (PLF) mapping from LQI to LDR derived by the physical experiments in [35] and then applying Equation (2.8). The LQI value in this case is calculated from the correlation value provided by the radio transceiver as an SNR estimate [21]. The PHY layer implementation obtains the LQI by correlating the first eight symbols after the start-of-frame delimiter of a packet. The PLF mapping is also shown in Fig. 4.2. Note that the BeeStack bases its LQI calculation on the RSSI of a packet, while the PLF metric uses the SNR estimate obtained by the correlation to determine the LQI value. The reader is encouraged to see the survey on radio link quality estimation by Baccour *et al.* for detailed information on these metrics and alternative ones [101].

As an alternative routing metric, the Path-DR metric [102] is included in the evaluations. Path-DR estimates the LDR of each link during a route discovery. In the evaluations, this estimation is also implemented based on the LQI to LDR mapping derived by Gomez *et al.* [35]. As already explained in Section 2.5.2, by multiplying the estimated LDR values at each link, an end-to-end PDR estimation is obtained. The

Path-DR metric selects the route with the highest estimated PDR. This metric is not ZigBee-compliant, since it does not apply the ZigBee path cost calculation algorithm. The reasons to include this metric in the evaluations are two-fold. First, it has been shown that this metric results in better network performance in an indoor scenario than other route selections metrics, such as HC and max-LQI [102]. Secondly, a metric that is not ZigBee-compliant is considered to be a benchmark for the ZigBee-compliant metrics.

4.4.3 APL

At the APL, the impact of the RTO algorithm on the network performance is further evaluated. In the previous chapter, it could be demonstrated that for single batch transmissions of data the dynamic RTO calculations applied by the TCP algorithm [52] generally yield the best results.

To prove whether this also applies to WHAN communications, the performances of four alternative RTO algorithms are evaluated and compared to the default one applied by ZigBee. The RTO algorithms can be split into two main components: the initial RTO assignment and the backoff method. The initial RTO value is used for the first transmission of a packet. If a loss is detected, *i.e.*, if the RTO timer expires and the packet needs to be retransmitted, a backoff mechanism can be applied, which increases the RTO value. In the following, a short introduction to the five RTO algorithms investigated is given. A summary of these five RTO algorithms can be found in Table 4.1.

Table 4.1: The RTO algorithms investigated and their main aspects.

Algorithm	Initial RTO	Backoff method
ZigBee	1.5 s	-
ZigBee + Backoff	1.5 s	BEB
Constrained Application Protocol (CoAP)	[2 s, 3 s]	BEB
CL	$4 \times RTT_{RE}$	BEB
RFC 6298	1.5 s	BEB

ZigBee

The ZigBee specification defines a static RTO value for end-to-end transmissions RTO_{ZB}, which is calculated by:

$$RTO_{ZB} = 0.05\ s \times (2 \times nwkcMaxDepth) + 0.1\ s, \tag{4.1}$$

where the *nwkcMaxDepth* parameter denotes the maximum depth of the network. As a result, the RTO values used by ZigBee are constant and the same for all of the nodes

of a network. If a retransmission is necessary, no RTO backoff mechanism is applied. Hence, the same RTO value is used for both the initial and the follow-up RTOs. The maximum depth that was achieved in the testbed was measured and found to be 14. This leads to an RTO value of 1.5 s for the ZigBee protocol stack investigated in the evaluated testbed according to Equation (4.1).

ZigBee + Backoff

Since the use of backoff is common in RTO algorithms and to be able to analyze how adding a backoff affects the performance, the tests are also carried out with an alternative RTO algorithm that adds the BEB mechanism to the default ZigBee RTO algorithm. This algorithm will be referred to as 'ZigBee + Backoff', and the RTO value calculated by this algorithm is referred to as $RTO_{ZB\text{-}BO}$ in the evaluations.

CoAP

CoAP is designed to be used in WHANs [103]. As in ZigBee, end-to-end ACKs are used to implement reliability at the APL (see Section 2.7.3). Therefore, the approach of CoAP to calculate an RTO value is evaluated in this chapter as an alternative to the ZigBee approach. In CoAP, the initial RTO value for a transmission, RTO_{CoAP}, is randomly chosen from an interval, where:

$$RTO_{CoAP} = [2\ s,\ 3\ s]. \tag{4.2}$$

This relatively large initial value has been chosen to cope with high delays that may be observed when communicating with devices over the Internet. Additionally, the action that is triggered by the data packet (for example, the request of a complex calculation) may have a large processing delay. To avoid further network congestion, the follow-up RTO values are doubled with each retransmission in CoAP.

Cross Layer (CL)

In this more dynamic approach, already part of the evaluations in the previous chapter, it is proposed to use the NWK layer information to calculate the RTO value for a transmission. The approach consists in measuring the RTT during the route establishment procedure (RTT_{RE}) and using it to calculate an RTO value, RTO_{CL}, by multiplying the RTT measured with a constant K, which is set to four by default. Hence,

$$RTO_{CL} = 4 \times RTT_{RE}. \tag{4.3}$$

RTT_{RE} measured during the route finding procedure is an approximation of the RTT expected for the subsequent data transmissions. To avoid possible congestion of

the network, the BEB method is applied. This type of RTO mechanism cannot be used by RFDs for data transmission, since they do not establish routes. Thus, the CL RTO mechanism is only applied to data transmissions of FFDs, while the RFDs use the default ZigBee RTO mechanism for this approach.

RFC 6298

TCP uses a more complex and dynamic algorithm to calculate the RTO value, as explained in Section 2.6.1. The initial RTO value, according to the RFC, is 1 s by default. In the following evaluations, it is set to 1.5 s to be identical to the value ZigBee uses. Additionally, a minimum RTO value is strongly recommended by the RFC. This led us to decide to use such a minimum RTO with a value of 1 s. The BEB method is employed for retransmissions in this algorithm.

4.5 Reference Scenarios

This section gives a detailed explanation of the WHAN reference scenarios evaluated through testbed experiments. These scenarios have been derived from the ZigBee HA profile [93] and the HA routing requirements defined by the IETF ROLL working group [82]. The design of the scenarios has been split into three main aspects: the logical roles of the nodes, the network traffic patterns and the evaluation setup. In the following, these scenario aspects are explained in detail.

4.5.1 Roles of the Nodes

From a use-case point of view, the WHAN setup represents wireless nodes associated with one or several electronic devices of everyday use. The nodes thus are commonly mains-powered [82]. The devices connected to a node in a WHAN may be sensors, actuators, controllers and other kinds of appliances. The wireless nodes are possible sources and destinations of information transmitted in the network. Three roles for the nodes that cover most of the typical WHAN use cases are defined:

(Role A) Nodes that have only a one-to-one relationships with other nodes of the network, for example, a light switch and a light bulb.

(Role B) Nodes that have one-to-one relationships with certain nodes of the network and that also produce data for a destination of many-to-one traffic. An example is a node with a motion sensor that causes an alarm for a security center upon noticing movement, while the node also periodically sends the measurements of a temperature sensor to a heating, ventilation and air conditioning (HVAC) system.

(Role C) Nodes that have one-to-one relationships with certain nodes of the network and are the destinations of many-to-one traffic. An example is an HVAC controller that collects temperature information and that is able to remotely control heaters, ventilators, *etc.*

4.5.2 Traffic Patterns

In WHAN scenarios, the amount of user data transmitted is usually small and, most of the time, only consists of a few bytes [82, 93]. The actual amount of traffic depends on the number and the type of WHAN devices and how frequently they are used. To evaluate the performances of the network settings investigated, the degrees of device activities are varied in the network, as well as the roles of the devices. As a result, three different traffic scales are defined for the evaluations: low, medium and high traffic. This allows one to observe how the WHAN performs under a variety of situations and reveals how the performance of a certain communication protocol stack configuration scales with increasing traffic load. Two possible causes for the generation of data are considered:

1. Aperiodic events: An aperiodic event can be, for example, an alarm generated by a motion sensor or by the command to turn on/off a device caused by toggling a switch. The amount of bytes that it takes to encode the information that needs to be transmitted is assumed to be three bytes. The total amount of bytes for the payload and the headers of such a message fits in the payload of an IEEE 802.15.4 packet, only requiring transmission of a single frame.

2. Periodic events: For example, a periodic event can be a heartbeat signal from the nodes of a network that needs to be sent to a security center regularly or temperature information that is periodically collected and reported to an HVAC controller. The messages include aggregated information, such as sensor data and/or plain text, as part of status reports or log files, to name some examples. As a consequence, the total amount of bytes of a periodical event is assumed to surpass the maximum payload available in one MAC frame. Thus, three packets are assumed to be generated for a periodic event.

Depending on the roles of the nodes described in Section 4.5.1, the type of traffic they generate is defined as shown in Table 4.2. For example, a light switch (Role A) may only generate aperiodic events, whereas a temperature sensor (Role B) would generate periodic events for temperature statistics, but it also would generate an aperiodic event if the temperature exceeds a certain threshold. Aperiodic events always generate data for a random destination within the network.

Periodic events are generated with a fixed interarrival time of 60 s that does not vary with the traffic scale. Instead, the number of nodes producing periodic traffic is varied to achieve different traffic loads in the network. The interarrival time values for aperiodic events, t_s, are defined to follow an exponential probability distribution

function, the individual values of which can be generated through the use of a uniformly random variable, R, with the formula:

$$t_s = \frac{-ln(1-R)}{\lambda}, \tag{4.4}$$

where R is uniformly distributed between zero and one and λ is the average number of events per minute. Different traffic scales are achieved by varying λ and the number of nodes generating the aperiodic event data. For low, medium and high traffic loads, λ is defined to be 0.75, 1.25, and 1.75, respectively.

Table 4.2: Overview of possible events for the three node roles.

Role	Aperiodic events	Periodic events
A	Yes	No
B	Yes	Yes
C	Yes	No

4.5.3 Evaluation Setup

As for the evaluations carried out in Chapter 3, the Castelldefels testbed is used for experimental evaluation (see the details in Section 3.5). Nodes are programmed with TinyOS and the fundamental mechanisms and parameters of ZigBee's communication protocol layers to carry out the evaluations are implemented. In contrast to commercial implementations of the ZigBee protocol stack, like BeeStack, with a fix set of changeable options and settings, the stack implementation allows to freely modify all of its parameters and mechanisms. There may exist differences between the performance of this proprietary implementation and that of commercial ZigBee implementations. The TinyOS implementation of the protocol stack introduced in Section 4.4 can be accessed at [104].

The physical phenomena of the testbed environment, like multipath propagation and signal scattering, are typical effects observable in a WHAN environment that are difficult to reproduce realistically in simulators. Furthermore, it is necessary to take into account that the physical channel conditions of a WHAN may change over time. These changes affect the behavior of the network and happen due to several reasons, such as the interference caused by other devices operating at the same frequency band as the one used in the WHAN. Further reasons for varying physical conditions include moving objects or persons. The Castelldefels testbed exhibits these requirements.

To imitate the typical characteristics of WHANs, the sensor grid needs to be configured adequately. In order to assure that connections with multiple hops are part of the evaluations, the transmit power level is configured to 1, which corresponds to a transmission power of –33 dBm. In the alternative scenarios defined in Section 4.8,

evaluations are carried out also using a transmit power level of 2. The average number of hops between two randomly chosen nodes with this configuration lies below four, with a transmit power of 1, and below two, for a transmit power of 2, which captures the most common cases in WHAN scenarios [83]. Furthermore, the nodes are configured to work on Channel 26 of IEEE 802.15.4, avoiding most of the possible Wi-Fi interference.

The measurements are carried out for different scenarios by varying the number of nodes and changing traffic patterns. For these evaluations, three different topologies are defined: the basic topology, and two alternative topologies, namely the dumbbell and the square topologies. In the basic topology, 52 nodes are configured to be FFDs and five nodes to be RFDs with the previously introduced configurations of the transmission channel and transmit power level.

The basic topology is illustrated in Fig. 4.3. In the low traffic scale, four nodes of type B generate periodic traffic for the destination node C_1, whereas all other nodes are of type A. The amount of nodes with role B increases linearly as the traffic scale increases (eight nodes at medium and twelve at high traffic scales). For each added group of B nodes, a new destination for their periodic traffic is added: C_2 for medium traffic and C_3 for high traffic. The additional B and C nodes are included in the grid by replacing the corresponding nodes with an A role.

The five RFDs represent battery powered devices that are used sporadically, such as light switches. They generate aperiodic traffic with the arrival rate defined for the low traffic scale in all analyzed scenarios and are not the destinations of packets from other nodes, except for end-to-end ACKs. The RFDs are configured to sleep until an event that requires a data transmission causes them to wake up. After transmitting the data, and while waiting for an end-to-end ACK, RFDs will poll their associated FFDs every 500 ms. After each poll, RFDs enter the sleep state until the next scheduled poll. If an RTO at the APL is triggered, the device wakes up and polls the associated FFD for the end-to-end ACK for which the RFD is awaiting. If no end-to-end ACK is available, the RFD retransmits the packet and continues with the cycle of sleep and poll. After receiving the ACK, the transmission is considered successful, and the RFD will sleep until the next data transmission. In addition, and in compliance with the ZigBee HA profile, RFDs will wake up at least once every minute to poll their associated FFD for data. The black node in the lower, left corner is used to align the internal timers of the nodes by sending synchronization messages at the beginning of a test run. Two other nodes (*i.e.*, the remaining black nodes in Fig. 4.3) were chosen randomly to be left unused with the goal of adding further irregularity to the grid, at the same time leading to areas of low connectivity.

Experiments in the grid are done during nighttime, where low human activity and external wireless interference is observed. A relatively steady physical environment facilitates the comparison of different system parameter sets evaluated over time. Still, there may be small variances in the channel conditions when doing the experiments at different times. For this reason, to test a single configuration of the communication

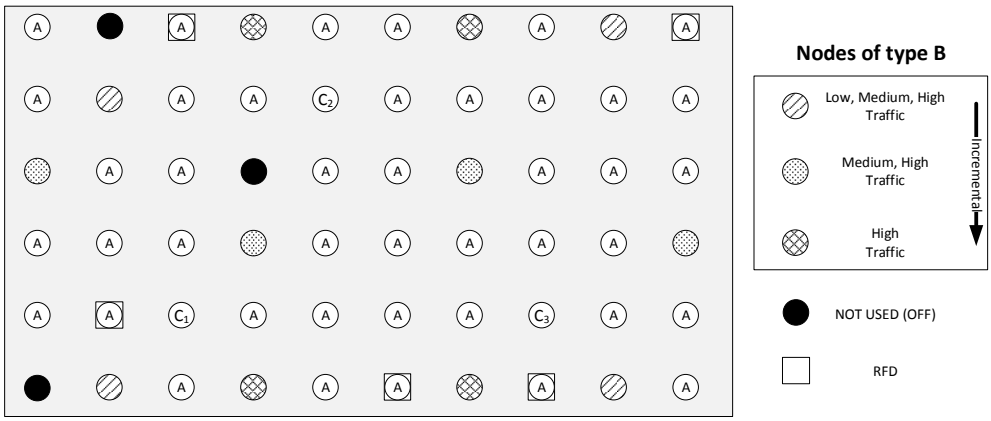


Figure 4.3: Testbed layout and Roles A, B and C assigned to the nodes of the testbed for the basic topology. Nodes that are not of Role B or of Role C at a certain traffic scale are of Role A.

protocol stack, 43 test runs are carried out, which are distributed over several days and add up to a total test duration of eight hours for each configuration.

At the beginning of each test run, all nodes are reset and the internal timers for aperiodic or periodic events are initialized. After completing a test run, the seeds of the nodes' random number generators are altered. Varying the seed changes the behavior of the timers for aperiodic events and the destinations selected for aperiodic data transmissions. The following section presents the results of the evaluation carried out in the basic topology.

4.5.4 Evaluated Performance Metrics

Additionally to the metrics introduced in the previous chapter (see Section 3.4), the Energy Efficiency metric is used for the performance evaluations carried out in this chapter. In low-power wireless sensor networks the efficient use of energy is an important performance metric for battery driven devices. The energy efficiency indicates how many Bits of (useful) information can be transmitted with 1 Joule. Efficient communications protocols allow the network to transmit large amounts of data while keeping the energy consumption low. The efficiency depends on many factors, but mainly it depends on how long devices spend in an active state, that is while transmitting or receiving data.

The energy efficiency indicates the average energy that is consumed by the RFDs to deliver one bit of useful application payload to corresponding destinations. The energy consumption is calculated by observing the behavior of the end devices, in particular by measuring the durations of different states and applying the energy consumption values listed in [105] for each state. Since a different transmit power level is employed, the energy consumption of the nodes at the transmit state (I_{tx}) was measured.

4.6 Evaluation Results: Influence of Each Layer to the Overall Performance

This section presents the results of the evaluations of the ZigBee stack configurations presented in Section 4.4, which have been carried out within the basic topology scenarios described in Section 4.5.3. At the MAC layer, the impact of the maximum number of MAC layer retries is evaluated. The effects on the performance of using different routing metrics at the NWK layer are also analyzed in this section. Finally, the influence of the APL RTO algorithm is studied. Each set of measurements for a specific layer is carried out, while the rest of the communication protocol stack is configured with the default settings (see Section 4.4).

4.6.1 MAC Layer

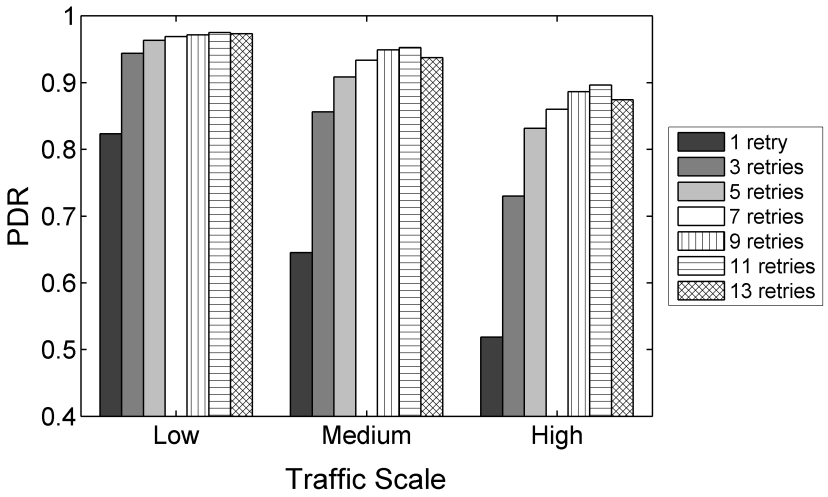


Figure 4.4: The overall PDR for different values of *macMaxFrameRetries* in different traffic scenarios.

The experiments with different values of *macMaxFrameRetries*, the parameter that defines the maximum number of MAC layer retransmissions, shows an important correlation between the setting of this parameter and the network performance. Changing *macMaxFrameRetries* affects the overall PDR noticeably, as depicted in Fig. 4.4. As seen in the figure, for the values in the allowed value range from zero to seven, as specified by IEEE 802.15.4 and hence by ZigBee, a higher amount of allowed retransmissions results in a higher PDR. With the default configuration of three, the overall PDR decreases from 94.3% to 85.6% and to 72.9% for low, medium and high traffic, respectively. Moreover, the difference in the overall PDR between the default configuration that allows three retransmissions and the two alternatives of using one and

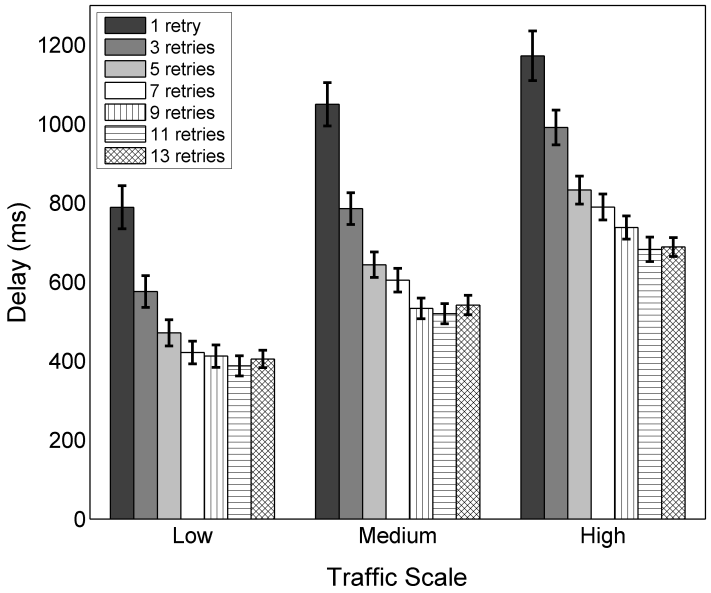


Figure 4.5: Average end-to-end delays observed with the corresponding 95% confidence intervals for different values of *macMaxFrameRetries* in different traffic scenarios.

seven retransmissions, respectively, becomes larger with increasing traffic. It reaches its peak value at high traffic with a relative degradation of 28.9% when only using one retry and a significant relative improvement of 17.8% when using seven retries. The average end-to-end delay and the energy efficiency evolve correspondingly, as shown in Figures 4.5 and 4.6. While the delay is relatively high for a low number of allowed retransmissions, it decreases as more retransmissions are allowed. The energy efficiency improves similarly, as the energy consumption per bit decreases with an increasing value of *macMaxFrameRetries*.

The reason for these results can be derived from the average data LDR, which depends on *macMaxFrameRetries*. Figure 4.7 shows that the LDR for different traffic scales increases rapidly with values above one for *macMaxFrameRetries*, flattening out with higher values when considering the allowed range of values for this parameter. As the traffic increases, the benefit of augmenting the number of allowed retries is higher, as packet losses due to intranet interferences and collisions become more frequent.

Since it is necessary to transmit packets over multiple links, the LDR has a major impact on the overall PDR, determining with which probability the packets reach the destination node. The end-to-end delay similarly depends on the LDR. In a theoretical case of a 100% LDR, the end-to-end delay only depends on the number of hops between the source and the destination and on the time spent at each hop. In a practical case, links are lossy and the LDR adopts lower values, like the ones observed during the experiments. If a loss happens, which is more probable in links with lower LDRs, the APL end-to-end reliability mechanism will retransmit the packet. However, this

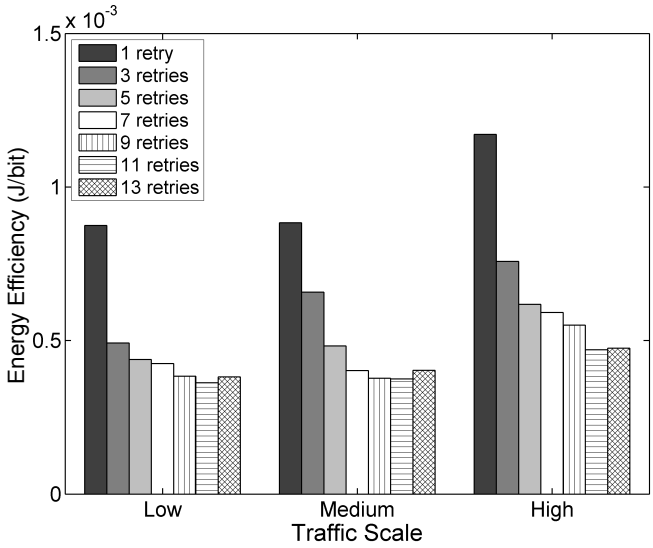


Figure 4.6: Average energy efficiency for different values of *macMaxFrameRetries* in different traffic scenarios.

requires the source node to wait for the RTO timer of the APL to expire, adding significant delay to the transmission. Therefore, as the LDR decreases, the end-to-end delay increases. The energy efficiency shows a similar improvement as PDR and delay for an increasing amount of allowed retransmissions. Packet losses that lead to APL retransmissions and that cause the RFDs to waste polls while waiting for end-to-end ACKs that never will arrive decrease the energy efficiency.

On the other hand, an important cross-layer mechanism depends on the value of *macMaxFrameRetries*. According to the ZigBee specification, an unsuccessful MAC layer transmission (after using all retries) is interpreted as a link failure at the NWK layer, which triggers the route repair mechanism. As a consequence, the network is flooded with RREQs, adding a significant amount of traffic to the network that may decrease the performance of ongoing transmissions. A high amount of allowed retries reduces the overall amount of route repairs that are initiated during the tests (see Fig. 4.8), decreasing the control message overhead and reducing the network congestion.

When using values higher than those allowed by the specification (9, 11 and 13 retries), at all traffic scales, further improvements are observable for nine and 11 retries. However, with a maximum of 13 retries, the PDR and energy efficiency drop and the delay increases. With 13 allowed retries, the congestion created by repeatedly trying to transmit frames at medium and high traffic rates becomes significant, leading to a degradation of the overall performance, as one-hop delays become larger and buffer overflows occur more frequently due to increased buffer occupation times. This also

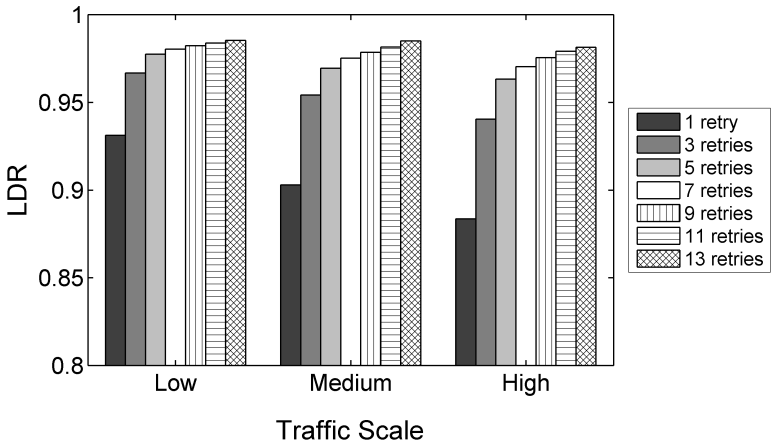


Figure 4.7: LDR for different values of *macMaxFrameRetries* in different traffic scenarios.

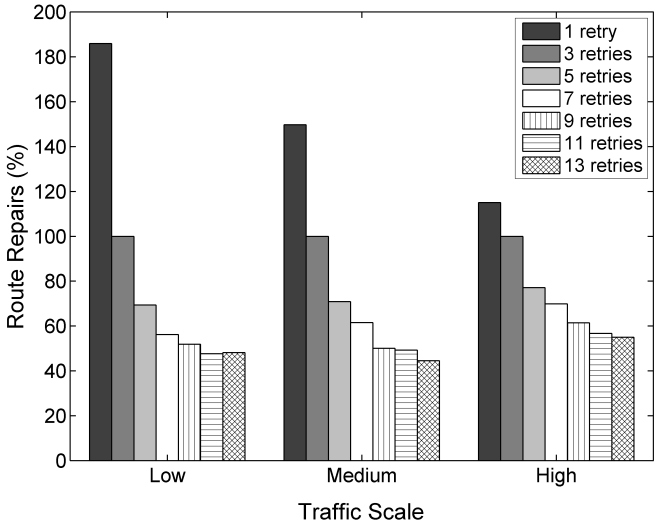


Figure 4.8: Relative amount of route repairs for different values of *macMaxFrameRetries* taking the default value of three as the base.

reflects in the energy efficiency, which increases for nine and 11 retries and decreases for 13 retries.

Judging from the results obtained in the experiments, it is highly recommended to increase the default value of *macMaxFrameRetries* for ZigBee HA profile systems to the maximum the specification allows, as it benefits the PDR, delay and the energy efficiency for end-to-end transmissions. Moreover, the allowed value range of *macMaxFrameRetries* should be increased, as further performance gains can be achieved by using values from the extended value range.

Apart from analysing the influence of the *macMaxFrameRetries* parameter on the end-to-end network performance, the impact of the CSMA/CA backoff mechanism is evaluated. The default TinyOS backoff and the backoff as specified by IEEE 802.15.4 are compared (see Section 3.7.1.1 for details). TinyOS provides a proprietary backoff mechanism that behaves quite differently from the IEEE 802.15.4 mechanism. A clear improvement of the end-to-end delay was observed in the evaluations of a single large file transmission. However, in contrary to the clear results obtained in the previous chapter, the performance difference in the WHAN scenario is negligible. No clear benefits or drawbacks were observed in the experiment results.

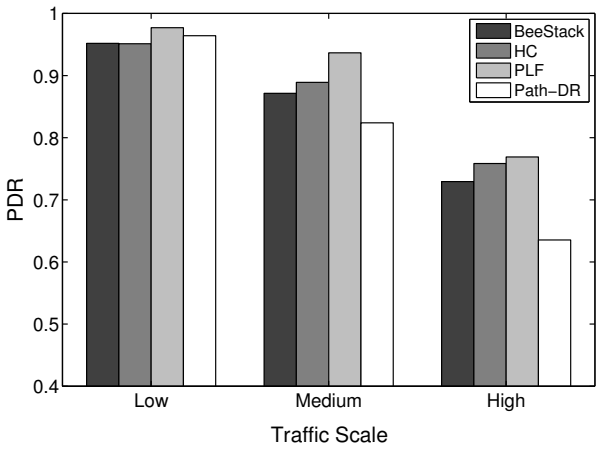


Figure 4.9: Comparison of the overall PDR for different routing metrics and traffic conditions.

4.6.2 NWK Layer

This subsection studies how the overall network performance of the commonly used BeeStack routing metric compares to the performance of the HC, PLF and Path-DR routing metrics. The routing metric only is relevant for the FFDs that are participating in the routing. RFDs are not capable of routing; they are associated over a one-hop relationship to an FFD that is responsible for routing messages to and from the RFD. However, the routing metric also has a relevant impact on the performance of the end

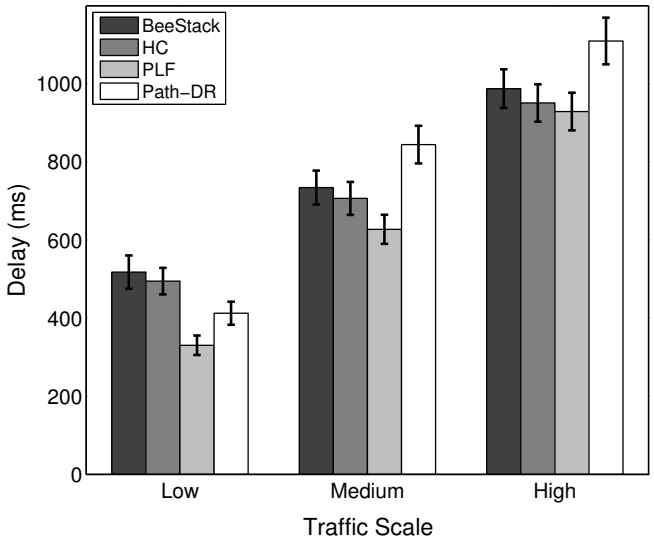


Figure 4.10: Comparison of the average end-to-end delays with 95% confidence intervals for different routing metrics and traffic conditions.

devices, as will be shown later, since all of the messages they are sending and receiving are routed by FFDs through the network.

Figure 4.9 shows that the PLF metric always performs best in terms of the overall PDR. The HC, BeeStack and Path-DR metrics follow in the given order, except for the low traffic case, where the Path-DR metric performs the second best. A strong degradation of the Path-DR performance can be observed for higher traffic loads, contradicting the observations made in the previous chapter for single end-to-end transmission flows (see Section 3.7.2.2).

A higher PDR is correlated with a lower end-to-end delay, as can be seen in Fig. 4.10. A similar dependency has been observed in the previous subsection between the PDR for the maximum MAC layer retransmission parameter and the resulting end-to-end delay. As explained in the analysis of the MAC layer results, the energy per bit should decrease with shorter end-to-end delay and higher PDR. The measurements confirm this assumption, as can be seen in Fig. 4.11.

Detailed investigations have been carried out to find out the reasons for the performance differences between the considered metrics and for the unexpected worse performance of Path-DR for the medium and high traffic scales. The main reason for the differences in the performance of the metrics lies in the characteristics of the chosen routes. The decision criteria of the routing metrics determine the number and the quality of the links of the resulting paths.

In the experiments, the HC metric results in the minimum average path length, as expected and as seen in Fig. 4.12, while the average length of routes chosen by Path-DR is the largest one. The average lengths of the routes chosen by the BeeStack and the

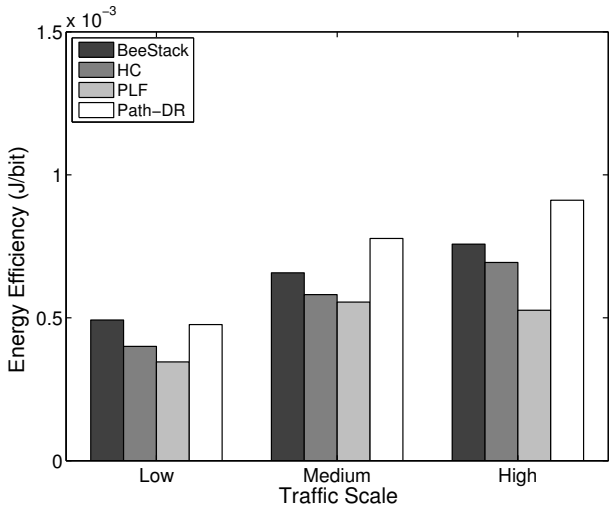


Figure 4.11: Comparison of the average energy efficiency for different routing metrics and traffic conditions.

PLF metrics lie between the ones chosen by the HC and Path-DR metrics. This can be explained by the fact that the BeeStack and PLF metrics apply an additive formula to calculate the path costs. Contrariwise, the Path-DR metric applies a multiplicative formula to calculate the expected PDR. Due to the multiplicative path cost calculation, Path-DR prefers longer routes with higher quality links over shorter routes with lower quality links. As an example, a route with a multitude of hops that have an expected LDR of almost 100% is preferred by the Path-DR metric over a single hop with a low expected LDR. This is unlikely for the BeeStack and PLF metrics, since each hop increments the path cost, making longer routes expensive. This behavior causes the routes chosen by the PLF and BeeStack metrics to be shorter than those selected by the Path-DR metric.

The end-to-end PDR of a route strongly depends on the link qualities along the route. Except for the HC metric, the routing metrics base their choice of routes on the measured link qualities. As indicators of the link qualities, LQI values are used, since a high LQI normally is correlated with a high LDR. Therefore, routing metrics that base their decision on measured link qualities expect a high end-to-end delivery ratio over these routes. The performance of the routing metrics depends on the mapping from LQI to link costs or expected LDR values. The BeeStack and PLF metrics use a coarse granularity mapping that converts LQI values into link costs, being three steps in the BeeStack metric and seven steps in the PLF metric. On the other hand, the Path-DR metric defines 61 steps when transforming the LQI values into expected LDR values. Metrics that use finer granularity mappings are able to detect a wider spectrum of link qualities and identify high quality links with higher certainty.

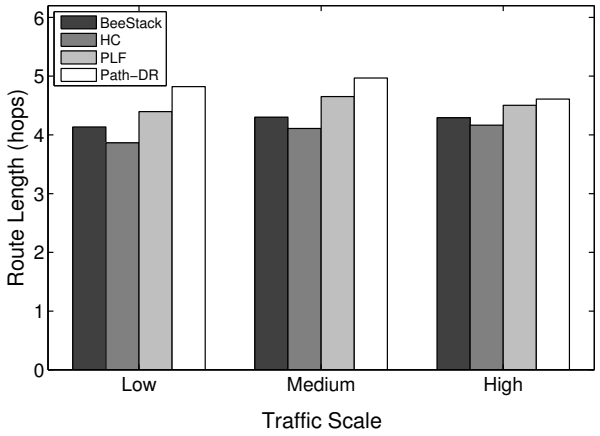


Figure 4.12: Average lengths of routes chosen by different routing metrics and traffic conditions.

However, the route length and the link qualities of a route alone do not decide how well a transmission over this route performs. An important factor that is not considered by any of the metrics is found to be network congestion. A higher amount of traffic increases the degree of contention among nodes inside the network, leading to interference and collisions. This directly affects the average number of retries used by the IEEE 802.15.4 MAC layer when transmitting unicast data frames. Table 4.3 demonstrates that the number of tries observed for each metric increases with the traffic, which is an indicator of congestion. Experiments have shown that LQI values measured with and without internal network interference are almost the same. One reason could be that only the link qualities from the correctly received packets are considered by the link quality-aware metrics, while the packets with the errors that result from collisions or interference are not considered. This leads to the conclusion that the measured LQI values do not serve as indicators for network congestion. Because of this phenomenon, the link quality-aware metrics are not able to identify congested links. If a metric tends to choose longer routes, like the Path-DR metric does, the fact that additional transmissions are necessary with each hop can lead to further congestion and to a lower end-to-end performance.

Furthermore, the experiments have shown that the link quality-aware metrics will more likely overuse the links with high LQI values, leading to local congestion. This issue especially affects the Path-DR metric. Figure 4.13 depicts an example where the Path-DR metric only finds one best route, while the evaluated ZigBee metrics are able to find three best routes of equal costs over different links. The restriction of Path-DR to routes of very high quality, due to its fine link and route cost granularity, increases the amount of transmission attempts over them and is an important source of congestion. As a consequence, the Path-DR metric performs poorly at medium and high traffic scales, when compared with the ZigBee metrics (including the HC metric), as shown

Table 4.3: Average number of tries that are performed by the MAC layer to transmit one unicast data frame for different metrics and traffic scenarios.

Metric	Low Traffic	Medium Traffic	High Traffic
BeeStack	1.32	1.44	1.63
HC	1.30	1.45	1.64
PLF	1.22	1.38	1.59
Path-DR	1.28	1.47	1.67

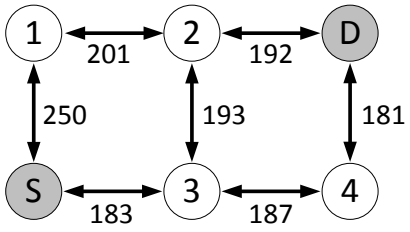


Figure 4.13: Example topology, where the Path-DR metric only finds one best route between the source (S) and destination (D) (S-1-2-D), while the ZigBee path cost metrics (HC, BeeStack and PLF) find three routes of equal cost (S-1-2-D, S-3-2-D, S-3-4-D). The numbers on the links represent the LQI values of the links. Note that all ZigBee path cost metrics will assign the same cost to all of the links shown in this example (see Fig. 4.2).

in Figures 4.9–4.11. In contrast, the link and path cost granularity of the BeeStack and the PLF metrics is significantly coarser than that of Path-DR. This increases the probability of determining the same path cost for routes that have slightly different link qualities when the BeeStack and the PLF metrics are used. Therefore, the BeeStack and PLF metrics provide greater path diversity, which naturally tends to reduce network congestion.

The PLF metric chooses links with high LQI values, but in contrast to the Path-DR metric, it does not restrict its set of available routes to a small subset of optimal routes, as illustrated in Fig. 4.13. The combination of the greater path diversity and sufficient link quality-awareness leads to a better performance of the PLF metric for all traffic scales. On the other hand, the BeeStack metric performs worse than the HC and PLF metrics. The coarse link and path cost granularity is one reason. Figure 4.2 shows that 70% of all LQI values are mapped to the same link cost, which means that the BeeStack metric assumes that all of the links in this large interval are of similar, if not equal, qualities. The second reason for the inferior performance may originate from the fact that the LQI value is obtained from RSSI measurements of the radio transceiver in the BeeStack implementation, instead of relying on correlation values. Finally, note that the HC metric is not link quality-aware and, thus, may select both high and low quality links, but it guarantees the shortest route.

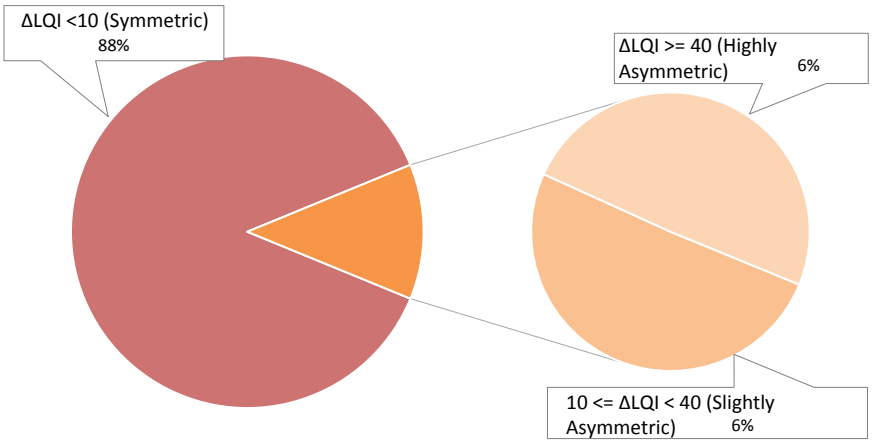


Figure 4.14: Distribution of symmetric and asymmetric links. Latter ones are split into slightly and highly asymmetric links.

Another important issue observed throughout experiments in the testbed environment is the negative effect of asymmetric links on network performance. Similar observations have been made in the literature [106]. Normally, the forward and backward links of a route between two nodes are of similar qualities in terms of LDR or LQI. However, it has been observed in the testbed environment that approximately 6% of all links between nodes are strongly asymmetric. These links are normally of very high quality in one direction and have a very bad or no connection at all in the other direction. To measure the symmetry, all available links of the Castelldefels grid have been observed during one day and the LQI value for each of the links in both directions was measured. Fig. 4.14 shows the distribution of symmetric and asymmetric links. Links with a difference in the LQI between up- and downlink of less than 10 are considered to be symmetric. Differences in the LQI of at least 10 but less than 40 are considered to be slightly asymmetric and any links that show a difference of the LQI of more than 40 are considered to be highly asymmetric. From all available links, 6% are highly asymmetric, in many cases only providing connectivity in one direction.

For metrics that take into account the link qualities of the forward route towards the destination, asymmetric links can result in unsuccessful route discoveries, as RREPs may not be able to return to the source node. Figure 4.15 illustrates this situation. In this scenario, a route discovery would not provide a valid route, even though there exists at least one. The path cost for a route that includes the asymmetric link would be better for all metrics than the path costs for the alternative routes that do not include asymmetric links. The issue of asymmetric links affects especially link quality-aware metrics. This leads to higher channel usage (all retries are spent to forward RREPs that never reach their destination nodes) and also to higher buffer occupation inside the nodes (the buffers are occupied longer, while retries are spent). While this phenomenon

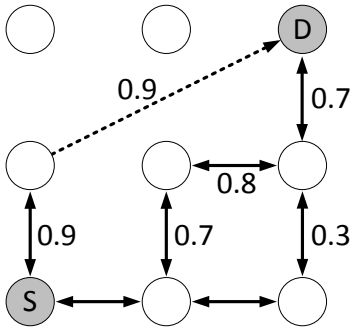


Figure 4.15: A network topology, where an asymmetric, unidirectional link of high quality (dashed line) is likely to be chosen by the routing metrics, even though its asymmetry impedes RREPs sent by the destination (D) from being returned to the source node (S). The numbers represent the LDRs of the links.

affects all of the analyzed link quality-aware metrics, the larger diversity of possible routes that are found by the BeeStack and PLF metrics augments the probability of finding a usable route with these two metrics. The Path-DR metric, however, has a lower probability of finding alternative routes, since the route diversity is much smaller, as illustrated in Fig. 4.13. This is another reason for the Path-DR metric to perform badly for medium and high traffic scales. Because of these observations, using an LQI to link cost mapping is recommended for the ZigBee path cost metric of medium granularity, which allows one both to discard low quality links and to assure sufficient path diversity, like in the PLF metric.

There are several ways to address the problem of asymmetric links. One solution is to use asymmetric routing, where the forward and backward paths are not the same. This requires two separate route discoveries. The ZigBee Pro specification includes this feature as an optional mechanism. Outside the ZigBee specification, modifications of the default AODV, like R-AODV [107], implement asymmetric routing.

4.6.3 APL

At the APL, the impact on the end-to-end network performance of the five RTO algorithms that have been introduced in Section 4.4 in terms of the overall end-to-end PDR and delay, as well as the energy efficiency of the RFDs is evaluated. Using an adequate RTO value for APL data transmissions is important for the end-to-end performance, because of two main reasons. First, the probability of producing spurious retransmissions and, thus, contributing to network congestion increases if the RTO value is set too low. Second, the probability of detecting a loss incurring a large delay increases if the RTO value is set too high.

For the low traffic scale, all RTO algorithms show a similar performance, as the number of APL retransmissions needed is small. The similarities in the performances

apply to the PDR, delay and energy efficiency metrics, as shown in Figures 4.16–4.18, respectively. In the low traffic case, most of the end-to-end data transmissions are successful, and for most of them, the average RTT lies below the initial RTO values. As a result, the impact of the RTO algorithms on the end-to-end performance is rather small for the low traffic scale. The higher delay and lower energy efficiency observed when using the CoAP algorithm is an exception, as even a few packet losses cause the average delay to increase and the energy efficiency to decrease noticeably due to the high initial RTO value used by this algorithm.

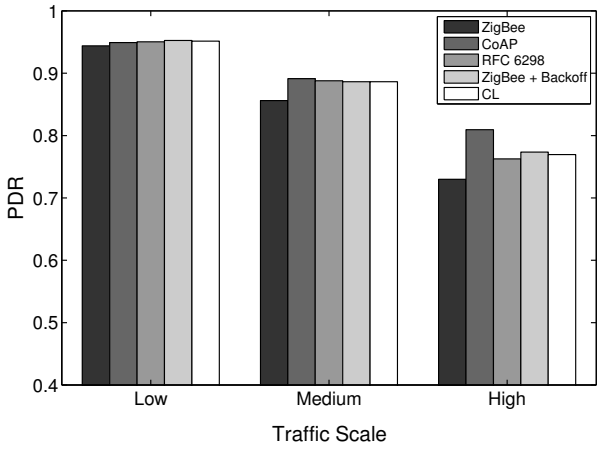


Figure 4.16: PDR results for different RTO algorithms.

In the medium and high traffic scales, end-to-end delays get larger due to congestion, and accordingly, the average RTT augments. Together with a higher packet loss ratio, the frequency of RTO expirations increases noticeably. For example, when using the default stack configuration with the ZigBee RTO algorithm, compared to the low traffic case, the number of RTO expirations is augments by 330% at medium traffic and by 813% at high traffic. As RTO expirations become more frequent, the impact of the RTO algorithms on the overall network performance becomes greater. The default algorithm is outperformed by all other algorithms in terms of PDR for medium and high traffic scales, as seen in Fig. 4.16.

With the medium traffic scale, the average delay obtained with the default ZigBee RTO algorithm lies in between the delay values observed when using the other RTO algorithms. The CoAP algorithm delivers the best PDR performance, but suffers from the highest delays. Similarly, the ZigBee RTO algorithm with BEB has a larger delay than that of the default RTO algorithm. The PDR and delays of the TCP RTO algorithm and the CL RTO algorithm are very similar, performing better than the default algorithm. The energy efficiency of the different algorithms follows a pattern that resembles qualitatively the curve observed for the end-to-end delay.

For the high traffic scale, the CoAP RTO algorithm achieves the highest PDR at the cost of the highest average delay. The other alternative algorithms achieve a similar

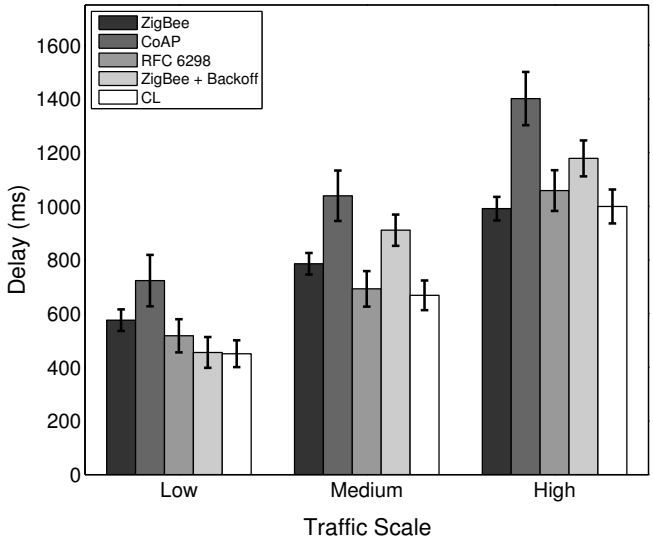


Figure 4.17: Average end-to-end delays with 95% confidence intervals for different RTO algorithms.

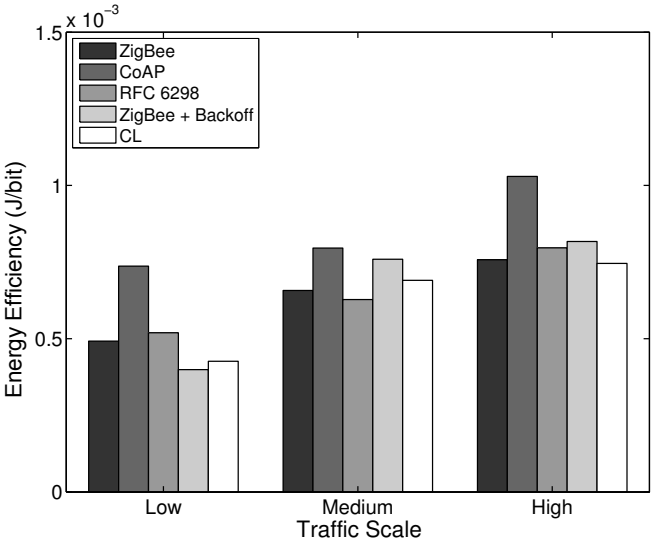


Figure 4.18: Energy efficiency for different RTO algorithms.

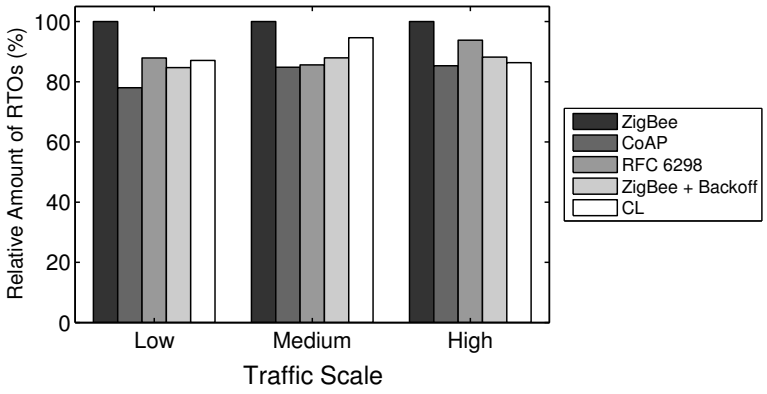


Figure 4.19: Relative amount of RTO expirations as seen from the APL, when using different RTO value calculation algorithms, taking the default ZigBee RTO method as the reference.

PDR, lying between the PDR of ZigBee and CoAP RTO algorithms. It is observed that like the CoAP algorithm, the RTO expirations of the ZigBee RTO backoff grow exponentially and lead to a higher amount of polls when packets are lost and to a higher energy consumption. The RFC 6298 RTO algorithm performs slightly better, explainable by the lower average delay. The CL RTO algorithm has a low delay and a higher PDR than the default RTO algorithm. As a consequence, the energy efficiency of this algorithm is the highest one.

When the default ZigBee RTO algorithm is used, under medium and high traffic load, the RTT frequently surpasses the 1.5 s of the initial RTO value. Since no RTO backoff is applied, two main effects are observable. If a packet really gets lost during a (re)transmission, the reaction time to the loss is constant, even after subsequent losses, contrary to the algorithms that apply a backoff. This assures a fast recovery from packet losses and helps to keep the delay low, even if the packets are lost consecutively. On the other hand, if the retransmission of the data packet is spurious and the network suffers from congestion, a quick retransmission may increase the congestion even further. This aggressive behavior comes at the cost of the lowest overall PDR and the highest amount of RTO expirations among the investigated RTO value calculation approaches (see Fig. 4.19). By adding a backoff to the default algorithm, the ZigBee RTO algorithm with BEB achieves higher PDR at the cost of higher delays for medium and high traffic scales.

When compared to the default ZigBee RTO algorithm, the CoAP RTO algorithm delivers a noticeably better performance in terms of PDR. Because of the large initial RTO value, spurious retransmissions are not likely. This positively affects the amount of RTO expirations (Fig. 4.19). The drawback of such a large initial RTO value is the large amount of time that has to pass until a packet loss is detected. Independent

of the amount of traffic, this causes the average delay and current consumption of the CoAP algorithm to be the largest, as seen in Figures 4.17 and 4.18.

The performance of the CL algorithm heavily depends on the network condition during the route discovery and the RTT-multiplier K. For $K = 4$, the CL algorithm performs on average slightly better than the default algorithm in terms of PDR for all traffics and in terms of delay with low and medium traffic. The energy efficiency of the CL algorithm on average is very similar to the one obtained by the default algorithm. However, it is important to point out that the values calculated by the CL algorithm vary strongly with the route HC. Since the RTO value is a multiple of the RTT in this case, short routes with a low end-to-end delay will have a relatively short RTO value compared to long routes with a large delay. These differences in RTO values cause the behavior of the CL algorithm to be adapted to the number of hops of a path. Overall, this algorithm provides higher PDR than the ZigBee algorithm, lower delay (except for high traffic), a similar energy efficiency and a lower amount of RTO expirations. Experiments with higher K values led to higher overall PDR, but also larger delay. Smaller values led to the opposite effect, offering lower overall PDR and a shorter delay. Choosing K to be four provided a good tradeoff in the considered scenarios.

The RFC 6298 algorithm performs slightly better than the default ZigBee algorithm at all traffic scales in terms of PDR, while the average delay is slightly smaller at low and medium traffic. At high traffic, the delay is larger for the RFC 6298 algorithm. In terms of energy efficiency, it performs slightly worse than the default algorithm in all traffic scenarios. In TCP, this algorithm is used to maintain an accurate estimation of the RTT during a transport layer connection. This normally involves the exchange of multiple packets between the endpoints of the transmission. Using this RTT information allows the sender to dynamically adapt the RTO value for subsequent transmissions to a specific destination. This means, however, that for end-to-end data transmissions with few packets, as are expected in a WHAN, the algorithm may not develop its full potential: the majority of events happen infrequently and only require a single data packet to be transmitted from the source to the destination node. On the other hand, when three packets are transmitted as a result of a periodical event (or even more packets, when adding up several events for one destination), the algorithm may actually do some RTT calculations to find an efficient RTO value. According to Karn's algorithm [51], RTT measurements may not be taken from retransmissions. While it is likely to get valid measurements at low traffic with a low amount of RTO expirations, the number of valid RTT measurements diminishes at medium and high traffic scales, where RTO expirations happen frequently. The ability to tune the RTO value to the RTT at low and medium traffic allows the RFC 6298 algorithm to achieve lower delay and higher PDR. At high traffic, the average delay changes from being lower to being higher than the one obtained by the default RTO method, due to a greater number of RTO expirations and the consequently applied BEB mechanism. To reduce the amount of RTO expirations and to increase the probability of getting valid RTT measurements at high traffic scales, the initial RTO value could be increased, as higher delays are

expected. However, a higher initial RTO value (like the one used by CoAP) would increase the delay and reduce the energy efficiency if packet losses occur on routes that have a low delay. Overall, since the amount of valid SRTT calculations is low, the TCP algorithm mostly uses the default RTO value and the BEB; therefore, it cannot clearly outperform the other algorithms.

It can be shown that from the five RTO algorithms that have been analyzed, there is not an optimal one that provides the highest PDR, the lowest delay and the best energy efficiency at the same time. However, the CL algorithm offers a good tradeoff between these three performance parameters. The implementation of this algorithm is a simple alternative to the default ZigBee algorithm; hence, the CL algorithm is considered as an interesting possibility to augment the performance of the default ZigBee stack. Overall, the influence of the RTO algorithm on the overall performance is rather small compared to the impact achieved by modifying the number of MAC layer retries or the routing metric. However, the results show that the effect of the RTO algorithm increases with the amount of traffic.

4.7 Recommended Stack Configurations and Their Comparative Performance Evaluation

In a WHAN scenario, high PDR, as well as low delay and high energy efficiency are important. Hence, a protocol stack configuration used in this kind of environment should satisfy all of these requirements. The results from the previous section have shown that the default ZigBee stack configuration can be outperformed with alternative configurations in a WHAN for all traffic scales. Therefore, its default parameters and mechanisms need to be modified. In the literature, there exist proposals for mechanisms that adapt crucial protocol stack parameters dynamically to the current network state to improve network performance. An isolation layer [108] or pTunes [109] are examples for such proposals, while they propose approaches that disrupt the ZigBee architecture or are specific to tree topologies. However, in this study, mesh topologies are considered (see Section 4.4.2), and alternative stack configurations that do not disrupt the ZigBee stack architecture are evaluated.

The first alternative configuration, called Recommended-Compliant, is compliant with the ZigBee specification, i.e., the parameter values lie in the allowed range and the protocol mechanisms are not changed. This configuration includes the following:

- The amount of allowed MAC layer retransmissions is set to seven, which is the maximum allowed by the IEEE 802.15.4 specification. Lower values have proven to deliver lower PDR, larger delay and worse energy efficiency, which is not desirable in a WHAN.

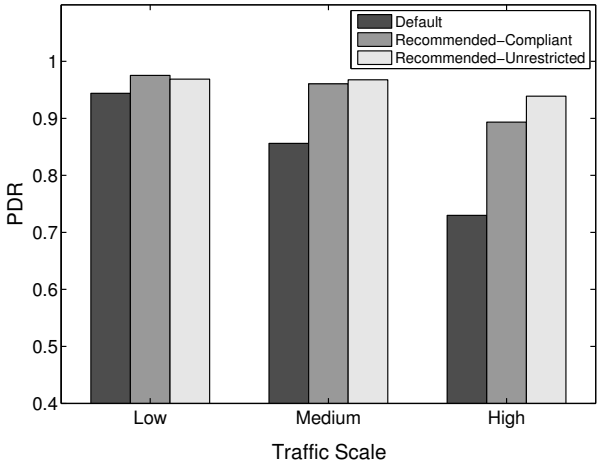


Figure 4.20: Comparison of the overall PDR between the default, Recommended-Compliant and Recommended-Unrestricted stack configurations.

- The ZigBee path cost routing metric should be capable of choosing routes that perform well under all tested traffic conditions. According to the results presented in Section 4.6.2, the PLF metric fulfills these requirements.
- The default ZigBee RTO algorithm is used.

The second recommended stack configuration, called Recommended-Unrestricted, includes the use of parameter settings and mechanisms beyond the bounds of the ZigBee specification:

- The maximum amount of MAC layer retransmissions is set to 11, since the results showed that with this value, the performance increases even more than that of using seven retries. A higher value, however, led to a decrease in the performance.
- Again, the PLF metric is applied for the ZigBee path cost metric.
- The default ZigBee RTO calculation mechanism is replaced with the CL algorithm ($K = 4$), since it delivers a good performance with a good tradeoff between PDR, delay and energy efficiency for all traffic scales.

Experiments have been carried out for the comparison of PDR, delay and energy efficiency performance of the default, Recommended-Compliant and Recommended-Unrestricted stack configurations, the results of which are shown in Figures 4.20–4.22, respectively.

When compared to the default configuration of the ZigBee stack, the results with the alternative protocol stack configurations show a noticeable improvement in PDR, delay and energy efficiency. Increasing the amount of MAC layer retransmissions augments

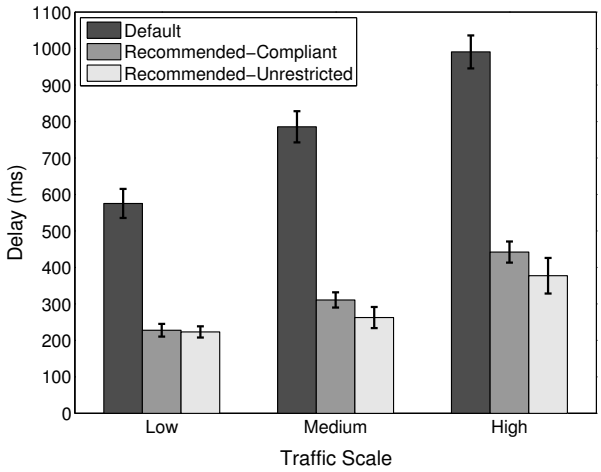


Figure 4.21: Comparison of average delays between the default, Recommended-Compliant and Recommended-Unrestricted stack configurations, including 95% confidence intervals.

the one-hop reliability, which is especially important for higher traffic scales, where losses due to collisions and interference have to be compensated for. Furthermore, insisting in the transmission over a specific link, instead of considering it broken and doing a route repair quickly, is less costly in terms of PDR, delay and energy efficiency. Additionally, the PLF metric uses a LQI to link cost mapping with sufficient granularity, which provides a good path diversity at the same time. As shown in Fig. 4.20, the PDR of the Recommended-Compliant configuration improves by 4.4%, 14.6% and 31.5% for low, medium and high traffic, respectively. At the same time, the delays decrease by 60.4%, 60.4% and 55.4%, as shown in Fig. 4.21, and the energy efficiency improves by 40.5%, 46.5% and 21.4%. These results evidence significant network performance improvements, which show that the default ZigBee stack configuration for WHANs should be revisited.

With the Recommended-Unrestricted configuration, an additional improvement in the PDR performance is observable for medium and high traffic scales. In terms of PDR, the improvement achieved by the Recommended-Unrestricted configuration in comparison with the ZigBee default one is 4.1%, 15.3% and 33.6% for low, medium and high traffic, respectively. Correspondingly, the end-to-end delays decrease by 61.2%, 66.6% and 61.9%, while the energy efficiency improves by 43.2%, 48.7% and 28%. Hence, it has been shown that by relaxing the allowed range of settings for network parameters and by allowing the cross layer feedback from the routing layer to the application layer, a further PDR improvement of up to 2.1%, a delay improvement of up to 6.5% and an energy efficiency improvement of up to 6.6% are possible. The energy efficiency of the Recommended-Unrestricted settings stays very similar to the one of the Recommended-Compliant protocol stack configuration (Fig. 4.22).

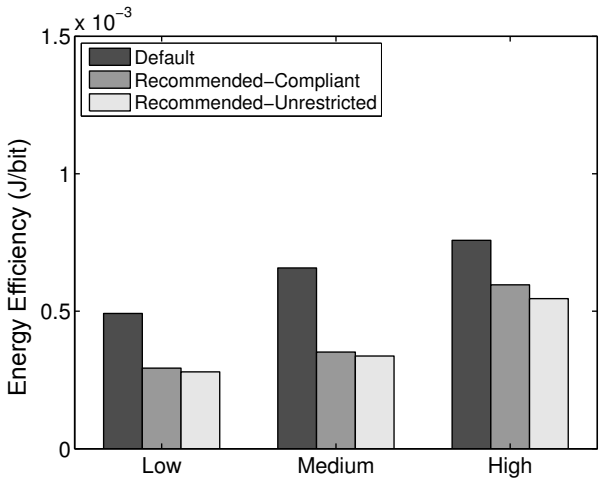


Figure 4.22: Comparison of average energy efficiency between the default, Recommended-Compliant and Recommended-Unrestricted stack configurations.

4.8 Performance Evaluation of the Recommended Stack Configurations in Alternative WHAN Scenarios

In this section, the performance of the default protocol stack configuration and the two recommended settings that have been introduced in the last section in alternative WHAN scenarios to cover a larger variety of possible network setups is evaluated. First, two alternative network topologies are introduced that differ in size and layout from the basic test scenario. Second, these evaluations are repeated in all three introduced topologies with the transmit power level 2, corresponding to a transmission power of –28.7 dBm. For this transmission power, a current consumption for the transmission state of $I_{tx} = 7.3$ mA was measured. In all topologies it is demonstrated that, independent of the transmission power level, an improvement can be achieved with the two recommended protocol stack configurations. For the validation of the recommended settings, the dumbbell topology is introduced (where nodes in the central zone are likely to suffer a greater degree of congestion than in the basic topology), as well as the square topology, where there exists a smaller node degree and, thus, a lower amount of end-to-end connectivity paths, in comparison with the basic topology. Figure 4.23 shows the layout of the dumbbell and square topologies and the assignment of roles to nodes for different traffic scales. The roles are the same as the ones defined for the basic topology in Section 4.5. The amount of traffic generated by each of the nodes in the network does not change when compared to the original topology, therefore the overall traffic generated in these alternative scenarios is less, which results in less network congestion.

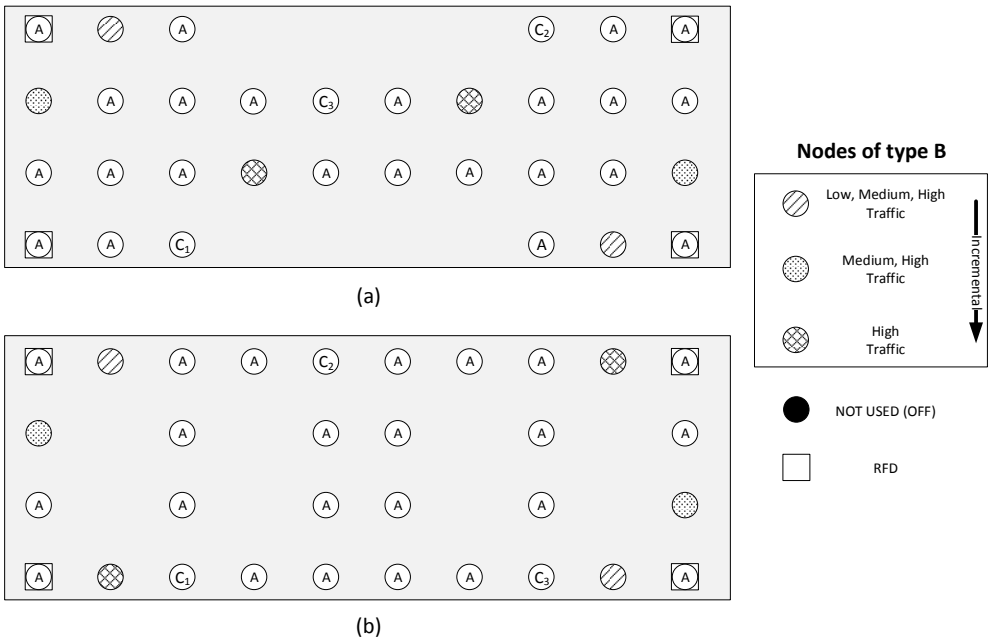


Figure 4.23: (**a**) Dumbbell and (**b**) square topologies, with roles as defined for the basic scenario.

The percentage improvement achieved by the recommended configurations over the default ZigBee stack configuration for different topologies and different traffic scales is extracted. The improvement achieved for a given topology over high traffic scale is presented in Tables 4.4 and 4.5 for a transmission power of –33.0 dBm and –28.7 dBm, respectively. The results confirm that the recommended settings are able to augment network performance without incurring any additional traffic overhead. As shown in Table 4.4, the overall PDR increases in the alternative topologies, while the delay and energy efficiency improve noticeably. The lower performance improvement achieved by using the recommended configurations in the alternative topologies can be explained by the fact that in these scenarios, the network congestion is low, since the amount of nodes that generate traffic is almost halved compared to the basic topology. This results in a higher stability of the routes, less packets drops and lower one-hop delays. With the default protocol stack configuration, a high PDR is obtained. Still, the improvements provided by the recommended protocol stack settings are able to increase the overall PDR further. Less APL retransmissions and route repairs are necessary, causing a significant decrease in the average delay and an increase in the energy efficiency.

Table 4.5 shows the improvement over the default communication protocol stack settings of the Recommended-Compliant (Compl.) and Recommended-Unrestricted (Unrestr.) configurations for the increased transmission power setting. As for the lower transmission power, the recommended configurations again improve network performance. There is a noticeable improvement of delay and energy efficiency, while the

PDR stays almost the same. It is observed that by incrementing the transmission power, the connectivity of the network changes dramatically, as the radio transmission range increases noticeably and so does the number of neighbors of the nodes. As a consequence, the average amount of hops between two nodes is much lower. Further, it is observed that the overall PDR is almost 100% for all topologies, independent of the amount of traffic. Therefore, the achievable performance improvements in terms of PDR are rather low. The improvement of delay and energy efficiency are higher, since the recommended protocol stack configurations reduce the amount of packet losses and, thus, the APL RTO expirations, which has a noticeable positive effect on the delay and the energy efficiency.

Overall, it has been shown that the recommended configurations for WHANs that have been derived in Section 4.7 achieve higher performance than the default one in a total of six different network scenarios for all considered traffic conditions.

Table 4.4: Performance improvements achieved by the recommended stack configurations over the default ZigBee stack configuration for different topologies (default transmission power, high traffic scale).

	Basic		Dumbbell		Square	
	Compl.	Unrestr.	Compl.	Unrestr.	Compl.	Unrestr.
PDR	31.5%	33.6%	2.7%	2.8%	19.2%	24.8%
Delay	-55.3%	-61.9%	-32.4%	-32%	-39%	-47.8%
Energy Efficiency	21.4%	28%	23.9%	22.8%	16.4%	19.5%

Table 4.5: Performance improvements achieved by the recommended stack configurations over the default ZigBee stack configuration for different topologies (increased transmission power, high traffic scale).

	Basic		Dumbbell		Square	
	Compl.	Unrestr.	Compl.	Unrestr.	Compl.	Unrestr.
PDR	0.8%	+1%	0.4%	0.3%	0.2%	0.1%
Delay	-47.7%	-49.1%	-24.7%	-21.8%	-27%	-30%
Energy Efficiency	37%	36.5%	12.2%	12.5%	11.1%	15.7%

4.9 Conclusions

Several WHAN scenarios with different topologies, traffic loads and node roles were defined and implemented in a real testbed to carry out experiments with a large variety of communication protocol stack configurations. The evaluation of these scenarios

has shown that the mechanisms and the configuration parameters from different layers of the ZigBee specification can have a significant effect on the overall network performance and that it is not trivial to find a configuration that works optimally in a HA environment.

At the MAC layer, the measurements that are carried out with different maxima of MAC layer retransmissions show the importance of one-hop reliability for overall network performance. The findings show that increasing this limit greatly improves the performance, especially as the traffic scale increases. Surpassing the limitations defined by ZigBee for the allowed amount of retransmissions may even lead to better results, although, eventually, a point is reached where performance saturates or even degrades due to congestion. The evaluations also showed that there is an important cross layer effect between the maximum amount of MAC layer retransmissions and the behavior of the link break detection mechanism of the NWK layer. A higher LDR, obtained by increasing the allowed number of MAC layer retransmissions, reduces the amount of route repairs, which are initiated after a data frame transmission failure. As a consequence, control message overhead and congestion decrease. Therefore, it is suggested to increase the default setting of maximum MAC layer transmissions and its allowed value range.

For the routing metric as part of the NWK layer routing mechanism, the ZigBee specification leaves the implementation of a link cost algorithm to the user. It has been shown that the metric of a default ZigBee stack, as implemented in the widely used BeeStack, or constant link costs (HC metric) do not result in the best performance. As the specification suggests, a mapping from LQI to LDR is recommendable and can lead to a noticeable improvement in the performance. The PLF metric achieves this by using a medium-level granularity mapping that provides sufficient link quality-awareness and, at the same time, a good path diversity. The latter is required to avoid overutilization of a few high quality routes, as the Path-DR metric does.

While the ZigBee specification does not foresee an alternative RTO algorithm for the end-to-end reliability mechanism, it could be shown that the overall PDR can be increased by using certain alternatives, at the cost of slightly higher delays and less energy efficiency. As a balanced alternative to the default algorithm, the cross layer algorithm presented is a simple to implement and effective solution.

The evaluations show that PDR, delay and energy efficiency can be improved significantly by altering the default stack configuration or by implementing alternative mechanisms at each layer. Two recommended stack configurations were derived from the results obtained from the evaluations of the mechanisms and parameter settings of the ZigBee layers. The Recommended-Compliant configuration of the ZigBee stack defines a higher number of MAC layer retries and uses the PLF routing metric. This configuration provided an improvement of up to 31.5% in terms of PDR, a delay drop of up to 60.4% and an improvement of the energy efficiency of up to 46.5% for specific traffic conditions compared to the default ZigBee configuration for the investigated scenarios. Finally, the Recommended-Unrestricted configuration requires the modification

of the maximum number of MAC layer retransmissions and a modification of the APL RTO algorithm. By resorting to the PLF as the routing metric, allowing up to 11 MAC layer retries and applying the cross layer RTO algorithm, a PDR improvement of up to 33.6%, a delay reduction of up to 66.6% and an improvement of energy efficiency of up to 48.7% were achieved compared to the default ZigBee configuration for the investigated scenarios. It could be shown that the two recommended stack configurations also outperform the default settings in three different network topologies for two different transmit power settings. In summary, by tuning a subset of the available ZigBee settings, an important performance improvement is achievable, and the addition of alternative settings and mechanisms may further increase it.

Based on the findings of this chapter, it is recommended that the ZigBee Alliance and the IEEE 802.15 Working Group reconsider the default configuration and/or relax the allowed range of settings for the investigated parameters and mechanisms. The findings also provide useful guidelines for the configuration and design of non-ZigBee, low-power, wireless networks, which are, however, based on IEEE 802.15.4.

In the next chapter, the work on improving the performance of end-to-end transmissions is extended to IPv6 capable communication protocols stacks, designed for the IoT.

5

Design, Evaluation, and Improvement of Congestion Control Mechanisms for IoT Communications

In Chapters 3 and 4, evaluations have been carried out to determine crucial parameters and mechanisms of different protocol stacks to improve end-to-end transmissions for networks of constrained devices. In particular, two different network scenarios have been investigated:

- A low-power wireless network using an IEEE 802.15.4-based communication protocol stack. In this scenario the reliable end-to-end transmission of data batches between two nodes was considered.
- A ZigBee WHAN scenario, where the results from the evaluations of the IEEE 802.15.4-based stack could be confirmed and extended by new findings. Multiple, parallel transmissions and varying traffic patterns between nodes, as well as sleepy devices were included in the evaluations to represent more complex WHAN scenarios.

The experimental evaluation demonstrated that in both network scenarios important performance improvements can be achieved by adjusting and modifying a set of parameters and mechanism settings from the MAC, NWK, and Transport layers. It was determined how end-to-end performance in each specific network scenario can be improved according to a variety of performance metrics, such as delay, PDR, and energy efficiency. Moreover, several general rules for the performance improvement of IEEE 802.15.4 networks were derived from the results of the analysis.

This chapter focuses on the investigation of end-to-end performance for one particular phenomenon observed during the previous evaluations that impacts on the performance of nearly all protocol stack layers in WSNs: Congestion. Congestion mainly manifests in form of packet drops that can lead to a lower PDR, increased end-to-end delays, and a higher energy consumption of the network devices, as observed in the evaluation results presented in Chapters 3 and 4.

In this chapter, the investigation of CC mechanisms is translated to IoT networks that run full IPv6 capable communication protocol stacks and are connected to the Internet. The evaluations carried out focus on CC mechanisms for CoAP, i.e., the application protocol designed by the IETF to be used for constrained devices in IoT networks. The content of this chapter was published in [BGDP13], [BGDPed], and has led to the coauthorship in [BBGD14].

5.1 Introduction

The development of IPv6 stacks for networks of constrained devices with limited hardware resources has paved the way for many new areas of applications and protocols that involve Internet communication. In order to make IEEE 802.15.4-based networks IPv6 capable, the specification of the 6LoWPAN adaptation layer plays a fundamental role (see Section 2.4).

The quantity and diversity of devices that are interconnected in the IoT are constantly increasing and have resulted in a large variety of new and appealing application scenarios. Recent studies indicate that more than 25 billion *things* are expected to be connected over the Internet by the end of 2020 [11].

A significant pillar for this development is CoAP, designed as the main application-layer protocol to be used by IoT devices for IP-based, HTTP-like interactions [110]. CoAP is a new Web protocol, designed by the IETF to enable the manipulation of resources for constrained devices that are capable of connecting to the Internet in a RESTful manner (see Section 2.7.3). Operating on top of UDP, CoAP must handle CC by itself. For this purpose, Section 4.7 of the CoAP RFC (RFC 7252) defines a basic CC mechanism for the exchange of CON messages. The algorithm follows a conservative, best effort approach to assure a safe and stable operation in WSNs and in the Internet. Yet, in spite of the high relevance of CC mechanisms for the performance of end-to-end transmissions, an evaluation of this mechanism is missing in the literature.

This chapter provides an answer to the question whether the simple CC mechanism provided by default CoAP is suited for IoT communications. To achieve this, an evaluation of the default CC mechanism used by CoAP is provided, showing that its simplicity limits its performance when facing the challenges imposed by typical IoT traffic patterns. For this reason, an advanced CC mechanism for CoAP proposed by the CoRE IETF working group is evaluated in the Cooja simulator: the *CoAP Simple Congestion Control/Advanced* (CoCoA). After providing results that show improvements of the performance when compared to the CC mechanism of default CoAP, several

shortcomings of CoCoA are derived from the observations made in the simulations. The addition of new mechanisms, and the improvement of existing ones, result in an updated proposal for advanced CC, *CoCoA+*. Evaluations of CoCoA+ show that it addresses the shortcomings of CoCoA and performs at least as well as CoAP, which is a crucial design criterion for an advanced CC mechanism for CoAP.

Section 5.2 provides an introduction to traffic patterns and CC in the IoT and in Section 5.3 default CoAP's approach to CC is presented. A state-of-the-art of proposals for CC mechanisms in the ambit of CoAP is given in Section 5.4. The initial version of the advanced CC mechanism CoCoA as defined in version 0 of the draft is explained in Section 5.5 In Section 5.6 a comparative performance evaluation of default CoAP and the initial version of CoCoA is presented. The evaluations are used to find shortcomings of CoCoA and to address them, resulting in the improved version CoCoA+, which is introduced and evaluated in Section 5.7. The changes applied to the draft based on the proposals made with CoCoA are detailed in Section 5.8. Section 5.9 concludes this chapter.

5.2 CC in the IoT

One of the main problems that has to be handled when designing a new end-to-end communication paradigm is network congestion. This phenomenon occurs when the traffic load offered to a network approaches the network capacity. Congestion mainly manifests in form of increased delays and packet losses due to collisions or high buffer occupancy, leading to an important network performance degradation which may even result in network collapse. In many traditional Internet applications, TCP provides end-to-end CC. However, CoAP operates over UDP to enable lightweight applications and must handle congestion by itself.

In IoT communications, the traffic patterns are different from the ones in conventional networks. Constrained devices often communicate periodically, for example to notify about their sensor measurements. Even when individual devices create small amounts of data, the large number of communicating devices can be a cause of network congestion. Another important IoT traffic pattern and possible reason for congestion is the generation of traffic bursts as a reaction to events, for example a large number of notifications sent after a sensor network equipped with accelerometers detects a seismic event, or sensor temperature thresholds are exceeded in various rooms in a building. These IoT traffic patterns, together with severe node and link constraints, lead to a series of challenges for the design of a CC mechanism for CoAP, which should be capable of assuring a safe network operation, while allowing an efficient use of network resources.

The core CoAP specification offers a basic CC mechanism, which is however insensitive to network conditions. In the evaluations carried out in the previous chapters it could be shown that the use of static conservative values may cause protocols to underperform in local WSNs. Similarly, default CoAP CC may significantly underperform

in IoT applications, often being too conservative or too aggressive (if the real RTT is larger than the initial RTO range), instead of adapting its behavior on the basis of network status information actually available to CoAP. The default CC mechanism mainly consists of the RTO calculations introduced in the previous chapters and a BEB and some further limitations that are explained in the following.

5.3 Default CoAP CC

The CoAP specification [15] introduces CC in several ways for confirmable messages (CONs) and non-confirmable messages (NONs). CONs require an ACK from the destination endpoint to which the message is directed to. Up to four retransmission are allowed by default. As already explained in the introduction to CoAP (Section 2.7.3), CoAP communications are based on the exchange of UDP messages between two or more endpoints. The basic CC mechanism of CoAP imposes several rather conservative restrictions on the rate of outgoing messages and on the amount of exchanges that are allowed in parallel.

For the transmission of the original message the RTO value is randomly picked from the interval [2 s, 3 s]. As in TCP, a binary exponential backoff (BEB) is applied to the RTO value for the retransmission, i.e., its value is doubled. If all retransmissions are used and no ACK was received, the message is dropped. Above that, the number of allowed outstanding interactions towards a destination endpoint by default is limited to 1. An outstanding interaction can be a CON request for which no ACK has been received yet or a NON request for which no reply has been received yet. This conservative approach is meant to ensure a safe and stable operation of the network, avoiding fast retransmissions or the start of multiple parallel interactions that could overload the network. Apart from that, the simplicity of the algorithm results in a small memory footprint.

Traffic loads that can cause such congestion are likely to happen in CoAP communications, where messages between large numbers of devices are exchanged. However, the basic CC mechanisms offered by the CoAP specification is insensitive to network conditions. Therefore, default CoAP CC may significantly underperform, often being too conservative or too aggressive, instead of adapting its behavior on the basis of network status information actually available to CoAP.

Advanced CC mechanisms for CoAP should resolve aforementioned issues, as long as these methods do assure a behavior that is safe in the Internet. They should also consider the three following basic aspects of CC mechanisms that have been identified based on the base specification of CoAP:

1. The RTO calculation for the initial transmission of a confirmable CoAP message.

2. The backoff behavior applied to the RTO before retransmission of a confirmable CoAP message.

3. The state information stored about destinations of confirmable CoAP messages.

Currently there exist two drafts with proposals for advanced CC mechanisms for CoAP. Apart from the CoCoA-draft, which is evaluated in this chapter, another Internet-Draft with several proposals for such advanced CC mechanisms entitled 'Congestion Control for the Constrained Application Protocol (CoAP)' has been submitted to the IETF by Lars Eggert. The content of this draft is introduced in the following.

5.4 Related Work

A first proposal for an advanced CC mechanisms for CoAP has been written in the Internet-Draft 'Congestion Control for the Constrained Application Protocol (CoAP)' [111] by Lars Eggert. Several mechanisms to extend CoAP's CC mechanisms are proposed in this document with the intention to make the community experiment with these proposals. In the following the main ideas that are raised in the draft are gathered.

The author states that if the response times for CoAP messages exceed the default range of RTO values (2 s to 3 s) the protocol may retransmit unnecessary messages that are still on the path between the two endpoints or being processed at the destination. This leads to the first addition for CoAP: it should include an RTO estimator that operates similarly to the one defined in RFC 2988 for TCP to adapt the RTOs to the real durations of transactions. As an alternative to this RTO estimator, a larger initial RTO is proposed.

In addition to the RTO estimator, a solution for aggregated traffic control is proposed. The draft points out that this is a required mechanism if several transactions to different destinations are initiated in parallel. It addresses the limitation introduced by NSTART, that limits the amount of open transactions to a destination endpoint to one but does not limit the number of destination endpoints to which open transactions are allowed. A regulation mechanism proposed by Eggert is based on keeping a global pool of transaction tokens and applying a windowing algorithm to determine how many transactions are allowed in parallel. For each exchange one of the available tokens is used. When no more tokens are available, no further transactions may be initiated to any destination endpoint. The successful completion of transactions returns tokens to the pool. The subsequent successful completion of transactions during a certain time interval leads to an increase of available tokens. On the other hand, failed transmissions lead to a reduction of the token pool size.

The proposal of aggregated traffic control is extended by the use of ECN at the IP layer. When congestion is detected by a router, it may inform end nodes to reduce the outgoing message rate, which in this case would be achieved by reducing the window size.

The last addition to the set of mechanisms to provide an advanced CC for CoAP is the regulation of multicast message retransmissions. Since CoAP does not apply

retransmissions to multicast messages, the retransmission of multicast messages is left to the user application. However, CoAP does not prevent the user application from retransmitting multicast messages. Multicast messages can generate a flood of replies from all destination endpoints to which the multicast message is directed to. Hence, the subsequent transmission of a multicast message and the replies it generates may cause congestion collapse. The author states that his proposal for the aggregate CC already helps to avoid an abusive use of multicast retransmissions. Still, he incites the reader to find and experiment with additional means to regulate these retransmissions.

The draft does not provide any details about the parameter settings used for RTO calculations or a specific proposal for the behavior of the windowing algorithm for the transaction tokens. Yet, the different proposals try to address several of the limitations of the default CC mechanisms. In the following, the proposals made in the draft are analyzed and it is discussed if they are apt for the design of an advanced CC mechanism for CoAP.

As shown in the evaluations for the ZigBee WHAN scenario in Chapter 4, the estimation of the expected transaction duration for an endpoint using a RTO estimator can lead to an increase of the performance when compared to the static approach followed by the CoAP base specification. The use of an RTO estimator for CoAP is a valuable addition to CC if the expected transactions durations are either longer or shorter than determined by the default RTO value of 2 s to 3 s. If the duration of transactions tends to vary over time, an adaptive RTO algorithm can improve performance too. However, the performance of a TCP estimator in WSN communications is unclear. The premiss for updating and adjusting the RTO estimator is the measurement of RTT values. According to RFC 6298 [52], RTT values can only be calculated from messages that have not been retransmitted. Since the natural packet loss rate in constrained networks is higher than in non-constrained networks, the chances of updating the RTO estimators are expected to be significantly fewer when used in CoAP. If on top of the natural losses caused by the channel qualities, losses are added as a result of network congestion, the probabilities of measuring valid RTTs is further reduced. This signifies a pending issue that is not addressed in the draft.

The aggregated traffic mechanism intends to manage the amount of open transactions towards endpoints using a pool of tokens. An upper limit of open transactions is controlled by requiring each new transaction to use a token from the pool. Yet, this particular proposal for a global pool of transaction tokens leaves important questions unanswered.

For example, the draft does not point out how in the context of a global pool of tokens NSTART is adjusted for each destination endpoint, depending on the currently available tokens. NSTART defines how many transactions can be open in parallel towards one destination endpoint. The windowing algorithms described in Eggert's draft suggests that NSTART is dynamic and has the same value for every destination endpoint. However, different destination endpoints may show a different behavior in terms of losses and delays and using the same NSTART value for all destination endpoints

may not deliver an optimal performance. On the other hand, after several failed transactions the number of available tokens in the pool could be reduced to a value that is smaller than the number of destination endpoints that need to be served. This stands in conflict with the CoAP base specification that allows at least one transaction to any destination endpoint. Thus, the overarching aggregated traffic control conflicts with the CoAP base specification and would require a change. For future versions of the draft, the aggregated traffic control requires a revision and needs to be evaluated against solutions that apply a per destination CC, instead of the proposed global CC for all endpoints at once.

Considering that the aggregated traffic control as specified in the draft is not a viable solution for an advanced CC mechanism, the addition of ECN to the traffic control mechanism and multicast CC is not feasible as long as the aforementioned issues are not resolved. Moreover, ECN would require the routers that are involved in CoAP communications to support ECN, which in networks of constrained devices is not a prevalent feature.

Overall, the document can be interpreted as a collection of ideas, leaving room for changes and improvements of the proposed features. If the proposals really do provide an efficient CC is not clarified and if they are implementable and actually work is not demonstrated.

The ideas behind the draft written by Lars Eggert and the consideration of several fundamental guidelines for UDP traffic control [112] served as base for the initial version of the CoCoA draft written by Carsten Bormann. The CoCoA draft also defines several advanced CC mechanisms for CoAP, addressing issues arising for CoAP IoT communications. In contrast to the draft written by Lars Eggert, the CoCoA draft gives enough details about the CC mechanisms to implement them. In the following, the detailed content of the initial version of the CoCoA draft for advanced CC mechanisms for CoAP is presented.

5.5 CoCoA: An Advanced CC Proposal for IoT Communications

This Section introduces the initial version of CoCoA, an advanced CC proposal for CoAP [BBGD14].

CoCoA has been designed to address some of the issues pointed out by Lars Eggert in his draft [111] and leans some of its content on the guidelines for UDP traffic [112]. In the initial version of the CoCoA draft four basic CC mechanisms are identified: the RTO calculation algorithm, the retransmission backoff mechanism, the aging mechanism and the management of parallel transmissions. Each of these aspects is detailed in the following.

5.5.1 RTO Algorithm

The main difference in the behavior of CoCoA compared to default CoAP is the use of RTT measurements to calculate the RTO of the first transmission of a CoAP message. The main behavior of CoCoA is consistent with the behavior of the TCP RTO estimator as described in RFC 6298 [52]: a source node of reliable CoAP messages uses ACKs to obtain RTT values. Based on these RTT measurements, RTO values are calculated. Since delays for CoAP messages can be large due to application processing at the destination node, the initial RTO value is increased from 1 (TCP) to 2 seconds in CoCoA.

The use of an RTO estimator like the one defined in RFC 6298 has already been proposed by Lars Eggert in his draft [111]. However, the algorithm used to update the RTO values is not considering the relatively high amount of natural losses in networks of constrained devices and possible large RTTs due to processing delay. These occurrences may degrade the accuracy and usefulness of the RTO estimator, as it is not capable of obtaining valid RTT measurements. CoCoA addresses this issue by including a feature that allows to use RTT measurements from retransmissions.

According to Karn's algorithm [52], an RTT measurement is considered to be valid if the corresponding message was not retransmitted. However, in constrained networks packet losses due to congestion or lossy links are likely. This increases the probability of message retransmissions, resulting in a lower probability of obtaining valid RTT measurements. CoCoA therefore runs two RTO estimators in parallel for each destination endpoint: a strong and a weak RTO estimator that are updated when measuring strong RTTs (RTT_{strong}) or weak RTTs (RTT_{weak}), respectively. RTT_{strong} is calculated using the strong RTTs, which are measured when an ACK is received after the first transmission of a confirmable CoAP message. On the other hand, RTT_{weak} is calculated using the weak RTTs measured when an ACK is received after at least one retransmission of a confirmable CoAP message. Due to the ambiguity of not knowing to which of the transmissions an ACK belongs to, a weak RTT is always set to be the duration between the first transmission of the CoAP message and the time of the reception of an ACK. The purpose of using a weak RTO estimator in despite of this ambiguity is to be able to benefit from all RTT measurements.

With the following formulas RTT_{strong} and RTT_{weak} are calculated when a new RTT measurement RTT_{X_new} is obtained:

$$\mathrm{RTTVAR}_X = (1-\beta) \times \mathrm{RTTVAR}_X + \beta \times |RTT_X - RTT_{X\_new}| \tag{5.1}$$

$$RTT_X = (1-\alpha) \times RTT_X + \alpha \times RTT_{X\_new}, \tag{5.2}$$

where X stands for *strong* or *weak* accordingly and with $\alpha = \frac{1}{4}$ and $\beta = \frac{1}{8}$.
The measurement of a RTT_X is used to update RTO_X as

$$\mathrm{RTO}_X = RTT_X + K_X \times \mathrm{RTTVAR}_X, \tag{5.3}$$

with $K_X = 4$.

According to the CoCoA draft, when RTO_{strong} or RTO_{weak} is updated after getting a RTT measurement, an overall RTO ($\text{RTO}_{overall}$) is recalculated:

$$\text{RTO}_{overall} = 0.5 \times \text{RTO}_X + 0.5 \times \text{RTO}_{overall} \tag{5.4}$$

The newly calculated $\text{RTO}_{overall}$ is used as the next initial RTO for a confirmable message to the same destination endpoint.

As in default CoAP in the base specification, dithering is applied to the initial RTO of a CoAP message in CoCoA: the initial RTO (RTO_{init}) for the first transmission is randomly chosen from the interval $[\text{RTO}_{overall}, \text{RTO}_{overall} \times 1.5]$.

5.5.2 Aging Mechanism

According to the CoCoA draft, if one of the RTO estimators has a value of less than 1 second and it is left without further update for more than 16 times its current value, its RTO value should be increased. In the draft, several proposals are made on how to increase the value, amongst others doubling it or setting it to $\frac{1}{8}$ of the time passed since the last update. For simplicity, in the upcoming evaluations of CoCoA, the third proposed option is chosen: the RTO estimator is always updated to a value of 1 second.

5.5.3 Backoff Mechanism

According to the initial version of the CoCoA draft, the BEB retransmission backoff mechanism is applied to retransmission of CON requests, as it is done in default CoAP. If this simple mechanism satisfies the needs of CoAP communications is to be shown.

5.5.4 NSTART Setting

Another important CoAP parameter that is discussed in CoCoA is the NSTART parameter, which determines how many CoAP transactions may be created to one destination endpoint in parallel. The base CoAP document states that this value must be set to 1. Higher values can lead to congestion of the network, since it allows various CoAP transactions between two nodes simultaneously.

However, it is assumed that a value of NSTART > 1 is applicable without the risk of losing performance and endangering the safe operation of CoAP when CoCoA's CC mechanism is used. The greater allowed amount of traffic generated by several parallel transmissions when setting NSTART > 1 is supposed to be regulated by adapting the RTO values adequately.

5.6 Analysis of CoCoA (Version 0)

In its initial version, CoCoA makes several proposals for CC mechanisms and parameter settings, as detailed in the previous section. However, the proposed features and in particular the parameter settings need to be tested, evaluated, and improved, which is done as part of the investigation carried out in this thesis.

In order to analyze the performance of default CoAP and CoCoA and with the goal to propose improvements for CoCoA, a series of network simulations are carried out. In these simulations, CoCoA is compared to the default CoAP CC mechanism and an alternative version of CoCoA that only uses the strong RTO estimator (and thus resembles the TCP RTO algorithm). This alternative version is referred to as CoCoA-Strong (CoCoA-S). Evaluating CoCoA-S serves the purpose of determining if the weak RTO estimator introduced in CoCoA has a relevant impact on the network performance or if is negligible.

To carry out the performance evaluations, the CC mechanisms of CoCoA and CoCoA-S are implemented for Contiki, so they can be used by applications that communicate over CoAP. The programs written for real nodes are uploaded to virtual nodes in Cooja in order to carry network simulations. The performance of CoAP, CoCoA, and CoCoA-S are evaluated in different network topologies, traffic loads, and configuration parameters. Results show that CoCoA and CoCoA-S are capable of improving performance in congested network scenarios.

Sections 5.6.1 and 5.6.2 explain the simulation setup and the nature of the simulated networks, respectively. The configuration of the simulation test runs is detailed in Section 5.6.3, followed by the results of the evaluations in Section 5.6.4.

5.6.1 Simulation Setup

This section gives an overview of the simulation environment used in this study. Also, this section covers the configuration of the nodes and explains the tested network topologies, as well as the test run configuration used to do the performance evaluations.

Cooja provides the necessary software to emulate the hardware of a set of real sensor nodes (see Section 2.9.4 for details). This means that real hardware limitations like memory and processing capacities are respected in the simulation environment. From the types of devices that are supported by Cooja, two IEEE 802.15.4 capable motes with the same type of radio transceiver (CC2420 [21]) are chosen for the simulations: the Z1 from Zolertia [113] and the Tmote Sky from Moteiv [114], the latter being equivalent to TelosB motes [74] used for the evaluation of the ZigBee WHAN scenario in terms of hardware. The most relevant features of the two motes are stated in Table 5.1. As seen in the table, the motes differ in ROM and RAM capabilities and the MCU. While Z1s offer more ROM for application code, the Sky Motes have higher RAM storage capacity.

Table 5.1: Hardware specifications of the Zolertia Z1 and Moteiv Tmote Sky wireless sensor nodes.

	Tmote Sky	Z1
RAM	10 KB	8 KB
ROM	48 KB	92 KB
MCU	MSP430F1611	MSP430F2617
Radio	CC2420	

The simulated network topologies used for CoAP performance evaluation include two types of nodes that differ in their resource requirements: CoAP nodes and RPL border routers. Each topology includes one RPL border router that implements the communication stack introduced in Section 2.9.3 up to UDP and that acts as RPL-root node. The simulated network topologies are local networks that do not require the border router to establish external connections.

According to the RPL specifications, when a node is missing a routing table entry for the destination node, it forwards the packet to the next node in the direction of the RPL-root. This process is repeated, until a node has a valid routing entry to forward the packet in the direction of the requested destination address. In the case that none of the nodes on the way to the RPL-root has valid routing information stored, the RPL-root eventually receives the packet. The RPL-root then in the optimal case looks up the entry with the destination address for the packet from the routing table and forwards it to the next hop in the direction of the destination node. To achieve that, the RPL border router needs to be able to store routing table entries for all possible destinations in the network, which is why the RAM requirement of this node is relatively high. Because of that, the RPL border router runs on Tmote Sky nodes that have a higher RAM capacity compared to the Z1 motes. Since the analyzed network topologies are of limited size, it is possible to assign sufficient routing table sizes to the RPL-root beforehand for each topology.

The rest of the motes in the network topologies are nodes that run the full Contiki stack and the Er-CoAP implementation based on CoAP version 13. In contrast to the RPL-root, the nodes with CoAP require more ROM to fit the additional application code that is part of the Er-CoAP implementation and also for the user application that controls the high level behavior of the node. Z1 nodes are better suited for this type of tasks, since they have higher ROM capacities when compared to Tmote Sky nodes.

The two types of motes are part of all network topologies that are used for simulation experiments in this chapter. In each of the network topologies, there is always a single RPL-root, while the rest are CoAP nodes, out of which one or two are set as the destination (sink) nodes as detailed in Section 5.6.2. The RPL border router is responsible for initiating the DODAG (i.e., the network structure created and maintained by RPL) and storing the collected routing information for all nodes that are part of the

DODAG [26]. For the simulations, the RPL border router is defined to serve only as relay for CoAP messages: it does not create CoAP messages and is not the destination endpoint of CoAP messages. Contiki implements only the storing mode for RPL [26]. Since it requires a certain amount of time for RPL to set up the DODAG across the whole network, each simulation starts with a RPL initialization phase of 60 s. During this phase, the network does do not generate any traffic apart from RPL control messages.

To observe the performance of the CoAP CC algorithms, a RPL malfunctioning is not desirable since it distorts the results. The network topology determines the configuration for some of the RPL parameters, that, when not configured adequately, may lead to performance drops. The routing table and neighbor table sizes are two important parameters that determine if packets can be routed correctly throughout the network. For the simulations carried out in this chapter, the neighbor tables are able to store information about all direct neighbors. The RPL border router has a routing table large enough to store entries for all nodes of the network. Due to the strict RAM limitations of CoAP nodes, their routing tables need to be smaller. Adjusted to the introduced topologies, the storable amount of RPL neighbor and routing table entries is set to the values shown in Table 5.2.

Table 5.2: Configurations of the RPL routing and neighbor table sizes.

	Chain Topology	Dumbbell Topology	Grid Topology
Neighbor Table (all nodes)	2	10	4
Routing Table (CoAP node)	10	10	20
Routing Table (RPL bord. router)	20	20	48

Another factor that influences the performance of the network essentially is the size of Contiki's MAC layer queue and the RDC mode. All packets that are generated or forwarded by a node need to traverse a queue located at Contiki's MAC layer. The queue is used to store packets that are pending to be sent and that need to wait the CSMA procedure before being transmitted by the radio. Since the amount of RAM for constrained devices is very restricted, the number of MAC layer queue entries is set to 4 for CoAP nodes and 6 for the RPL border router. Contiki disposes of different RDC mechanisms, such as NullRDC, B-MAC, X-MAC and ContikiMAC. For the evaluations carried out in this chapter, the NullRDC is chosen. In NullRDC, nodes do not sleep and their radio is constantly turned on. For the performance analysis of CoAP, the

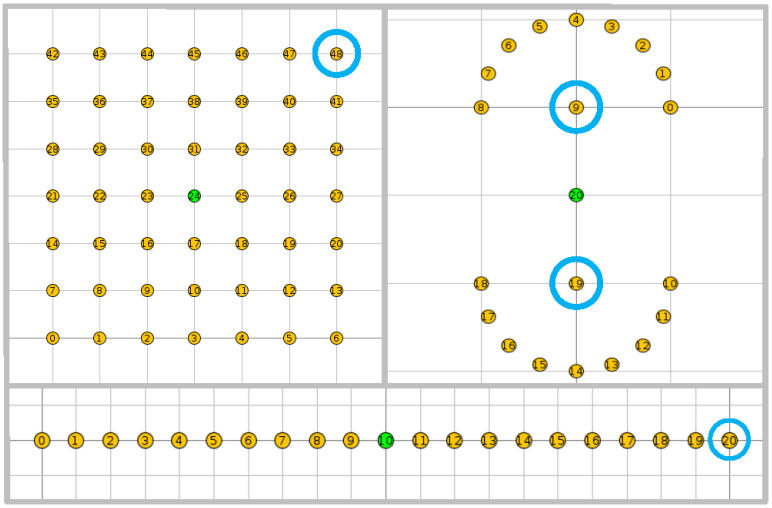


Figure 5.1: The three network topologies used for performance analysis (grid, dumbbell, and chain). The width of a square corresponds to 10 m. Nodes with a circle are sink nodes for CoAP messages. The nodes in the center of the topologies are RPL border routers.

use of any RDC alternative is discarded, thus avoiding the cross-layer interactions they require.

5.6.2 Network Topologies

To carry out the performance evaluation of CoAP CC mechanisms, three different network topologies are defined for the simulations. They differ in the number and positioning of the nodes, which amongst other things lead to differences in the number of direct neighbors of the nodes, the lengths of routes between source and destination of CoAP requests, and the number of nodes that may compete for the radio channel simultaneously. The evaluated topologies are i) a chain of nodes with 21 nodes, ii) a rectangular grid of 49 nodes (7x7), and iii) a dumbbell topology with 21 nodes.

Figure 5.1 depicts snapshots of the Cooja simulator GUI showing the two-dimensional positioning of the nodes for the three different topologies. The transmission range is set to 10 meters, whereas the interference range is set to be twice of the transmission range, i.e., 20 meters. The distance between nodes in the chain topology is chosen in such a way that only direct neighbors are in transmission range. Since the interference range is twice the transmission range, a packet transmission may interfere the radio of nodes up to two hops away. Each node may therefore have up to two neighbors. The same distance rule applies for the grid topology, where each node may have up to 4 direct neighbors. In the dumbbell topology, the nodes on the half circles are exactly 10 meters away from the node that is part of the dumbbell's axis.

Since the RPL border router acts as dispatcher for messages that need to be forwarded through the RPL root, it should be located in a central position of the network

to make it equally accessible from all nodes in the network. In the chain topology, the RPL border router is located in the middle of the chain. In the grid it is located at the center of the topology, where the 4th line of nodes crosses with the 4th column. In the dumbbell topology the RPL border router is the center node that connects the two laterals of the dumbbell formation. In all the topologies, CoAP messages are created by the CoAP nodes and directed to one or two sink nodes. Except for the dumbbell topology, each network has a single sink, marked with a circle in the topology overviews. In the dumbbell topology there are two sink nodes. They are the destinations of traffic that is generated by nodes from the opposite side of the dumbbell topology. While in the chain and grid topologies the sink nodes do not generate any CoAP requests, in the dumbbell topology each of the sink nodes also creates CoAP requests for the sink node on the opposite side.

5.6.3 Test Run Configuration

When a test run is started, the simulated nodes are initialized and the RPL-root creates the RPL-DODAG by spreading DIOs throughout the network. CoAP performance tests begin after the complete RPL-DAG is built. After this, the nodes start generating periodical CoAP requests. The generation of periodic traffic is implemented by running cyclic timers that upon their depletion create a new CoAP request. The CoAP request is a POST message that sets the color of a LED at the sink node and increases an internal counter of the sink node by one. Additionally an increasing message ID and the short address (node ID) of the originator node are included in the payload of the CoAP request. The sink node that receives this CoAP request carries out the requested actions and responds to the originator node with a CON message. A CoAP request message including all headers and the payload has a size of 95 bytes.

If a node generates a CoAP request, but the limitation of parallel transaction to the destination node dictated by NSTART does not permit to generate another request, the generated packet is dropped. Since all CoAP requests in the analyzed network topologies only have one possible destination, packets at the application layer are only dropped when a node is waiting for a confirmation of the previously sent CoAP request in the case of NSTART=1. If NSTART is larger, packets at the application layer are only dropped, if already NSTART transactions are active. In real life scenarios, where a single sink node is employed, NSTART can play an important role for possible congestion of the network. Thus, the performance of CC mechanisms for CoAP with a higher NSTART value is analyzed.

The duration of the simulation phase during which CoAP requests are generated is always 360 seconds. After this time, the test script generates a signal for all nodes in the network that tells them to stop spawning further CoAP requests. Subsequently the simulation terminates. All events that occur during the simulation and are of interest for the performance evaluation are collected from the nodes over their virtual serial interface. During the simulation, the nodes can send messages over their serial port

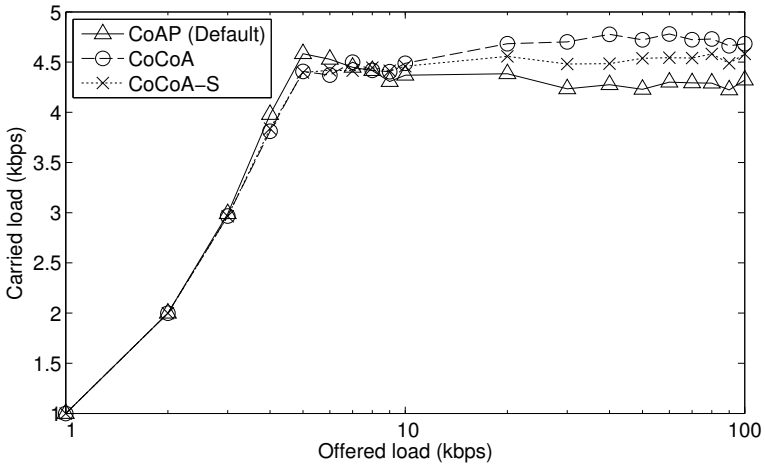


Figure 5.2: Throughput achieved for different offered loads in the dumbbell topology with NSTART=1.

that can be intercepted by the simulation script and can then be stored into log-files for further evaluation. Such messages can contain the status of internal variables, such as buffer sizes or can be used to notify about an event, such as the reception of a confirmation message.

5.6.4 Simulation Results

This section presents the results of the experimental evaluation of the CC mechanisms in reliable CoAP communication for the three previously introduced network topologies. As the performance metric, the relation of carried load against the offered load is chosen. The offered load refers to the total amount of data created by the nodes per second on average when CoAP requests are generated, and is given in kilobits per second (kbps). The carried load indicates the average amount of data that is successfully delivered to the sink node(s) per second, and also given in kbps.

For all analyzed configurations of the CoAP CC mechanisms, the test runs have been repeated three times with different random simulator seeds to obtain meaningful average results.

5.6.4.1 CC Performance for Different Traffic Loads

When NSTART is set to 1, the main difference of the analyzed CC algorithms lies in the RTO calculation algorithm. The results show that this mechanism has a high impact on the network performance and that it influences the effectiveness of the CoAP CC.

Figure 5.2 illustrates how the carried load evolves as the offered load increases when using the three RTO algorithms for the dumbbell topology. Since at low traffic rates, the network does not reach a state of congestion and nearly all packet transmissions

across the network are successful, the carried load is identical to the offered load, independently of the RTO algorithm used. As the offered load increases, the ratio of successfully delivered CoAP requests decreases because of congestion. This is where differences in the performance of the different CC algorithms become visible.

The performance of the three CC algorithms is almost identical up to an offered load of 3 kbps. For offered traffic loads up to 7 kbps, CoCoA and CoCoA-S show a slightly worse performance in terms of throughput. Since the number of hops between source and destination routes is small and there is almost no congestion in form of packet collisions or packet drops, the CoCoA algorithms have small RTO estimations. As the estimated RTO gets smaller and reaches the real RTT, the probability for spurious retransmission increases and sometimes causes the nodes to unnecessarily retransmit a packet they assumed to have been lost. These spurious retransmissions can lead to drops in the throughput performance. The default CoAP RTO does not suffer from these effects, as the initial RTO for transactions always is 2 s or higher. As the offered traffic load increases beyond 7 kbps, the CC mechanisms of CoCoA and CoCoA-S take effect, outperforming default CoAP. CoCoA-S has a slightly lower performance than CoCoA. At high traffic rates that lead to increased packet drop rates across the network, CoCoA-S has a lower probability to obtain valid RTT measurements, thus it is not able to calculate improved RTO estimations.

The relevance of RTO estimations becomes clearer when looking at the amount of packets that are dropped during the simulation. Packet drops occur when i) the MAC layer buffer overflows, or ii) when CoAP refuses to send a packet, because the number of parallel transactions allowed for a destination, i.e., NSTART parallel transactions, has been reached for the destination of the packet.

MAC layer buffer overflows are a clear sign of congestion, as they indicate that a high amount of packets are created and intended to be forwarded throughout the network, exceeding the capabilities of the network. The effect of the RTO algorithm on the amount of dropped packets is deducted for the dumbbell topology and Fig. 5.3 shows the overall amount of dropped MAC layer packets for this topology.

As seen in the figure, when applying the default RTO algorithm, the number of MAC layer packet drops grows quickly above 3 kbps of offered traffic load and reaches an asymptotic value at around 20 kbps. When using the alternative RTO algorithms for CoAP CC, a slower increment of the number of dropped packets and a much lower asymptotic value are observed. This means that the CoCoA algorithms are capable of effectively decreasing the congestion in the network. The clear difference in the number of dropped MAC layer packets between CoCoA and CoCoA-S is a result of the different RTO estimations they apply. In contrast to CoCoA-S, CoCoA additionally uses weak RTT measurements from retransmissions that may lead to large RTOs. This throttles the output of packets and leads to a further reduction of buffer overflows in the network.

An asymptotic behavior of the amount of dropped MAC layer packets is observable, because the actual amount of traffic transmitted over the radio channel does not increase significantly further with the offered load at high traffic rates. On the other hand, due

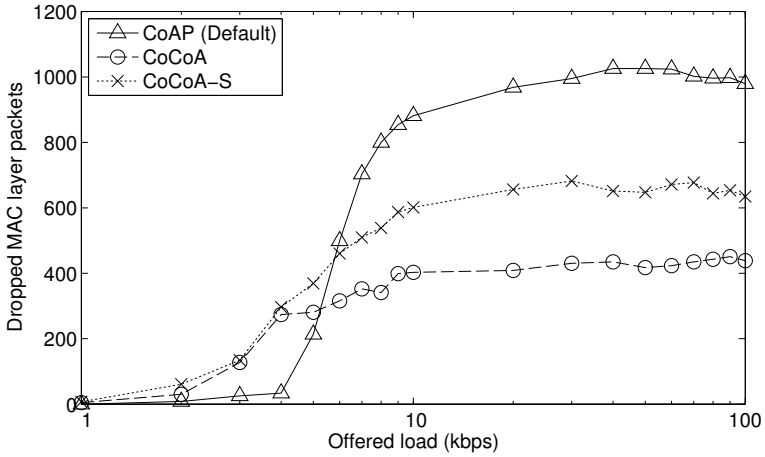


Figure 5.3: Dropped MAC layer packets for different offered loads in the dumbbell topology with NSTART=1.

to the limitation given by NSTART=1 and increasing packet generation frequencies at each node, the probability of dropping newly generated CoAP messages gets higher with the offered load. For offered load beyond the threshold where the amount of dropped MAC layer packets reaches its asymptotic value, the CoAP packet drop rate is observed to increase almost linearly with the amount of generated packets.

As mentioned before, the amount of hops between source and sink nodes in the dumbbell topology is low, resulting in small RTTs. How the CC algorithms perform when larger RTTs are observed can be demonstrated in the chain topology, where packets may have to travel along many hops before reaching their destination. The radio links along the chain are not utilized equally. The links closer to the sink will be required more frequently for transmissions, as all the packets from the other end of the chain need to traverse them. On the other hand, due to spatial reuse, it is possible for several nodes to transmit in parallel along the chain, without interfering each other.

For this particular setup, the difference between the default CoAP and the CoCoA mechanisms becomes larger, as can be seen in Fig. 5.4. This shows that the CoCoA mechanisms are able to adapt to different RTT values and to the traffic characteristics of the network. This observation can be backed up by observing the amount of dropped MAC layer packets (Fig. 5.5), showing a similar behavior as in the dumbbell topology, where the drop rates for the CoCoA mechanisms are much smaller than for default CoAP.

On the other hand, in the grid topology, the number of hops between source and destination nodes varies a lot, since many combinations of links for the connection of source and sink nodes are possible. Therefore, RTTs of different scales can be observed. Since within the transmission, and also the interference range of a node there will be

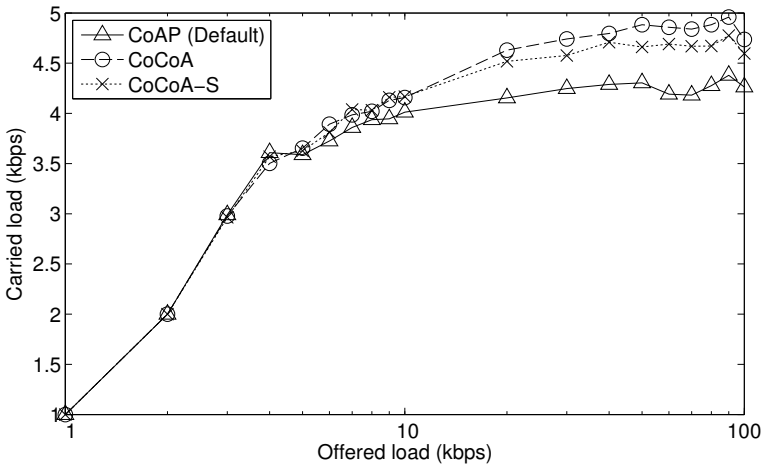


Figure 5.4: Throughput achieved for different offered loads in the chain topology with NSTART=1.

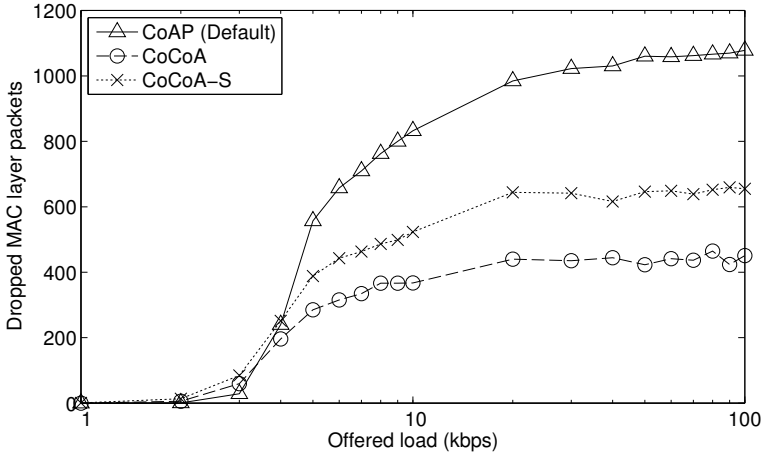


Figure 5.5: Dropped MAC layer packets for different offered loads in the chain topology with NSTART=1.

more nodes, a congestion of the radio channel is more likely than in the other analyzed topologies.

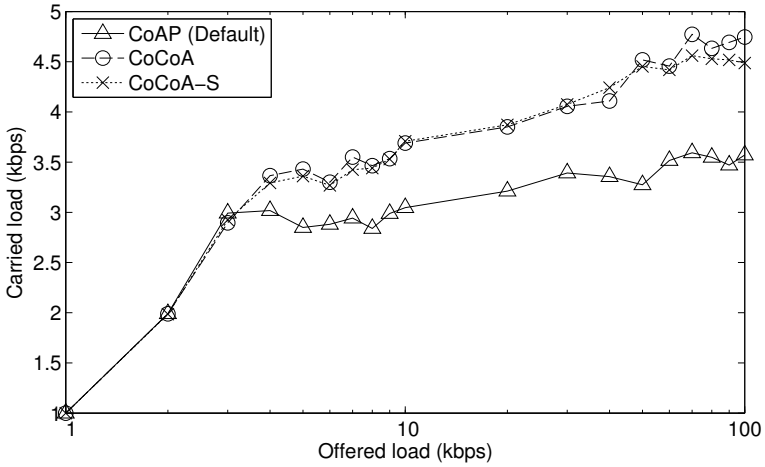


Figure 5.6: Throughput achieved for different offered loads in the grid topology with NSTART=1.

The performance comparison for the grid topology reveals that CoCoA mechanisms are able to perform significantly better than default CoAP. Figure 5.6 shows that CoCoA and CoCoA-S perform better than the default CoAP, even when the offered load is small (starting at 4 kbps). However, the performance achieved by the two advanced CC mechanisms is very similar for this topology. For the intermediate traffic rates, CoCoA gets many weak RTT measurements from retransmissions, increasing the estimated RTO to large values. In case of a loss this results in a long idle period. CoCoA-S and default CoAP use shorter RTOs. The amount of valid RTT measurements gets lower with higher traffic for nodes that apply CoCoA-S and that are not very close to the sink. Thus the CoCoA-S RTO estimations do not grow as much as CoCoA RTO estimations, while default CoAP maintains the default initial interval. Due to these lower RTO estimations, CoCoA-S will use more retransmissions in a fixed time interval for nodes that have difficulties to successfully complete transactions on their first transmission, when compared to CoCoA. This can increase the PDR, until the traffic rates become higher and this behavior is the cause of more congestion. This is the case above 60 kbps, where the performance of CoCoA-S is lower than the performance of CoCoA. The results for the MAC layer packet drops are very similar to the results for the chain and dumbbell topology and will not be detailed further. As before, CoCoA and CoCoA-S lead to a much lower drop rate of MAC layer packets.

The applicability of the observations to several network topologies confirms that the improvements obtained by using CoCoA and CoCoA-S are crucial for reliable CoAP communication. In the following, this conclusion is strengthened by observing the

performance of the CC mechanisms when allowing multiple parallel CoAP transactions to a destination node (NSTART=4).

5.6.4.2 Effect of NSTART on CC Performance

When NSTART is set to 4, the amount of maximum parallel transactions to one destination is increased from 1 to 4. Nodes that apply the basic CoAP CC initialize independent RTO timers for each transaction. Nodes that create transactions when using CoCoA and CoCoA-S can resort to already obtained RTO information for a destination node.

With a higher NSTART value, the potential degree of congestion in the network is higher, increasing the importance of an efficient CC mechanism.

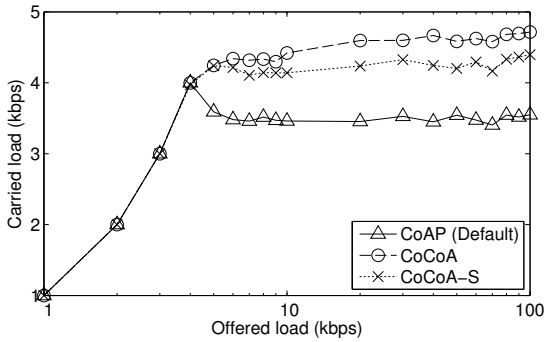
Figure 5.7a illustrates the carried load in the dumbbell topology. As seen in the figure, the default CC mechanism is not able to cope with NSTART=4. When using the default CoAP CC mechanism the performance drops noticeably compared to the performance achieved with NSTART=1. For default CoAP, offered load above 4 kbps leads to a drop and the carried load starts fluctuating around a value of 3.5 kbps. Analogical to the NSTART=1 scenario, CoCoA and CoCoA-S are able to surpass the performance of the default CoAP CC. However, the differences in the performance are greater for NSTART=4. When analyzing the amount of dropped MAC layer packets (Fig. 5.7b), it becomes clear that the default CoAP is not able to control several parallel transactions efficiently. The number of MAC layer packet drops is 3 to 4 times higher, compared to the number of drops observed for CoCoA and CoCoA-S. This means that the advanced CC mechanisms are able to better detect congestion and reduce it.

Figs. 5.7c and 5.7d show the throughput results for the chain and grid topology, respectively. The observations for NSTART=1 also apply for NSTART=4, but with a greater difference between the three CC mechanisms in the dumbbell and chain topologies.

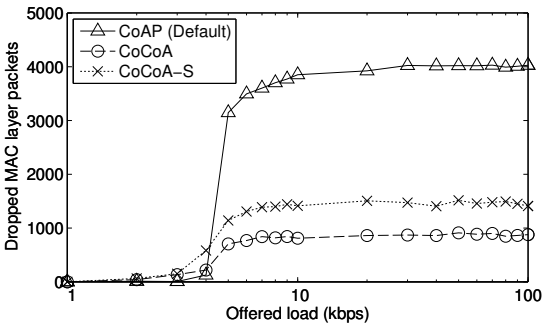
In spite of the performance improvements observed so far in this section, a series of shortcomings of the CC mechanisms have been identified. In the next section, they are introduced and solutions to these shortcomings are proposed, resulting in the shaping of an improved version of CoCoA, also referred to as CoCoA+.

5.7 Improvement of CoCoA: CoCoA+

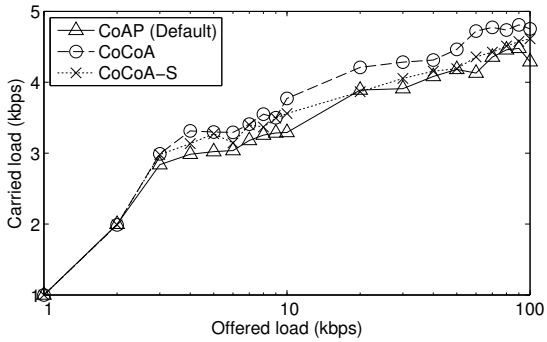
The evaluations presented in the last section showed that the limited CC capacities of default CoAP can be improved with CoCoA. However, due to several shortcomings, CoCoA may perform worse than default CoAP under certain network conditions. In this section, modifications, as well as additions to CoCoA are presented to address these shortcomings, resulting in a new and improved advanced CC mechanism for CoAP, *CoCoA+*. Via simulations of different network and application scenarios it is shown that the improved mechanisms of CoCoA+ promise significant performance



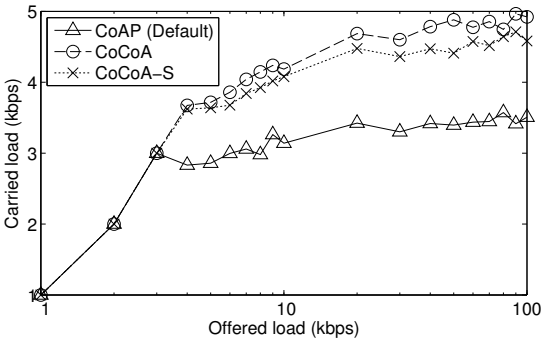
(a) Throughput achieved for different offered loads in the dumbbell topology with NSTART=4.



(b) Dropped MAC layer packets for different offered loads in the dumbbell topology with NSTART=4.



(c) Throughput achieved for different offered loads in the grid topology with NSTART=4.



(d) Throughput achieved for different offered loads in the chain topology with NSTART=4.

Figure 5.7: CC performances for NSTART=4.

improvements for the majority of the evaluated cases: PDR improvements of up to 19.8% and a reduction of average delays during bursts of notification messages of up to 31.2% are measured.

5.7.1 Shortcomings of Default CoAP and CoCoA

In network scenarios with a high number of packet losses due to congestion, subsequent updates of the weak RTO estimator can cause several undesired effects. CoAP allows a total of four retransmissions of a CoAP message, before considering the exchange to have failed. Therefore, a RTT_{weak_new} might be obtained after the second, third, fourth, or fifth transmission. For the source of a confirmable CoAP exchange it is not possible to know to which transmission intent a CoAP ACK corresponds to. According to the CoCoA draft, the RTT_{weak_new} is therefore the time between the reception of the CoAP ACK and the first transmission of the CoAP message.

This behavior has multiple consequences for the calculation of timeout values: first, if a RTT_{weak_new} is measured after applying multiple retransmissions and then it is used to update $\text{RTO}_{overall}$, the new timeout value can grow to a multiple of the previous RTO_{init}. Second, the probability of a RTT_{weak_new} measurement to differ from the actual RTT grows with every retransmission, since it is not clear to which transmission the ACK corresponds. Third, consecutively calculated RTT_{weak} values may differ substantially from each other. According to Equation 5.3, four times the RTTVAR_{weak} value is added to RTT_{weak} to calculate the new RTO_{weak}. Thus, large differences between two subsequent RTT_{weak} values may cause a significant increase of RTTVAR_{weak}, which reflects in an important augment of $\text{RTO}_{overall}$. Figure 5.8a shows the Cumulative Distribution Function (CDF) of RTTVAR_{weak} calculated when using CoCoA in a simulation of a 6x6 grid topology (see the details of the simulation environment and scenarios in Section 5.7.3.1). Less than half of the variances are below 5 s, with a measured average RTTVAR_{weak} of 8.36 s. When nodes measure such high RTTVAR_{weak} values, according to Equation 5.3 they experience a severe augmentation of their RTO_{weak} value. This causes subsequent retransmissions of CoAP messages to spread over time, which helps to reduce the degree of congestion in the network. However, packet losses may not be caused by congestion but by lossy links, due to strong interferences, occluding objects, nodes with mobility, etc. In such a case, contributions of the weak estimator towards a large $\text{RTO}_{overall}$ should be avoided, as they would increase the timeouts for following exchanges further. This can result in an underutilization of the available bandwidth.

While the CoCoA draft in version 0 states that RTO estimators should increment their RTOs if their RTO values are below 1 s, the document does not state if large RTO values should decay over time. The RTT information that has been used to calculate an RTO may have become obsolete over time and the network conditions may have changed.

Another issue arises if RTO_{init} drops to very low values or exceeds the default RTO of 2 s. It was observed that when a confirmable message transmission is initiated with a very low RTO, a node may spend all CoAP retransmissions within a very short amount of time. This is not recommended in the CoCoA draft, however, there is no indication on how to address this problem. On the other hand, when an exchange initiates with a very large RTO and applies several retransmissions with BEBs, the duration of the exchange extends significantly in comparison with that of default CoAP. CoCoA also does not include a decay mechanism for long RTOs that may become obsolete after longer periods of not obtaining fresh RTT measurements.

5.7.2 CoCoA+

In the previous subsection, it was pointed out that there exist several inefficient aspects of default CoAP and CoCoA. In this section changes are proposed to the three fundamental aspects of CC mechanisms for CoAP that were derived in Section 5.3, with

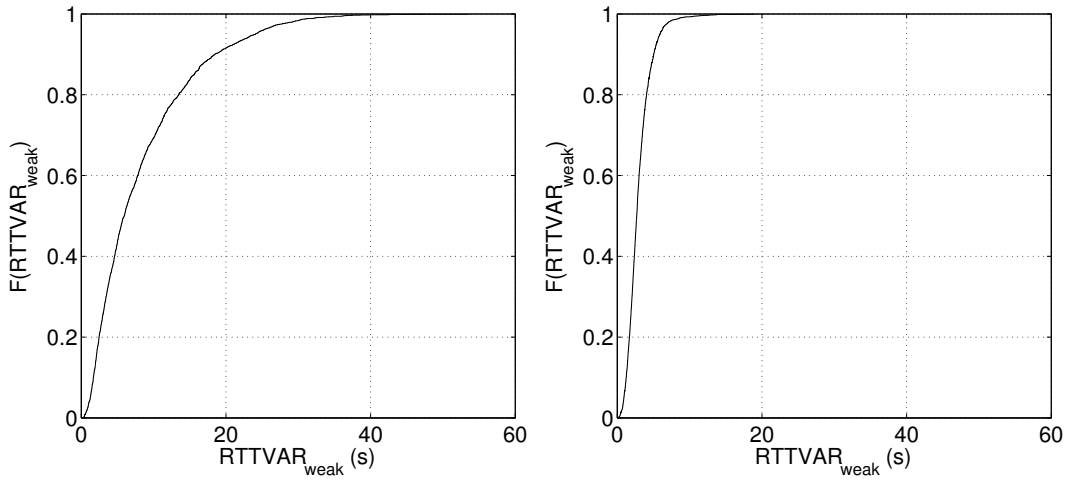


Figure 5.8: CDF for RTTVAR$_{weak}$ values calculated with (a) CoCoA and (b) CoCoA+ during simulations of a 6x6 grid at an overall traffic rate of 6 kbps.

the goal to improve network performance. The combination of all proposals results in a new advanced CC mechanism for CoAP that is called *CoCoA+*.

CoCoA+ comprises the following proposals to address the issues that have been observed when CoCoA is used as CC mechanism:

1. A modification of the weak estimator calculations to reduce the impact of RTT_{weak} variations and its impact on RTO$_{overall}$.
2. A replacement of the BEB used for retransmissions by a Variable Backoff Factor (VBF).
3. A new approach for aging large RTO$_{overall}$ values.

5.7.2.1 Modification of the Weak Estimator

To dampen the impact of large RTTVAR$_{weak}$ values caused by strong fluctuations of RTT$_{weak}$, as explained in Section 5.7.1, the value of K in the calculation of RTT$_{weak}$ (3) is reduced from 4 to 1. A reduction of the K value to a value greater than 1 is not sufficient, considering the distribution of RTTVAR$_{weak}$ that shows a high probability for large values (Fig. 5.8a).

Also, in CoCoA+ the weight of the weak estimator is halved to limit the effect of ambiguities inherited in the weak RTT measurements and the potential strong fluctuations by updating the RTO$_{overall}$ formula for RTO$_{weak}$ as

$$\text{RTO}_{overall} = 0.25 \times \text{RTO}_{weak} + 0.75 \times \text{RTO}_{overall} \tag{5.5}$$

Reducing the weight of the weak estimator further reduces its impact, rendering it inconsequential.

Apart from reducing the impact of RTTVAR_{weak} in the formulas and reducing the weighting of RTO_{weak} in (5.4), it is proposed to limit RTT_{weak} measurements to be obtained only from the first transmission and the first retransmission. RTT measurements obtained after the second retransmission are ignored when this limitation is applied. Therefore, large increments of $\text{RTO}_{overall}$ due to long RTT_{weak} measurements or large RTTVAR_{weak} values that do not reflect current network conditions are avoided more efficiently. Yet, this mechanism allows to utilize the weak estimator in a more reasonable manner.

As a result of the improvements proposed in this subsection, the RTTVAR_{weak} values are considerably smaller in CoCoA+, when compared to those of CoCoA. The CDF of RTTVAR_{weak} values measured by CoCoA+ is shown in Fig. 5.8b.

5.7.2.2 Variable Backoff Factor

The introduction of the VBF is an important change to the backoff behavior used by default CoAP and CoCoA, replacing the BEB applied to retransmissions. Instead of doubling the previous RTO value ($\text{RTO}_{previous}$) to obtain the RTO applied to the next retransmission (RTO_{new}), it is multiplied by a variable factor

$$\text{RTO}_{new} = \text{RTO}_{previous} \times \text{VBF}, \tag{5.6}$$

where VBF depends on the RTO_{init} of a CoAP exchange as follows:

$$\text{VBF}(\text{RTO}_{init}) = \begin{cases} 3, & \text{RTO}_{init} < 1s \\ 2, & 1 <= \text{RTO}_{init} <= 3s \\ 1.3, & \text{RTO}_{init} > 3s \end{cases} \tag{5.7}$$

The VBF mechanism introduces alternative backoff values for low and high values of RTO_{init}:

1. If RTO_{init} of a confirmable message transmission is larger than 3 s, the backoff factor is reduced from 2 to 1.3. By applying this lower backoff factor, the time between two transmissions does not increase as much as in CoCoA, where a high RTO_{init} could lead to unreasonably large backoff values. At the same time the maximum duration of an exchange is reduced, avoiding a long term blocking that could affect other exchanges with the same destination endpoint.

2. If RTO_{init} of a confirmable message transmission is smaller than 1 s, the retransmission mechanism is prevented from spending all available retransmissions in a very short time interval. This helps to avoid further congestion, since retransmissions would be carried out very quickly.

The VBF is a novel addition to CoCoA. The choice of the backoff values is based on two criteria: For large RTO_{init} values, the backoff factor should be smaller than 2 to avoid large idle times. It should, however, not lie below 1 to ensure a safe mechanism in terms of CC. For small RTO_{init} values, the backoff factor should be greater than 2 to avoid spurious retransmissions but not too large to avoid long idle times. Based on these criteria, the values 1.3 and 3 were chosen, respectively. Several alternative backoff values for small and large RTO_{init} values were evaluated, where 1.3 and 3 delivered the best results.

Figure 5.9 illustrates how RTO values evolve for three different RTO_{init} values (0.5 s, 1.5 s, and 6 s) when applying the BEB or VBF mechanisms to retransmissions. While the RTO evolves in the same way for default CoAP and CoCoA+ when RTO_{init} is 1.5 s, clear differences are visible for RTO_{init} values of 0.5 s and 6 s. For the small RTO_{init} value of 0.5 s, the RTO grows faster with the VBF than with the BEB to avoid spurious retransmissions. Contrarily, when the RTO_{init} value is 6 s, the RTO grows slower with the VBF than with the BEB to avoid unnecessarily large idle times in case of losses. Note that in Fig. 5.9, for example, the 4th transmission intent is the 3rd retransmission.

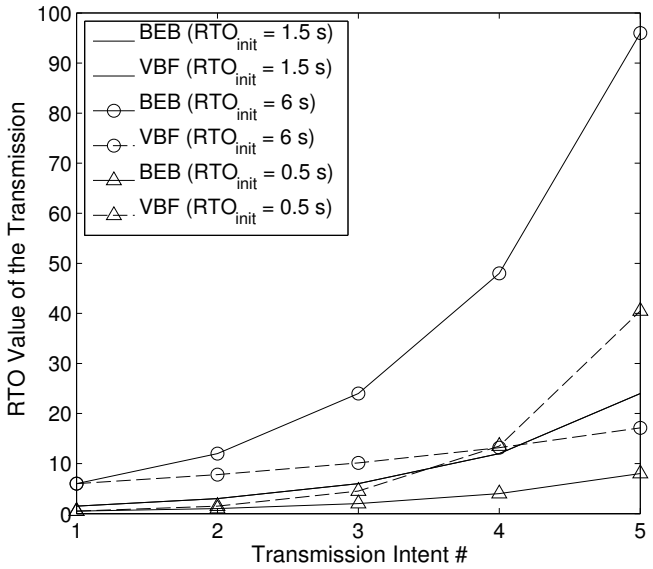


Figure 5.9: Evolution of the RTO timeout value for each transmission when applying a BEB or VBF to initial RTO values of 0.5 s, 1.5 s, and 6 s.

Figure 5.10 compares the total duration of a message transmission with a short RTO_{init} used with BEB and VBF. With the BEB mechanism, all retransmissions are spent quickly in under 5 s, whereas with the VBF the last retransmissions starts after 10 s. The total duration of the transmission before it times out with a BEB is 7.75 s, while with a VBF it is 30.25 s, respectively. The extension of the total duration of a

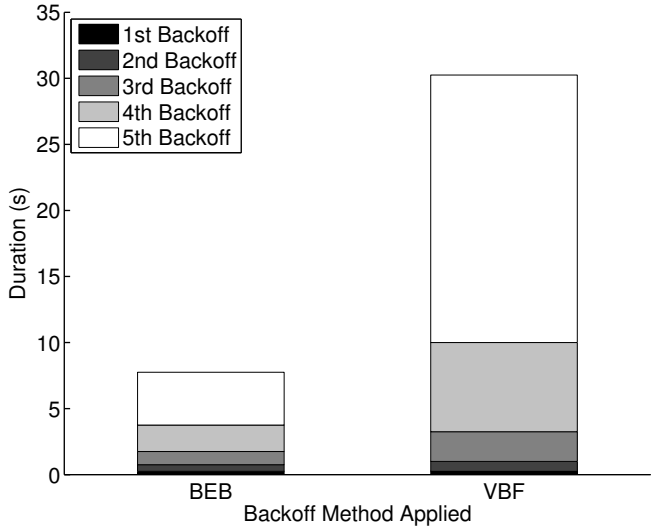


Figure 5.10: Backoff durations for up to 5 message transmissions, starting with $\mathrm{RTO}_{init} = 0.25$ s. The maximum duration of all transmission intents extends from 7.75 s to 30.25 s.

message and the additional delay between retransmissions gives the destination node more time to reply and helps to avoid further congestion due to spurious retransmissions, respectively.

5.7.2.3 RTO Aging

Another improvement of CoCoA+ is an RTO aging mechanism that is applied when RTO estimators with a large $\mathrm{RTO}_{overall}$ value are left without updates for a certain time. It is plausible to assume that RTO information may become obsolete after a certain time if no new RTT measurements are carried out to update the RTO estimators. Thus it is proposed that if $\mathrm{RTO}_{overall}$ is larger than the base RTO from the default CoAP specification of 2 s, and it is not updated during more than 30 s, on the next transmission the $\mathrm{RTO}_{overall}$ is updated as

$$\mathrm{RTO}_{overall} = (2 + \mathrm{RTO}_{overall})/2\ s. \tag{5.8}$$

Applying this formula updates the $\mathrm{RTO}_{overall}$ value of an estimator to a value that is closer to the default value of 2 s. The limit of 30 s for the aging of RTOs was chosen after evaluating several options, including 45 s and 60 s. Applying the 30 s threshold setting in a constant traffic scenario, an average PDR improvement of 0.92% was observed in comparison with the alternative settings of 45 s and 60 s in different network topologies. Thus, the better performing 30 s threshold was chosen for the RTO aging mechanism. However, in other network scenarios and for other use cases, a different aging threshold may work better. An evaluation to determine alternative

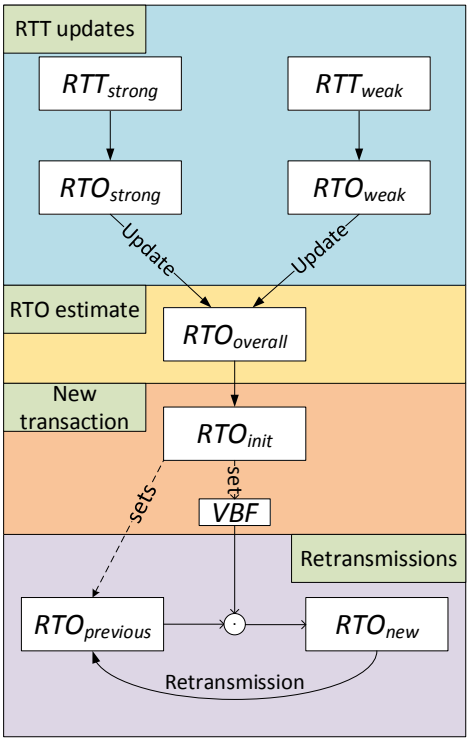


Figure 5.11: An overview of the different RTO variables used to maintain and update the RTO state information for a destination endpoint in CoCoA+.

values or an algorithm to calculate a dynamic threshold is to be carried out as future work.

In this section, the modifications of the weak estimator, the VBF as a new backoff method, and an RTO aging mechanism were introduced to address the shortcomings of default CoAP and CoCoA the CC mechanisms. An overview of the different RTO variables used to maintain and update the CoCoA+ RTO estimates and timers is shown in Fig. 5.11.

5.7.3 CoCoA+: Evaluation Setup

In this section the details on the simulation setup used to carry out performance evaluations of the three CC mechanisms is presented. This includes the configuration of the simulator, the traffic scenarios, the network topologies, and the performance metrics used to carry out the performance evaluations for CoCoA+.

5.7.3.1 Simulation Setup

Like in the introductory analysis of CoAP and CoCoA in Section 5.6.1, the Cooja simulator is used to carry out a performance comparison of default CoAP, CoCoA

based on version 0, and CoCoA+.

The evaluations of the CC mechanisms are analyzed for three different types of traffic scenarios:

1. Constant traffic scenario: a scenario where nodes periodically generate CoAP messages directed towards a sink node. This network scenario allows to analyze how the CC mechanisms perform for different amounts of offered traffic load.

2. Global event scenario: a scenario where, during the simulation, a global event requires all nodes of the network to transmit one or several packets at the same moment to a sink node. Before and after the global event, the network constantly generates a low amount of background traffic directed to the same sink node. In this scenario it is analyzed how the network reacts to the state of sudden congestion and how long it takes for the network to operate normally again. An example scenario is a network where all nodes of the network register a global event, such as heavy vibrations caused by an earthquake, and need to transmit the measured data to the sink.

3. Mixed traffic scenario: a scenario where nodes generate a low amount of constant traffic for a primary sink node but they also generate periodic bursts of notification messages for a secondary sink node. In this scenario the CC mechanisms need to be able to resolve repeatedly states of sudden congestion and at the same time maintain the performance for constant traffic. An example application scenario is a network that collects environmental data, such as temperature, humidity, etc., that is periodically sent to a control center responsible for checking and reacting to the gathered data. Additionally, every minute the nodes send notifications to a proxy node with updated values for a cloud service, that allows web users to check the current state of the network sensor readings.

All CoAP messages are POST messages with a size of 96 bytes that are used to update the state of a variable at their destination.

Four different simulation network topologies with variable amount of static nodes are defined for these traffic scenarios. The network layout inter alia affects how many direct neighbors each node has, how many nodes compete concurrently for the radio channel, and it determines the diversity of available links between any two nodes of the network. The topologies used for the performance analysis are i) a chain of 17 nodes, ii) a dumbbell topology with 21 nodes, iii) a grid of 36 nodes (6x6), and iv) a grid of 49 nodes (7x7).

Each node is assigned a role that determines the type of CoAP messages it generates and whether the node is a possible destination of CoAP messages from other nodes. Figure 5.12 shows the four topologies with one RPL border router (dark node), message generator nodes (bright nodes), primary sink nodes (nodes with a continuous circle), and secondary sink nodes for notifications in the mixed traffic scenario (nodes with a

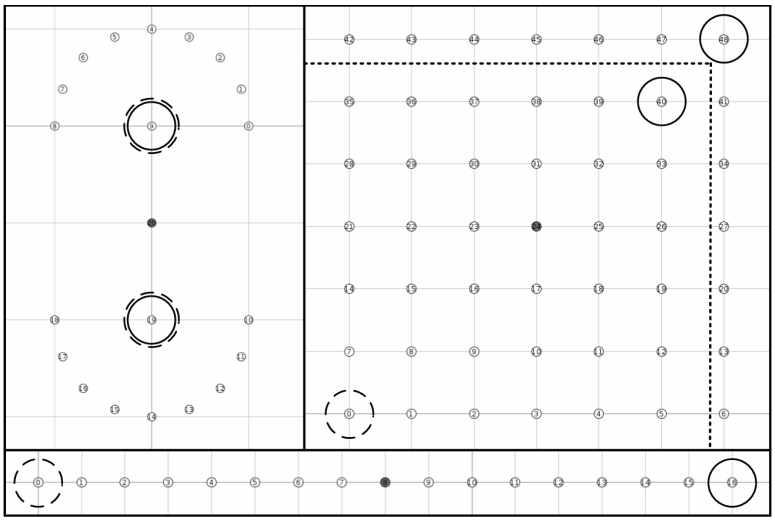


Figure 5.12: The four network topologies used for performance analysis. Starting at the upper left topology and going clockwise: dumbbell, 7x7 grid, 6x6 grid, and chain. The 6x6 grid is depicted as a subset of the 7x7 grid. Nodes with circles are sink nodes for CoAP messages. Dark nodes are RPL border routers. The edges of the unit squares are 10 m long.

dashed circle). The 6x6 grid topology is illustrated as a part of the 7x7 grid topology, however, in the simulation these topologies are separated network scenarios.

As soon as the RPL network is set up (a setup duration of 60 s is assumed), nodes periodically generate CoAP messages that are directed towards a primary sink node. In the global event traffic scenario, additionally all nodes send a notification message to the same sink when a global event is detected. In the case of the dumbbell topology nodes on one side of the dumbbell generate packets for the primary sink on the other side of the dumbbell. Notifications in the mixed traffic scenario are directed towards a secondary sink node that represents a border router/proxy that then forwards the notifications towards an external destination, like a cloud service. In the dumbbell scenario, the notifications are always sent to the secondary sink node that is closest to the origin of the notification (that is on the same side of the dumbbell). Since external entities like a cloud service cannot be simulated in Cooja, the border router only creates ACKs for the notifications it receives, indicating that the notification to external destination will be handled by the proxy.

For the analysis of the CC mechanism carried out in this chapter, a destination endpoint is assumed to be a single IPv6 address, which in the simulations is a single network node. The recommendation of the CoAP base specification is adopted to set NSTART to 1, meaning that only one exchange per destination endpoint is allowed at any time.

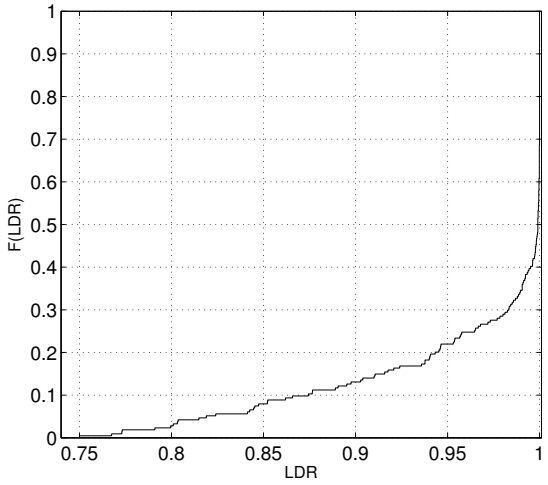


Figure 5.13: Empirical CDF of LDR values observed in a real testbed in the range of 75% to 100%.

For radio transmissions, Cooja's default UDGM radio model with circular transmission and interference areas is applied. The transmission ranges of the nodes are set to be 10 m, which corresponds to a unit square edge in the grids depicted in Fig. 5.12. The interference range of the nodes is set to be 20 m.

When applying the UDGM radio model, Cooja calculates a LDR for a radio transmission, determining with which probability a packet can be received correctly by a destination node within the transmission range. Outside of the transmission range, the LDR is always 0. For the evaluations carried out in this section, two LDR models are applied to the simulations, a static LDR setting and a dynamic LDR setting. When the static setting is applied, the LDR is configured to be always 100%. With this setting, packets are only lost due to packet collisions or due to packet drops as consequence of full buffers.

The dynamic setting is based on a modification of the default UDGM model, where for each transmission a new LDR is calculated. The simulator calculates a random LDR for each packet transmission that follows the CDF (Fig. 5.13) obtained from measurements in a real indoor sensor grid built with 60 TelosB nodes [35]. The LDR for each packet transmission is limited to lie in the interval between 75% and 100%, corresponding to the range of LDR values observed to be normally chosen for data transmissions by link quality aware routing protocols in the grid. Using the dynamic setting, it is possible to observe how the CC mechanisms perform if packet losses occur due to the use of lossy links.

The simulations of the constant traffic and mixed traffic scenarios have a duration of 10 minutes, while the global event scenario has a duration of 5 minutes. For each evaluated configuration, the simulations are repeated 8 times with different random

seeds. Using the simulation setup and the network scenarios described in this section, the different CC approaches for CoAP are evaluated using several performance metrics.

5.7.3.2 CC Performance Metrics

To evaluate and compare the performance of CC mechanisms for CoAP with the given evaluation setup, a set of performance metrics is chosen: The Overall PDR, the end-to-end delay and the Settling time (ST).

The PDR is an indicator of how reliable the network is and whether losses are to be expected. Losses of CoAP messages may lead to network malfunctioning at the application level (e.g. if an alarm is triggered and the message carrying the alarm notification does not reach its destination). As pointed out before, during a congestion state, the probability of losses is high. An effective CC mechanism should be able to detect the state of congestion and take measures to dilute it. As a consequence of applying these measures, it is expected for the CC mechanisms to improve the PDR. From the overall PDR, the network throughput in terms of carried load versus offered load can be derived.

The end-to-end delay is evaluated by measuring the average delay and the CDF of delays. The end-to-end delay is the time it takes for a CoAP message to reach its destination from the moment it is sent at the application layer of the source node to the moment it is received by the application layer at the destination node.

The ST is a new metric that has been defined to determine the time in seconds it takes for the network to revert to a stable state of operation after an event causes a burst of notifications throughout the network. There is no specific formula to calculate ST, the following definition is used: the average end-to-end delay D_{B} of CoAP messages measured prior to the event that causes the burst of messages is used to determine whether the network reverted to a stable state. ST is defined to be the duration it takes for the average delay D_{A} after the burst to get back to a value $D_{\mathrm{A}} <= D_{\mathrm{B}} \times 1.1$. D_{A} is always calculated for a window of 20 s that is shifted forward in time, starting at the time of the global event, until the aforementioned condition is met and ST can be determined. The ST metric is used as performance metric in the global event scenarios.

With this set of performance metrics, it can be analyzed how crucial problems caused by congestion, namely packet losses, large delays, and long ST values are addressed by the different CC mechanisms. In the next section the simulations results are presented for the different traffic scenarios in the order they were introduced before.

5.7.4 CoCoA+: Evaluation Results

The results obtained for the different traffic scenarios in the four network topologies are presented in the following.

5.7.4.1 Results for the Constant Traffic Scenario

In the constant traffic scenarios the CC algorithms are evaluated in 4 different topologies and for varying traffic loads. The traffic generation rate at the nodes is adjusted to vary the average network-wide traffic load from 1 kbps to 10 kbps in steps of 1 kbps.

Considering the simulation results with the lossy link model as detailed in Section 5.7.3.1 (Fig. 5.13), in the majority of cases CoCoA and CoCoA+ increase network performance. In the 6x6 grid topology, the three CC mechanisms perform similarly up to 4 kbps. At higher offered loads CoCoA achieves the highest throughput, followed by CoCoA+ and default CoAP. In the 7x7 grid an analogical behavior of the CC mechanisms as in the 6x6 grid can be observed. In the chain topology, CoCoA+ is able to perform better than default CoAP and CoCoA up to 9 kbps, then it is surpassed by CoCoA. The use of advanced CC mechanisms in the dumbbell topology does not improve performance up to higher offered load (6 kbps). Until reaching this mark, default CoAP performs slightly better than CoCoA+, and CoCoA performs worst.

This means, CoCoA is generally able to provide a good performance compared to default CoAP in the considered conditions (Constant Traffic Scenario, and with MAC layer reliability enabled). The only exception has been observed in the dumbbell topology at low offered traffic rates, where CoCoA underperforms the other CC mechanisms. On the other hand, CoCoA+ performs always better or at least very similar to default CoAP and also performs better than CoCoA in the chain and dumbbell topology. Only the results for the 6x6 grid topology are presented in detail, as an example for a network setup where CoCoA is able to outperform default CoAP and CoCoA+. It is explained under which conditions this is possible and show why in other setups the behavior of CoCoA can lead to degradation of performance. The detailed results for the other topologies are available online [115].

In the 6x6 grid topology (Fig. 5.14), up to offered loads of 2 kbps the three CC mechanisms perform similarly. Both advanced CC mechanisms outperform default CoAP at traffic rates above 2 kbps, where congestion starts having a relevant impact on the overall performance of the network. As with the traffic rates congestion increases, a higher performance is achieved with CoCoA than with CoCoA+. While CoCoA initializes RTOs with large values due to the contribution of the weak estimator to the $\text{RTO}_{overall}$, CoCoA+ normally does not initialize exchanges with such large RTO values. In Figs. 5.15a and 5.15b the values of RTO_{init} in the 6x6 grid scenario with an offered traffic load of 6 kbps are displayed for CoCoA and CoCoA+. While the initial timeouts for a message transmission accumulate below 10 s with CoCoA+, the values spread widely above this value when using CoCoA. The mean RTO_{init} for CoCoA in this case is 9.1 s, while the average value of CoCoA+ is 2.8 s, which is very close to default CoAP's average of 2.5 s.

The main reason for the distribution of RTO_{init} values lies in the limitation of the weak estimator to allow RTT measurements only up to the first retransmission. However, with packet drops as a result of congestion, at higher traffic rates the use of multiple retransmissions due to packet losses is likely. Therefore, CoCoA+ does

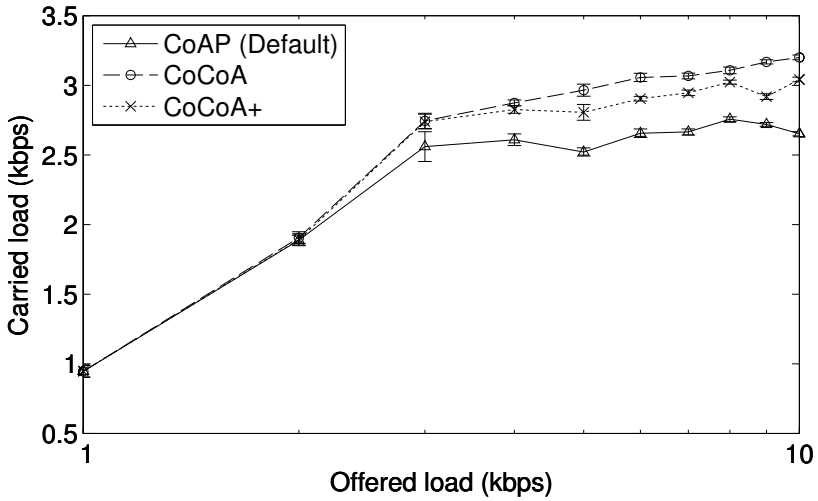


Figure 5.14: Average throughput with standard deviation achieved for different offered loads in the 6x6 grid topology with lossy links.

not update its RTO timers with weak RTTs frequently and when it does, the measured RTTs are considerably smaller than the ones measured with CoCoA. As a consequence, $RTO_{overall}$ values calculated by CoCoA+ are not as large as those calculated by CoCoA. This results in larger RTO_{init} values and in larger backoffs for retransmissions in CoCoA, leading to a reduction of the traffic across the network, which has a positive effect on the congestion and is the reason for CoCoA to perform better than CoCoA+ in this scenario. However, in other network scenarios, the RTO behavior of CoCoA can lead to severe underperformance, which will be shown later.

The simulation results for the 100% LDR links setting are very similar to the ones obtained for lossy links [115]. The main reason for the similar results is the MAC layer reliability mechanism that allows up to three retransmissions of a frame at the link layer. This leads to high one-hop delivery ratios, even with lossy links. However, normally, the application layer does not have any information about characteristics of lower layer (e.g. radio interface) protocols used. Since the MAC layer reliability (which is optional in IEEE 802.15.4) might have a relevant impact on the network performance, it shall be observed how the results may change when MAC layer retransmissions are disabled. Network performance with disabled MAC layer retransmissions and a 100% LDR changes significantly when compared to the case with enabled MAC layer retransmissions. Figure 5.16 shows a completely different evolution of the throughput in the 6x6 grid topology, where CoCoA+ outperforms default CoAP and CoCoA. Latter performs noticeably worse than default CoAP and CoCoA+ in this network scenario.

The amount of message transmissions that start with large RTO values in CoCoA is high and the timeouts get larger when retransmissions are necessary. While delaying retransmissions by using large timeouts may help to reduce congestion, waiting for

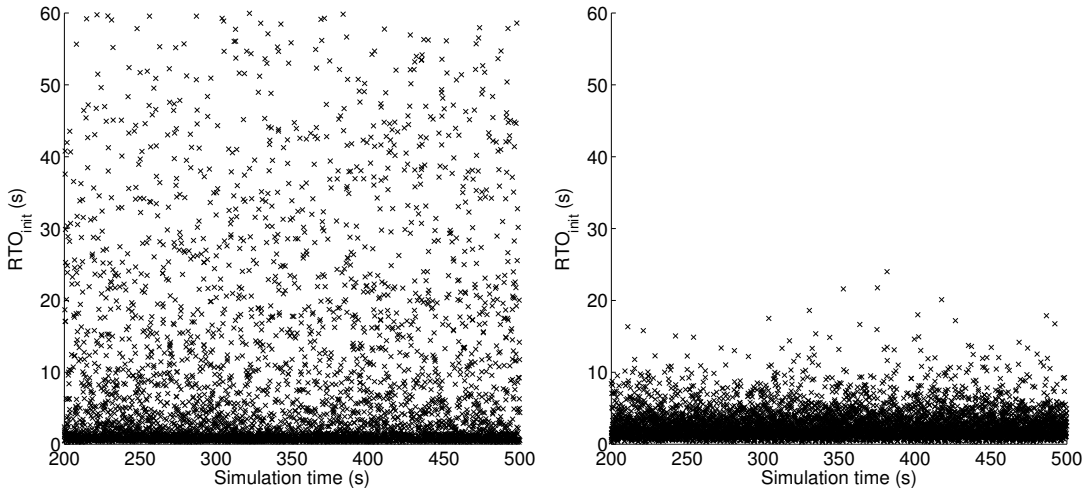


Figure 5.15: Comparison of RTO_{init} applied to message transmissions during 5 minutes in the simulation of the 6x6 grid topology with lossy links at a traffic rate of 6 kbps with CoCoA (a) and CoCoA+ (b).

ACKs that will not arrive in case of packet losses blocks the exchange of other messages to the destination node due to the NSTART = 1 limitation. This behavior can lead to a substantial reduction of the PDR due to packet drops at application layer. The average timeouts applied by CoCoA+ are larger than those of default CoAP, but their range is reduced in comparison to CoCoA, since RTT_{weak} measurements are only allowed up to the first retransmission. Above that, CoCoA+ uses the VBF on large RTO_{init} values to limit the growth of the retransmission backoff. As a result of the improvements included in CoCoA+, when it comes to providing end-to-end reliability without additional per-hop reliability, CoCoA+ turns out to be the best solution, while CoCoA performs worst. The same tendency for the scenario with disabled MAC layer retransmissions also applies to the other topologies [115].

The results for the constant traffic scenario show that in networks with lossy and error-free links CoCoA normally delivers a good performance. Within these scenarios, CoCoA+ is generally able to outperform CoCoA at low traffic rates and performs very similar to or noticeably better than default CoAP. In scenarios with disabled MAC layer retransmissions, CoCoA+ outperforms the other CC mechanisms, and CoCoA underperforms default CoAP to a significant extent.

5.7.4.2 Results for the Global Event Scenario

In the next step of the performance evaluation, the settling time (ST) is analyzed for the global event scenarios. In these scenarios the nodes produce an overall average

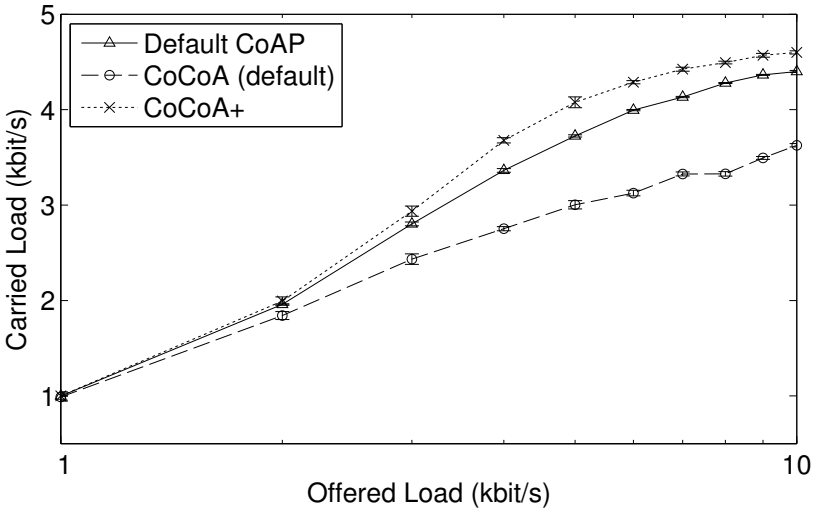


Figure 5.16: Average throughput with standard deviation achieved for different offered loads in the 6x6 grid topology without MAC layer retransmissions and 100% LDR links.

network load of 4 kbps in the grid topologies and 2 kbps in the other topologies towards the primary sink. After 2 minutes a global event causes all CoAP nodes to send a notification message to the same sink. This burst of notifications causes a sudden state of congestion across the network that needs to be resolved by the CC mechanisms. After the burst, the network continues to operate normally again. The performance of the CC mechanisms in this scenario is measured by calculating the ST, which indicates the number of seconds that have to pass until a state close to the initial network state is recovered.

When the global event occurs, all nodes send notifications to the sink, which results in heavy congestion with packet losses and thus retransmissions are necessary. During such a burst, CoCoA and CoCoA+ measure many weak RTTs, leading to an increase of the overall RTO. After transmitting all the notifications, the grid starts operating normally again. Default CoAP has no state memory and applies the default RTO_{init} to all message transmissions. Upon a state of sudden congestion it therefore does not adapt its RTO values. The RTO timers of CoCoA and CoCoA+ after the burst are still affected by the RTT measurements obtained during the state of heavy congestion and need to adapt to the normal traffic again.

Table 5.3 lists the STs for the three CC mechanisms in all topologies considered for the static 100% LDR setting and the lossy link setting (75% - 100% LDR) with enabled MAC layer retransmissions. In addition to the average ST, the 95% confidence intervals are given, indicating how much the ST values oscillate for the repetitions of the burst traffic simulations. In the grid and chain topologies, CoCoA+ recovers from congestion faster than the other CC mechanisms, except for the 7x7 grid scenario with dynamic LDRs, where default CoAP recovers as fast. When using CoCoA, it takes longer for

the network to get back to its original state, since the RTOs are set to very high values and packet losses lead to large delays. Also, high RTO values calculated during the notification burst require the exchange of several CoAP messages in the not congested network to get back to low values. Default CoAP does not increase its RTO timers during the burst event and therefore continues transmitting messages during and after the burst with default RTOs. As a result, it does not adapt its initial RTOs like CoCoA and CoCoA+ do. One one hand this means that it avoids the issues CoCoA has when adapting to the congestion by heavily incrementing the RTOs. On the other hand, it does not increase the RTO values at all. As a consequence of behaving independently from network conditions, it can be observed that the performance of default CoAP in comparison to the other CC mechanisms varies with each topology. In the grid topologies it performs better than CoCoA and in the chain and dumbbell topologies it performs worse than CoCoA. However, CoCoA+ always outperforms default CoAP or performs at least as well.

In the dumbbell topology, CoCoA performs better in terms of ST than CoCoA+. This can be explained by comparing the PDR values of the two settings. While in the majority of scenarios the PDR obtained by the three CC mechanisms differs only slightly ($< 2\%$), in the dumbbell topology a difference in the PDR of more than 10% in favor of CoCoA+ has been observed. In this specific case, the higher reliability comes at the cost of a larger settling time, since a higher delivery ratio is achieved after the burst, at the expense of increased delivery delay. Leaving this specific case apart, CoCoA+ is the mechanism that adapts the fastest to short term changes of the network congestion and performs similarly or better than default CoAP in all the considered scenarios.

Table 5.3: Average settling times and 95% confidence intervals of the settling times for different topologies and LDR settings (the best performing mechanism is highlighted with bold letters).

Topology / LDR setting	Default CoAP	CoCoA	CoCoA+
6x6 Grid / 100%	47 ± 6 s	57 ± 6 s	**38 ± 7 s**
6x6 Grid / Lossy	33 ± 10 s	42 ± 9 s	**21 ± 6 s**
7x7 Grid / 100%	44 ± 11 s	48 ± 7 s	**36 ± 7 s**
7x7 Grid / Lossy	23 ± 7 s	33 ± 6 s	**19 ± 8 s**
Chain / 100%	**15 ± 8 s**	17 ± 9 s	**15 ± 9 s**
Chain / Lossy	14 ± 5 s	12 ± 8 s	**10 ± 6 s**
Dumbbell / 100%	19 ± 8 s	**8 ± 5 s**	12 ± 6 s
Dumbbell / Lossy	17 ± 5 s	**8 ± 4 s**	13 ± 2 s

5.7.4.3 Mixed Traffic Scenario

The last part of the performance analysis focuses on the mixed traffic scenario with two separate sinks. While the traffic load directed to the primary sink is fixed at an overall network average of 3 kbps, the notifications for the secondary sink are generated every 60 s. This results in parallel cross traffic for this more complex scenario, in which the CC mechanisms need to be able to adapt to repeated global bursts of notifications that compete with the constantly generated traffic.

This scenario is tested with the static and dynamic link settings with enabled MAC layer retransmissions. Moreover, it is evaluated how the network performance changes, if different amounts of notifications (1, 2, or 3) have to be transmitted every 60 seconds. The results shown in the following only consider the case for 1 notification, the rest of the results can be found online [115]. The performance of the CC mechanisms in this scenario is determined by comparing the overall PDR, the average delay from the start of the notification bursts up to 10 s after the burst, and the distribution of end-to-end delays.

Table 5.4 shows the overall PDR values for all three CC mechanisms, including the 95% confidence intervals for both LDR settings when a single notification is sent per burst. As seen in the table, except a minor deterioration of 0.4% in one scenario, in all other scenarios, CoCoA+ achieves a better performance than default CoAP and CoCoA.

Table 5.4: The overall PDR values with 95% confidence intervals for different topologies and LDR settings (the better performing mechanism is highlighted with bold style).

Topology / LDR setting	Default CoAP	CoCoA	CoCoA+
6x6 Grid / 100%	85.2 ± 0.8%	82.7 ± 1.8 %	**85.8 ± 0.9%**
6x6 Grid / Lossy	76.7 ± 2.1%	73.9 ± 3.2%	**79.2 ± 3.9%**
7x7 Grid / 100%	80.6 ± 2%	78.3 ± 1.5%	**82.3 ± 2%**
7x7 Grid / Lossy	67.1 ± 1.9%	70.3 ± 1%	**73.6 ± 1.9%**
Chain / 100%	**92.7 ± 0.2%**	91.9 ± 0.8%	92.3 ± 0.6%
Chain / Lossy	88 ± 0.8%	86.8 ± 1%	**89.1 ± 0.8%**
Dumbbell / 100%	33.5 ± 0.5%	30.3 ± 0.9%	**40.2 ± 0.5%**
Dumbbell / Lossy	36.5 ± 3.5%	31.7 ± 4.6%	**38.2 ± 3.1%**

Delay is also an important metric, especially for applications that require fast reactions and short notification times, such as alarms. In the mixed traffic scenario the average delays of message transmissions initiated at the beginning of each burst up to 10 s after each burst are calculated. Table 5.5 shows the average delays with their corresponding 95% confidence intervals for all three CC mechanisms. In the majority of cases, CoCoA suffers from larger delays than default CoAP, unlike CoCoA+ which

is able to deliver CoAP messages faster. An exception to this pattern has been observed in the dumbbell topology. In this topology, the mechanisms used by CoCoA+ are providing a substantially higher PDR than default CoAP, which comes at the cost of larger delays. Also, it needs to be taken into account that the average delays without congestion in the dumbbell topology are rather short (less than 100 ms), thus, differences of the average delay values between the three CC mechanisms are rather small. Despite of the slightly worse performance in terms of average delays in the dumbbell topology, using CoCoA+ is the best option if CoAP messages need to be delivered fast. Relative delay decrease of more than 30% is possible with CoCoA+ in comparison to default CoAP, while providing a higher reliability in all of the considered cases.

Table 5.5: Average delays and correspondent 95% confidence intervals during bursts and 10 s after the burst for different topologies and LDR settings (the best performing mechanism is highlighted with bold style).

Topology / LDR setting	Default CoAP	CoCoA	CoCoA+
6x6 Grid / Static	1444 ± 50 ms	2051 ± 123 ms	**1350 ± 44** ms
6x6 Grid / Dynamic	2838 ± 122 ms	3283 ± 271 ms	**2392 ± 188** ms
7x7 Grid / Static	2676 ± 215 ms	3083 ± 287 ms	**2259 ± 172** ms
7x7 Grid / Dynamic	5014 ± 287 ms	5401 ± 187 ms	**3820 ± 170** ms
Chain / Static	**476 ± 29** ms	545 ± 29 ms	486 ± 34 ms
Chain / Dynamic	976 ± 31 ms	997 ± 46 ms	**939 ± 26** ms
Dumbbell / Static	63 ± 3 ms	**55 ± 3** ms	76 ± 6 ms
Dumbbell / Dynamic	92 ± 10 ms	**88 ± 13** ms	120 ± 23 ms

The delay information given so far does not depict information about the distribution of the delays. Therefore, we evaluate the CDF of the delays for the different CC mechanisms. End-to-end delays of 60 s and more appear as 60 s in the statistic. Unsuccessful CoAP exchanges with an 'infinite' delay are integrated in the CDF. Since analyzing the CDFs for each combination of network topology and traffic scenario is not possible due to space restrictions, only the CDFs of an example scenario are shown. This scenario is representative and shows a general tendency for cases where CoCoA+ achieves higher PDR values than the other two CC mechanisms. According to the previously shown results, this is applies to a large part of the considered cases.

Figure 5.17 shows the CDF for end-to-end delays in the 7x7 grid scenario with lossy links, where 1 notification per burst is transmitted. The CDF comparison reveals that CoCoA+ clearly has the highest probability of achieving short delays ($< 10\,s$), while CoCoA clearly underperforms default CoAP except for very short delays ($< 3\,s$). The CDF curves of default CoAP show a staircase pattern that repeats throughout all obtained CDFs. Since default CoAP applies a BEB to initial RTO values from a fixed interval, retransmissions are carried out also within a fixed time interval, leading to the

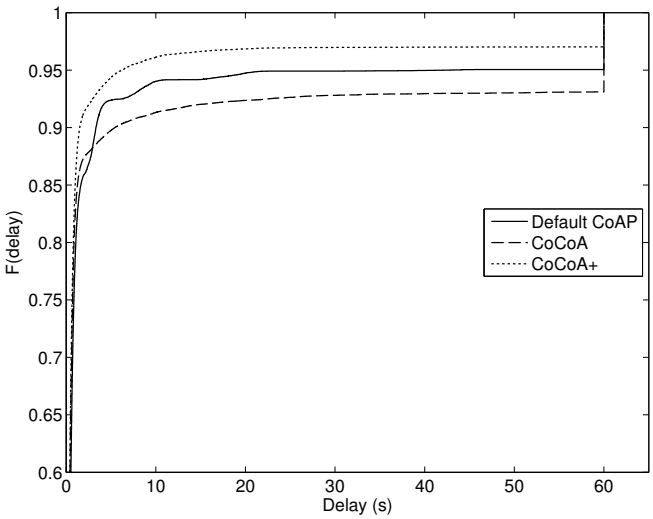


Figure 5.17: CDF of delays for the 7x7 grid topology with lossy links and bursts of 1 notification in the mixed traffic scenario.

visible accumulation of delays. CoCoA and CoCoA+ do not show these patterns, as their initial RTO is not chosen from a fix interval, but dynamically changes depending on the measured RTT values. The RTO calculations of CoCoA+ result in the highest probability of delays below 10 s. CoCoA also has higher probability of providing delays lower than about 3 s, however, due to the lower PDR achieved by CoCoA when compared to the other CC mechanisms, the probability of delays greater than 3 is the smallest. Eventually, all curves reach an asymptotic value after some time, to jump to 100% at 60 s. For the observed scenario this indicates that there are very few measurements with large delays shorter than 60 s. Most of the delays larger than 60 s are caused by lost packets that have an 'infinite' delay. Judging from the CDF, CoCoA+ offers the best performance with the highest probability for short delays.

In this section the performances of three CC mechanisms for CoAP in three traffic scenarios and four topologies have been evaluated. The results show that there is no CC mechanism that always performs best in all scenarios and for all performance metrics. While CoCoA performs well in a variety of scenarios, it often performs noticeably worse than default CoAP. The new proposal for an advanced CC mechanism for CoAP presented in this chapter, CoCoA+, performs better than default CoAP in the majority of scenarios or at least very similarly. The results in this chapter are an important and necessary step towards the definition of a solid alternative to the CC mechanism provided by default CoAP.

Based on the results obtained in this section, CoCoA+ is considered to be a very solid and promising proposal for an advanced CC mechanism for CoAP. A large part of

the findings of this study has been used to shape versions 1 and 2 of the CoCoA draft [BBGD14]. The changes applied to the draft are detailed in the following section.

5.8 Update of the CoCoA Internet-Draft

The evaluations of CoCoA shown in the previous section have been proposed to the IETF CoRE working group. After discussing the results and change proposals with the author of the CoCoA draft, Dr. Carsten Bormann, and the working group, the draft was updated to version 1. This added new content and changed the existing one based on the proposals made.

After this update, further changes were discussed with the CoRE working group and the main author, which shaped version 2 of the draft and led to our co-authorship of the draft. In the following the changes that come with version 2 of the CoCoA draft are detailed.

5.8.1 Blind RTO Estimate

The use of NSTART > 1 showed that especially when several transactions are started in quick succession while no RTT information is able for a destination endpoint, the initial RTO values grow very quickly. Using very large initial RTOs when there is no RTT information available may result in performance drops when packet losses are observed. For example, in the case of NSTART $= 4$, the 4th transaction uses an initial RTO of [16 s , 24 s]. If the correspondent CoAP message is lost but there is no congestion or only a congestion peak that lasts a few seconds, the retransmission will be delayed in average by 20 s. Considering there might not be any congestion (since no RTT information was gathered) this can result in long idle times. Thus the related text in the draft is changed, so RTO_{init} for blind RTO estimates now no longer grows exponentially with each additional parallel transaction, instead it grows linearly. The calculation of RTO_{init} for a new transaction $i+1$, where i is the number of already ongoing parallel transaction towards a destination endpoint is set to

$$\text{RTO}_{new} = 2s \times (i+1). \tag{5.9}$$

5.8.2 Measured RTO Estimate

The changes to the weak estimator proposed in this thesis have been adapted with some minor modifications.

1. Only the RTT measurements of the first two retransmissions are used for updates of the weak RTO estimator.

2. The K factor used in the weak estimator formulas is set to 1:

$$\text{RTO}_{weak} = RTT_{weak} + 1 \times \text{RTTVAR}_{weak} \tag{5.10}$$

3. While the weighting of the strong RTO estimator on $\text{RTO}_{overall}$ updates is maintained, the weighting of weak RTO estimator updates on the overall RTO is reduced to 0.25:

$$\text{RTO}_{overall} = 0.25 \times \text{RTO}_{weak} + 0.75 \times \text{RTO}_{overall} \tag{5.11}$$

4. The VBF is used as follows:

$$\text{VBF}(\text{RTO}_{init}) = \begin{cases} 3, & \text{RTO}_{init} < 1s \\ 2, & 1 <= \text{RTO}_{init} <= 3s \\ 1.5, & \text{RTO}_{init} > 3s \end{cases} \tag{5.12}$$

5. The aging mechanism is updated slightly, compared to the version proposed in this thesis: An RTO value is considered to be small below 1 s and large above 3 s. A small RTO value is updated if after 16 times the current RTO value no update has been obtained. A large RTO value is updated if after 4 times the current RTO value no update has been obtained. The values are updated then as follows:

$$\text{VBF}(\text{RTO}_{new}) = \begin{cases} \text{RTO}_{previous} \times 2, & \text{RTO}_{current} < 1s \\ 1s + (0.5 \times \text{RTO}_{previous}), & \text{RTO}_{current} > 3s \end{cases} \tag{5.13}$$

The updated version of the CoCoA draft integrates nearly all proposals made as stated so far in this thesis. The updates applied to some of the algorithms are fruits of further discussions between the CoRE working group and the authors of the draft.

5.9 Conclusions

In this chapter, the basic CC mechanisms for the IoT protocol CoAP are introduced, alongside with proposals for advanced CC mechanisms. From these proposals, CoCoA is chosen to be implemented [Bet15] and evaluated with regard to its suitability as future standard advanced CC mechanism for CoAP-based IoT communications. The results obtained via simulations of different network scenarios in Cooja show important performance improvements when using CoCoA instead of the default CC mechanism. Also, a simplified version of CoCoA with only a strong RTO estimator, namely CoCoA-S, shows noticeable improvements of the end-to-end network performance.

The CoCoA CC mechanism for CoAP based on Draft version 0 on average performs equal to or better than the default CC mechanism in terms of throughput. CoCoA is

able to reduce the quantity of MAC layer buffer overflows for congested networks, an important aspect in constrained networks. The results obtained with NSTART set to 1 could be confirmed for the case where NSTART is set to 4, where an even greater difference between the performance of the default CoAP CC and CoCoA was observed.

The simplified CoCoA-S mechanism does not perform as well as the CoCoA mechanism, accentuating the need to use the weak RTO estimator, as opposed to using only the strong RTO estimator. However, CoCoA-S is able to outperform the default CoAP CC mechanism, while not requiring as much state information as CoCoA. With the results obtained in this section, the relevance of advanced CC mechanisms for CoAP was confirmed.

However, a continuative in-depth analysis of the different CC mechanism reveals several shortcomings of CoCoA that prevent it from matching the performance of default CoAP's CC mechanism under certain network conditions due to some design flaws. Therefore, modifications and additions for CoCoA are proposed, including the novel VBF and an updated aging mechanism for outdated RTO values. Simulations of different network topologies and traffic scenarios show that the improved version of CoCoA, named *CoCoA+*, is capable of surpassing the performance of default CoAP in all analyzed network scenarios, including those where the original CoCoA performs worse than default CoAP. The improvements add robustness to CoCoA, yielding higher PDR values, shorter delays, and shorter STs. CoCoA+ then provides a higher degree of reliability and lower delays. In the analyzed scenarios and topologies, a PDR improvement of up to 19.8% and a reduction of average delays during bursts of notifications of up to more than 30% are observed. Additionally, it is resilient against sudden changes of network traffic and adapts quickly to different states of network congestion. In some scenarios, CoCoA+ does not achieve the same performance as CoCoA, however, it then performs still better than or very similarly to default CoAP. On the contrary, CoCoA is unable to provide a consistently better performance than default CoAP, often underperforming it significantly.

Eventually, the results of the evaluations of this chapter are used to reshape and rewrite the CoCoA draft, incorporating most of the proposed additions and modifications and resulting in two updated versions of the IETF draft. In the following chapter, evaluations of the CC mechanisms for CoAP are performed in IoT network scenarios with real hardware and Internet communications to underline the relevance of CoCoA and how it improves the end-to-end performance.

6

Evaluation and Verification of the CoCoA Design for the IoT

In the previous chapter, *CoCoA*, an advanced CC mechanism for CoAP-based communications in the IoT, has been introduced. CoCoA is evaluated with regard to its impact on the end-to-end performance of CoAP communications and shows performance improvements in most of the analyzed networks scenarios. Yet, under specific network conditions, CoCoA shows a poorer performance than the default CC mechanism defined in the CoAP base specification. After determining the shortcomings that cause CoCoA to underperform, its CC mechanisms are optimized and new CC mechanisms are added. Its performance then is compared to default CoAP and the original version of CoCoA. The improved version of CoCoA that results from these efforts, CoCoA+, shows clear improvements over the default CC mechanism defined in the CoAP base specification in congested networks. Moreover, it does not lead to a significant degradation of the performance in any of the analyzed scenarios, unlike the initial version of CoCoA.

This chapter complements the investigation on end-to-end performance improvement for CoAP with additional in-depth evaluations of the CC mechanisms in real network setups. These evaluations comprise the use of different wireless communication technologies to connect devices over CoAP, focusing on several performance metrics like reliability, robustness, bandwidth efficiency, and fairness. It can be shown that the design choices of CoCoA+ are justified: The advanced CC mechanism for CoAP leads to network performance improvements in a variety of traffic scenarios and network setups. The first part of this chapter addressing the evaluation of CoCoA+ for cloud services has been published in [BGDK14] in collaboration with Matthias Kovatsch from the Institute for Pervasive Computing at ETH, Zürich. The content of the second part of this chapter is currently under review as a journal publication [BGDPew]. Further, the work on CoCoA resulted in the publication of two CoCoA Implementations for Erbium

[Bet15] and Californium [Bet14], a Contiki implementation and a Java implementation, respectively.

6.1 Introduction

CoAP is designed to be the de facto application layer protocol to be used by billions of constrained devices in the IoT. The challenge of handling congestion in IoT networks is an important one and inefficient CC mechanisms can lead to important performance drops, as shown in the previous chapter. In spite of the relevance of CC for CoAP, it could be shown that it is addressed insufficiently by the CC mechanism applied by CoAP. In order to provide a CC mechanism that is capable of performing well in typical IoT traffic scenarios, CoCoA[1] has been designed.

The evaluation results of the simulations for CoAP and CoCoA presented in the previous chapter show that CoCoA is capable of improving the end-to-end performance. Yet, evaluations of CoAP's CC mechanism that involve the interaction of devices over the Internet have not been carried out so far. To simulate communications between devices that are connected over the Internet accurately within a simulator is difficult. Thus, in order to strengthen CoCoA as a candidate for an official advanced CC mechanism for CoAP and to endorse that CoCoA improves end-to-end performance for IoT communications, further evaluations of CoAP communications carried out in real testbeds are presented in this chapter.

The performance evaluations involve the analysis of CC mechanisms for CoAP in IoT network setups with real hardware. The evaluations of CoAP CC are not limited to wireless IEEE 802.15.4 networks, but include the analysis of CoAP communications over alternative constrained and unconstrained communication technologies, such as GPRS and IEEE 802.3, respectively. The experiments in these testbeds cover several aspects of CoAP communications that cannot be covered in simulations or represented insufficiently. Among other things, this includes the use of real radio channels with complex signal propagation, real bit errors, and the effects of external signal interferences. Further, in contrast to the setups analyzed so far, the devices in the evaluated scenarios communicate over the Internet. Since congestion is a recurring issue that can affect any type of networks, other well-known protocols, like TCP, also apply CC mechanisms. To prove that the design choices made in CoCoA to perform well in IoT traffic are justified, CoCoA is compared with alternative state-of-the-art CC mechanisms that have not been specifically designed for IoT communications.

In the first part of this chapter, the performance of the default CoAP and CoCoA CC mechanisms is evaluated when applied in Internet cloud services. From the evaluation results obtained, it can be shown that CoCoA achieves much better performance

[1]Please note that for the remainder of this chapter, for simplification purposes the improved version of CoCoA (CoCoA+) is referred to as 'CoCoA'.

over default CoAP in such scenarios. Moreover, it can be demonstrated that the conservative restrictions imposed by the CoAP base specification can be relaxed when using CoCoA, in particular the NSTART limitation. In the second part of this chapter, a set of final performance evaluations of CoCoA for alternative communication technologies, including GPRS and IEEE 802.15.4, and a comparison with other advanced CC mechanisms are presented. The extensive comparisons, involving different communication technologies, traffic scenarios, and network setups conclude the work on performance improvements for CoAP, showing that the proposal for an advanced CC mechanism made in this thesis improves performance for CoAP-based IoT communications in many ways.

The remainder of this chapter is structured as follows: Section 6.2 introduces the implementation of CoCoA for Erbium (Er), as well as the Californium (Cf) Java framework and optimizations made to this implementation. The evaluations of CoAP for Internet cloud services is presented in Section 6.3. In Section 6.4 a comparison of default CoAP and CoCoA with alternative state-of-the-art CC mechanisms is carried out in two testbeds. The conclusions to this chapter are given in Section 6.5.

6.2 CoCoA for Cloud Services: Implementation and Optimization

It was shown that CoCoA yields an even higher performance than CoCoA and is capable to overcome the shortcomings identified in Section 5.7.1. As a result, most of the changes proposed in this thesis were adopted in version 2 of the CoCoA draft. Apart from the resulting contributions in conference papers, journal papers, and the CoCoA Internet-draft, the work of CoCoA has also resulted in the publication of source code for two different frameworks.

In the evaluations carried out in the previous chapter, CoCoA, was implemented for Erbium, the official CoAP implementation for Contiki [116][2]. The Erbium implementation of CoCoA includes all the features as detailed in version 2 of the CoCoA draft [BBGD14]. In this section, the implementation of the CoCoA draft version 2 is presented for the Cf framework, a Java implementation of CoAP for unconstrained devices [117].

In the following, the CoCoA implementation for Er and the CoCoA implementation for Cf is presented for its use on constrained and unconstrained machines that run CoAP-based Internet cloud services. Since the Cf implementation runs on unconstrained devices that are not imposing strong limitations on the available memory (ROM and RAM), two additional improvements for CoCoA are introduced that result in a slightly higher memory consumption of the advanced CC mechanism.

[2]To be released soon as official addendum in the Erbium Git repository

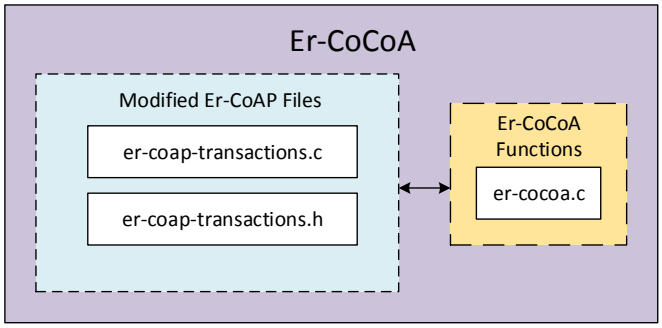


Figure 6.1: Er-CoCoA consists of the original Er-CoAP files that have been modified and the additional er-cocoa.c file carrying the methods to manipulate the RTO estimators.

6.2.1 Implementation of CoCoA for Erbium

For the analysis carried out in this chapter, CoCoA was implemented for Er-CoAP. It consists of the slightly modified er-coap-transactions.c and er-coap-transactions.h files from the original Er-CoAP implementation, responsible for managing CoAP transactions, and the er-cocoa.c file (see Fig. 6.1).

The er-cocoa.c file provides all the functions necessary to initiate, update, and maintain the RTO estimators for a destination endpoint. For each destination endpoint a *coap_rtt_estimations* structure is maintained, that contains all the RTO estimator variables. The Er-CoCoA implementation [Bet15] is optional and can be activated by setting the *COCOA* flag at compile time. It is compatible with the newest version of Er-CoAP available at the time of writing this thesis (implementing the CoAP RFC) and is to be included in the official Git repository for Er-CoAP.

6.2.2 CoCoA Californium Implementation (pre-Git version)

CoCoA is implemented as an optional CC layer for the Cf CoAP framework that has been released in the official Git repository of Cf CoAP under an Eclipse Foundation license [Bet14]. The implementation provides all necessary data structures and methods to carry out the mechanisms as defined in the draft version 2. A `RemoteEndpoint` object stores the information needed by the CC layer, such as the RTT and the current state of the RTO estimators (see Fig. 6.2).

Since Californium is designed for unconstrained environments, each remote endpoint is identified at the highest granularity, that is, by its unique IP address and port number. This allows to even react to congestion at specific services, which are provided at different UDP ports of a host.

Whenever a new request-response exchange is created, it is associated to its remote endpoint. After performing this association, it is possible to access the state information of the remote endpoint from all layers of the CoAP stack, as the `Exchange` object is

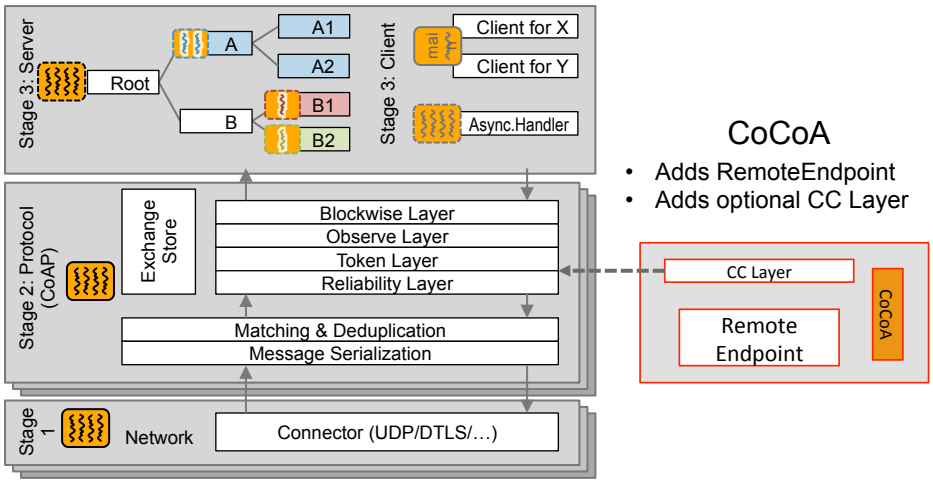


Figure 6.2: Simplified overview of the Californium CoAP stack [4] with the optional CoCoA layer in the pre-Git version.

passed around. The CC layer first determines whether the transmission is a CON or a NON, as each type is processed differently.

When an exchange with a CON message reaches the CC layer, the NSTART limit for open interactions with the corresponding remote endpoint is checked. If less than NSTART exchanges are active to that remote endpoint, the request can be processed and it is forwarded to the reliability layer. In this context, observing does not count as active exchange. If the limit of NSTART is already reached, the new request is added to a queue that is bounded through a maximal lifetime for stored exchanges. As soon as an incoming reply is handed up from the reliability layer, the CC layer updates the RTO estimators of the associated endpoint. After updating the state information of the remote endpoint, the outstanding interaction is closed and the next CON exchange can be pulled from the queue.

When a NON exchange is handed over to the CC layer, it is stored in a queue, which implements a leaky bucket traffic shaper [118] for the associated remote endpoint. With a rate of $1/RTO_{overall}$, NONs are pulled from the queue and handed down to the lower layers in the stack. Following the guidelines of the CoCoA draft, every eighth NON is converted to a CON in order to obtain a RTT measurement to assure that the RTO value gets updated from time to time. The evaluation of this mechanism is out of scope of this chapter, though, as the focus lies on the adaptation of the RTO value for CON requests.

6.2.3 CoCoA Californium: Optimizations over the Draft

For the evaluations carried out in the following section and in line with the ongoing optimization of the CoCoA draft, some of CoCoA's mechanisms are modified and new

ones are added. The aging mechanism for RTO estimators with large RTOs is modified experimentally. If an RTO estimator was not updated for at least one minute and its RTO value is larger than the default of two seconds, the RTT and RTTVAR of the estimators are adjusted to be reduced:

$$\mathrm{RTT}_X = (2 + \mathrm{RTT}_X)/2; \mathrm{RTTVAR}_X = \mathrm{RTTVAR}_X/2. \tag{6.1}$$

The reasoning for this mechanism is that RTT information may become obsolete after some time and may no longer reflect the current network state, in particular when low-power wireless links are involved.

Another improvement to avoid strong fluctuations of the RTO is the use of a history with configurable size for implementations of CoCoA for unconstrained devices, where the last calculated values for $\mathrm{RTO}_{overall}$ are used to calculate an averaged RTO value. Alternatively, this behavior can be approximated by adjusting the weights used in the calculation of the estimator. However, it is easier to implement in form of a RTO history, given the higher amount of RAM available to the unconstrained devices.

6.3 Evaluation of CoAP for Cloud Services

In this section, the use of default CoAP for Internet cloud services is compared to the use of CoCoA. It is evaluated if the use of advanced CC mechanisms improves the quality of service in terms of higher throughput and faster processing of requests by the network. In an experimental setup, Cf-clients try to access information stored on constrained devices in an LLN over the Internet. Three different scenarios for the evaluations that differ in the amount of generated traffic and in the interaction patterns between the Cf-clients and the LLN are defined. In the following the testbed and the general experiment setup are introduced. Then the details and results of the different scenarios are presented.

6.3.1 Testbed Setup: FlockLab Sensor Grid at the ETH (Zürich)

The FlockLab testbed is an in-/outdoor sensor testbed located in a university building of the ETH in Zürich. It consists of 30 TelosB nodes that are spread irregularly across two building floors. The official FlockLab website [3] provides a map that shows the distribution of the nodes across two floors of the office building (Fig. 6.3). Please note that the website containing this information can only be accessed with a valid account.

Each node can be accessed via the serial port, enabling remote programming and the interchange of data such as logs or other output of the nodes. The serial port at the same time serves as power source of the nodes. The FlockLab provides a front-end to the users of the testbed so that tests can be scheduled and planned over a web-interface

[3] Taken from https://www.flocklab.ethz.ch/user/testbedstatus.php

Figure 6.3: Map of the location of the 30 TelosB motes in the FlockLab, spread across two floors.

or via scripts. The results of a test run are stored on a server and can be accessed at any time for their evaluation.

There is no official characterization of the link qualities and studies of interference for the FlockLab. The experiments carried out in the FlockLab in the context of this thesis have revealed that there exists external network interference that varies with the time of the day. The transmission power of the TelosB motes for all the experiments is set to the default value of 0 dBm (maximum transmission power).

6.3.2 Experiment Setup

The experiment setup can be divided into two parts: the client side and the server side. On the client side, a PC to run multiple instances of the Cf client is set up to represent cloud services that access Web resources on sensor nodes. For the server side, the FlockLab testbed [119] is used with CoAP servers on 30 Tmote Sky motes [114], which provide a test resource that allows GET requests. The motes are programmed with the full Contiki 6LoWPAN communication stack. This includes RPL [26] and Er-CoAP [116]. One of the central motes in the FlockLab is set up as border router to connect the LLN with the Internet. For the evaluations, furthermore two link layer configurations are compared: RDC with ContikiMAC [1] and no RDC.

On the client side, four different CoAP CC schemes are evaluated as listed in Table 6.1. The first one uses default CoAP, that is, the CoCoA layer for Californium is

Table 6.1: CoAP CC schemes.

CC used in clients	RTO base value	NSTART
Default CoAP	2 s	1
Default CoAP$_B$ ($\frac{\text{RTO}_{init}}{2}$)	1 s	1
CoCoA	2 s	1
CoCoA$_4$	2 s	4

Table 6.2: Results for the Baseline scenario (No RDC).

Config	Throughput (req./s)	Exch. duration	# of retries
CoAP	0.67	1.49 s	0.40
CoAP$_B$	0.88	1.13 s	0.52
CoCoA	1.18	0.84 s	**0.38**
CoCoA$_4$	**1.42**	**0.70** s	0.56

disabled. The second one explores how the performance of default CoAP is affected when it uses less conservative RTO values: the initial RTO is reduced from 2 s to 1 s (CoAP$_B$), expecting CoAP to react faster to packet losses not originating from congestion but from lossy links. The latter two configurations use the CoCoA layer implemented for Cf, while the evaluations are carried out for the cases NSTART=1 (CoCoA) and NSTART=4 (CoCoA$_4$). With NSTART=4 tests, it is to be determined if relaxing the restriction of only one open request to an endpoint at a time increases the performance, while maintaining network stability when applying advanced CC mechanisms.

All test runs for each method are repeated ten times with a duration of 15 minutes each. In the following, three scenarios for the interaction between the cloud services and the LLN are defined and the results of the measurements are presented.

6.3.3 Scenario and Experiments Results

In the following, the evaluation scenarios and the results obtained in these scenarios are presented.

6.3.3.1 Baseline Scenario: 1-to-1

In the baseline scenario, the performance is measured under simplified network conditions: Apart from the exchange of messages between one client and one server, no further CoAP traffic is generated. It is observed how long it takes for a single client to exchange 50 CON-ACK pairs (request and piggybacked response) with a single CoAP server in the LLN. As metrics, the average throughput (# of requests processed per second), the average exchange duration (time it takes to obtain an ACK to a CON

Table 6.3: Results for the Baseline scenario (ContikiMAC).

Config	Throughput (req./s)	Exch. duration	# of retries
CoAP	0.56	1.76 s	0.21
$CoAP_B$	0.79	1.26 s	0.35
CoCoA	0.89	1.12 s	**0.07**
$CoCoA_4$	**1.18**	**0.84 s**	0.35

Table 6.4: Results for the Many-to-Many scenario (no RDC).

Config	Throughput (req./s)	Exch. duration	# of retries
CoAP	4.23	2.32 s	**0.28**
$CoAP_B$	4.05	2.15 s	1.08
CoCoA	**5.34**	1.94 s	0.43
$CoCoA_4$	4.59	**1.65** s	0.67

request), and the average amount of retries used per original CoAP request are given. This experiment is repeated for every mote in the LLN separately and determines how well CoAP and CoCoA perform when there is no congestion within the LLN and at the border router.

Table 6.2 shows the metrics when no RDC is configured. The average throughput achieved by clients using CoCoA and $CoCoA_4$ is higher than the throughput achieved with default CoAP. The adaptive RTO mechanisms ensure that advanced CC mechanisms use the available bandwidth more efficiently than default CoAP. The peak performance is achieved with NSTART=4, which doubles the average throughput. The results for $CoAP_B$ reveal that reducing the fix interval of RTO_{init} can help to improve the throughput as well in this scenario. Following the tendency of the throughput, the average exchange duration is the lowest for $CoCoA_4$ and the highest for CoAP.

With ContikiMAC (Table 6.3), the same tendencies are observed, however, the average throughput decreases for all CC schemes, since the duty cycling introduces larger delays to communications within the LLN. Independent from the RDC, the conclusion is drawn that in networks with low or no congestion, the use of CoCoA with NSTART>1 is recommended, as it leads to a significant increase in throughput.

6.3.3.2 Many-to-Many Scenario

In the many-to-many scenario, it is evaluated how the CC mechanisms perform in environments that suffer from heavy congestion. During the setup phase of the experiment, a separate client to each of the CoAP servers in the LLN is assigned. As soon as the test starts, all clients continuously exchange CON-ACK pairs with their associated CoAP

Table 6.5: Results for the Many-to-Many scenario. (ContikiMAC).

Config	Throughput (req./s)	Exch. duration	# of retries
CoAP	1.60	5.72 s	1.81
CoAP_B	1.45	6.06 s	2.33
CoCoA	1.79	**4.16** s	**1.69**
CoCoA_4	**1.90**	5.12 s	2.16

servers until the test finishes. The continuous requests cause significant congestion at the border router bottleneck and inside the LLN.

Tables 6.4 and 6.5 show that the highest overall throughput is achieved with clients that use CoCoA. With NSTART=1, each client adapts the RTO to a saturated and highly congested network and a higher overall throughput is achieved when compared to default CoAP. Allowing more parallel exchanges with NSTART=4 does not increase the performance noticeably. In a situation where the LLN and the border router are highly congested, it is difficult to adjust the RTO timers adequately when more CoAP messages are transmitted in parallel by the clients. If the clients use CoAP_B, a performance loss is observed, since the more aggressive yet static RTO setting does not adapt to the heavy congestion of the network. The aggressive behavior also results in using the highest number of retries, exceeding an average of one retry per CoAP packet sent. It is also important to note that the clients running CoCoA have the shortest exchange durations for both configurations, with and without RDC.

The results of the many-to-many case show that the use of adaptive RTOs has a high impact on the performance. Figure 6.4 shows the typical distribution of RTO values observed during the test runs in the many-to-many scenario. The static RTO intervals of default CoAP that result from the BEB for retransmissions are visible for the CoAP and CoAP_B cases. Since CoCoA dynamically adjusts the RTO, it uses a wider range of RTO values: the boxplots for CoCoA and CoCoA_4 show lower minimums and a wider spectrum of outliers.

In the cross traffic burst scenario, it is observed how clients that continuously exchange data with an LLN react to a state of sudden congestion in the LLN. For this experiment, two groups of clients are distinguished: The first group, consisting of four clients, continuously sends requests to four randomly picked CoAP servers in the LLN throughout the whole test duration. The second group consists of up to 25 clients, one for every other CoAP server of the LLN, and is set to initiate half-way through the test (at the instant of $t = 450\ s$). At the point the second group of Cf-clients initiates, they begin the exchange of a limited number of requests (50) between them and the CoAP servers. The sudden appearance of these CoAP messages causes a peak in the network congestion. It is determined how this traffic burst affects the performance of the different CC schemes by measuring the overall throughput and looking at the time it takes to transmit the burst of traffic.

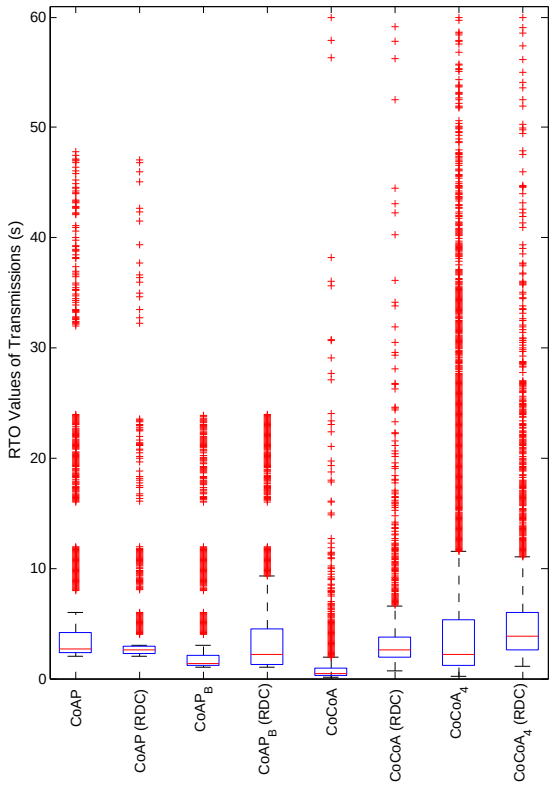


Figure 6.4: Boxplots for all CC schemes (and RDC configurations) showing the RTO values observed during test runs in the many-to-many scenario with median (lines within the boxes), the first and third quartiles (bottom and tops of boxes), 1.5 times the interquartile range of the first and third quartiles (whiskers) and outliers (crosses).

Figures 6.5 and 6.6 show the observed RTO values used for (re)transmissions as well as the cumulative distribution functions (CDFs) for the constant traffic and interfering burst. In both figures, the 100% mark of the constant traffic CDF refers to the amount of completed requests with CoCoA during the experiment.

Up to the burst at 450 s, the network only suffers from a low degree of congestion. During this phase, the clients are able to achieve a high throughput, since the performance is similar to the one observed in the baseline scenario, where no congestion was present. CoCoA therefore is able to transmit more messages than default CoAP in the same time interval. For CoAP$_B$, and CoCoA$_4$ (not visualized) this is the case as well. A change occurs as soon as the additional traffic is introduced in the network, causing a high degree of congestion for a limited time. Now the constant and burst traffic clients are contending. The CDF of the successfully transmitted packets of the burst traffic shows that default CoAP needs more time to process the burst traffic requests. Conversely, CoCoA processes the messages of the burst in a shorter time span and returns to the initial network state much faster, thanks to the adaptive RTO that adjusts to

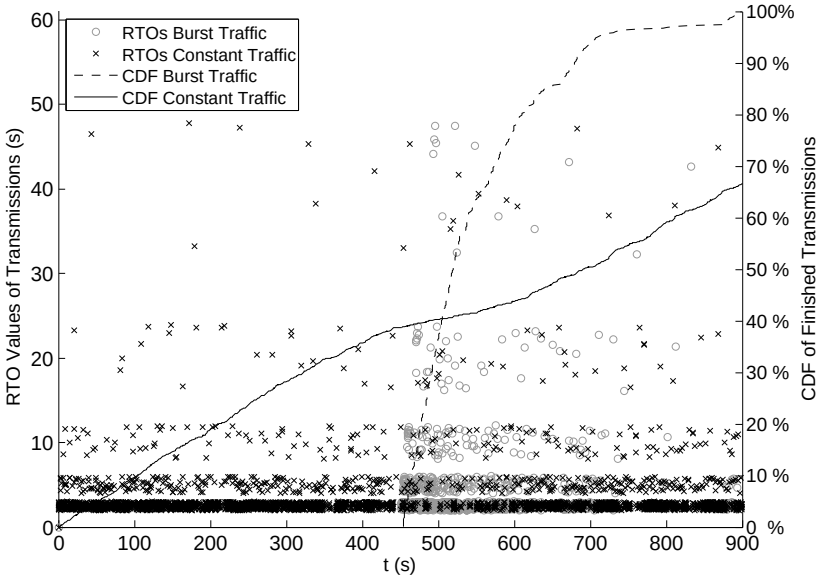


Figure 6.5: RTO values for (re)transmissions and CDF of successful requests for constant and burst traffic during a test run with clients using CoAP with no RDC.

the sudden traffic burst.

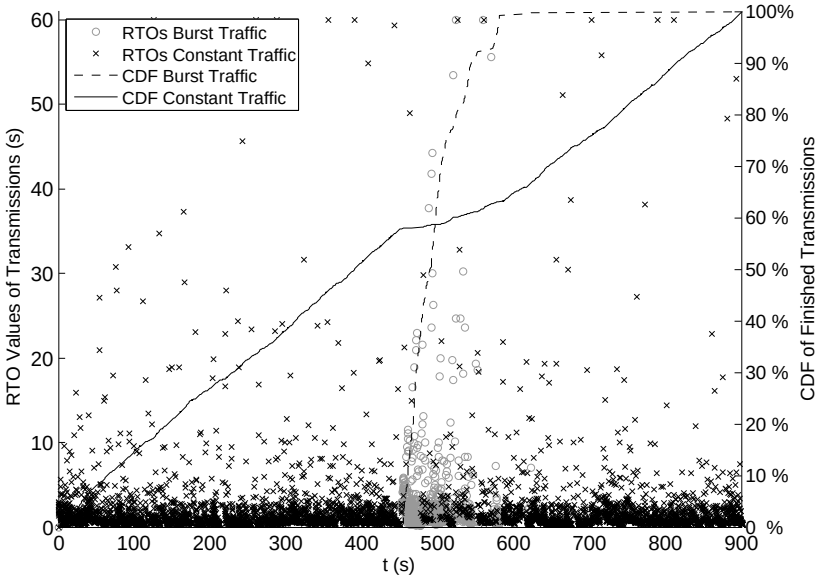


Figure 6.6: RTO values for (re)transmissions and CDF of successful requests for constant and burst traffic during a test run with clients using CoCoA with no RDC.

Table 6.6: Results for the Burst scenario (No RDC).

Config	Throughput (req./s)	Exch. duration	# of retries
CoAP	2.65	1.86 s	0.22
$CoAP_B$	2.98	2.21 s	0.37
CoCoA	**3.98**	**0.84 s**	**0.21**
$CoCoA_4$	2.48	1.43 s	0.52

Table 6.7: Results for the Burst scenario (ContikiMAC).

Config	Throughput (req./s)	Exch. duration	# of retries
CoAP	1.58	2.21 s	**0.35**
$CoAP_B$	1.32	2.52 s	0.98
CoCoA	1.69	2.09 s	0.39
$CoCoA_4$	**1.95**	**1.94** s	0.79

6.3.3.3 Cross Traffic Burst Scenario

Tables 6.6 and 6.7 show that CoCoA performs best with and without RDC. Since with $CoCoA_4$ every client may send up to 4 requests in parallel, the network is driven into congestion in the pre-burst phase in the case where no RDC is applied. Since the calculated RTOs are small, clients react with fast retransmissions to congestion losses or losses from lossy links, which increases congestion further and can lead to an important increase of the RTOs of subsequent retries with a VBF of 3. Both, fast retransmissions and large backoffs after the LLN recovers from congestion lead to a poor overall throughput. On the other hand, since with ContikiMAC the delays in the LLN are larger, the calculated RTOs are larger and fast retransmissions are avoided. This leads to a more stable behaviour and a better overall performance of $CoCoA_4$ when ContikiMAC is enabled. $CoAP_B$ cannot adapt to the network state and the reduced initial RTO interval is prone to cause congestion. Only when RDC is disabled and during the congestion-free pre-burst phase, $CoAP_B$ is able to achieve a higher trouhput than CoAP. In all other situations (with ContikiMAC, burst phase) $CoAP_B$ performs worst.

The generally short average exchange duration and low amount of retries can be explained by the fact that most of the measured values are obtained during the initial and end phases, where the degree of congestion is low and fewer retransmissions are needed.

6.4 Comparison of CoCoA with Alternative CC Mechanisms

In this section a comparative performance analysis of CoCoA and a variety of alternative algorithms including state-of-the-art mechanisms developed for TCP is performed, based on experiments carried out in real testbeds.

The first set of experiments is performed in the FlockLab testbed. The second set of experiments uses General Packet Radio Service (GPRS), which is a common M2M communication solution for IoT devices. Along with CoCoA, the potential contribution that state-of-art algorithms used in TCP can provide over default CoAP CC is assessed. The remainder of the section is organized as follows.

The state-of-the-art CC mechanisms along with the two testbeds used for comparative evaluations are detailed in Section 6.4.1. Results are elaborated by providing technical insights on the observed performances in Section 6.4.2.

6.4.1 Experimental Setup and Test Configuration

This section presents the testbeds, the CC mechanisms and the traffic scenarios used in the study to evaluate various CC mechanisms for CoAP.

6.4.1.1 Testbeds

Typical communication technologies used in IoT communications include GPRS and IEEE 802.15.4. These communication technologies have different bandwidth and delay characteristics. Moreover, GPRS involves a single wireless hop, whereas IEEE 802.15.4 networks are often deployed as multihop networks.

GPRS is a common M2M technology that allows a flexible network setup in areas where no connection to the Internet is available or when it is either too expensive or too complex to set up. IEEE 802.15.4 targets low-power communication and is a common communication solution employed by many IoT standards, including ZigBee and 6LoWPAN.

The GPRS and IEEE 802.15.4 communication technologies are used in this section for evaluations in two experimental setups that are also using different hardware to run CoAP servers and clients, as depicted in Fig. 6.7.

In the first setup, a laptop running CoAP clients uses a GPRS modem to connect to the Internet, from where packets are routed towards a PC running a CoAP server. When compared to a wired connection, much larger RTTs and a much stronger RTT jitter are observed over the GPRS link, as well as a higher chance for packet losses. A rather limited bandwidth of approximately 15 kbit/s and 40 kbit/s is observed in the uplink and downlink, respectively, that depends on the capacities assigned by the mobile operator. In this setup, both the CoAP clients and server are running the Java Cf CoAP implementation [117]. The alternative CC mechanisms considered in

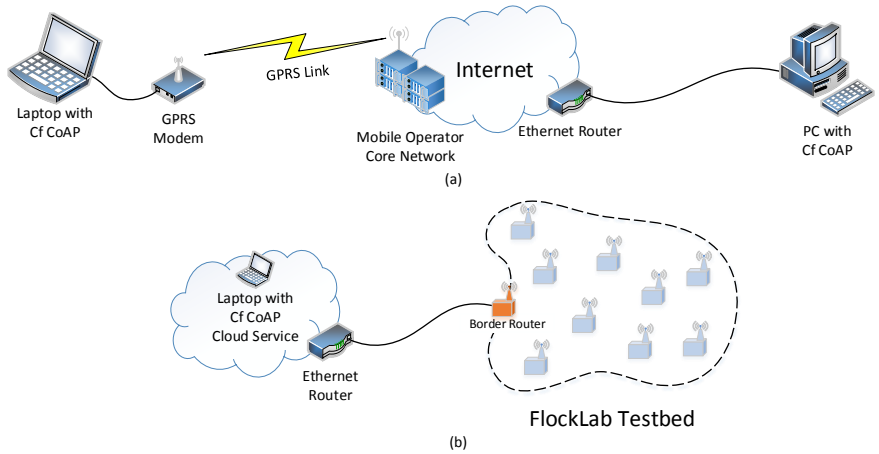


Figure 6.7: Illustration of (a) the GPRS test setup and (b) the FlockLab test setup.

these evaluations are implemented and publicly available as an optional CC layer in Cf [Bet14].

In the second setup, the interaction between a cloud service and a multihop, low-power wireless network is analyzed. CoAP clients running in the cloud service are connected via a LAN and a border router to the FlockLab running CoAP servers (see Section 6.3.1 for details on the FlockLab). The FlockLab motes run ContikiMAC [1] with the RDC setting enabled, which is an important requirement for real world IoT deployments in order to save energy. On the client side, Cf is used as CoAP implementation, while the server motes in the FlockLab run the full IP-based Contiki stack for constrained devices, including the Er CoAP implementation [120].

When compared to the GPRS setup, the FlockLab setup imposes additional challenges for the CC mechanisms. In this IEEE 802.15.4 multihop network a considerable packet loss rate and RTT variance are observed due to different route lengths and the operational mode of the radio, also observed in [120]. Packet losses in this setup mostly emerge from lossy links and from packet drops due to full buffers in the border router and relay nodes close to it. The border router is the FlockLab node that provides Internet connectivity to the Flocklab motes. Therefore, the network area in its proximity may easily become congested.

6.4.1.2 Traffic Scenarios

For both network setups (GPRS and FlockLab), two traffic scenarios are defined to explore the effect of different CC mechanisms on the performance of CoAP communications:

1. *Continuous Traffic*: In this scenario CoAP clients send CON requests to a CoAP server. As soon as a client receives a reply from the server, the client immediately

sends another CON request. Sending messages back-to-back by many clients simultaneously can create congestion and it allows to determine how efficiently the different CC mechanisms work. The number of clients that interact with servers is varied from 10 to 40 (in steps of 10) in order to achieve different degrees of bandwidth usage and congestion. In the GPRS setup, one server is running on the destination device and the evaluations are carried out for a varying number of clients. In the FlockLab setup, one client is assigned to each of the CoAP server motes in the testbed. A continuous traffic test lasts 180 s.

2. *Burst Traffic*: This scenario starts with a low congestion level, where 10 clients (GPRS) or 5 clients (FlockLab) generate continuous traffic of back-to-back CON requests. Then, a burst of traffic is introduced into the channel by a new group of clients that send a total of 50 (GPRS) or 25 (FlockLab) back-to-back CON requests to the servers. The burst of messages causes a peak of congestion. For the GPRS setup, the number of clients that generate burst traffic is varied. In the FlockLab setup, for each mote that is not a destination of continuous traffic, a client is created that generates burst traffic.

In average, tests are repeated more than 16 times for each specific configuration with an accumulated test duration of over 100 hours for both network setups.

6.4.1.3 CC Mechanisms Evaluated

In order to provide a wider understanding of CC for the IoT, besides comparing default CoAP with CoCoA, other RTO calculation algorithms are considered in the evaluations. The Linux TCP RTO (Linux RTO) estimator [121], the peak-hopper TCP RTO estimator (PH-RTO) [122] and also a CoCoA variant that only uses the strong RTO estimator (CoCoA-S) are evaluated, in comparison to default CoAP and to each other.

The Linux RTO algorithm adds two mechanisms to the basic TCP RTO algorithm. First, in contrast to the original algorithm, when a new RTT measurement is smaller than the previously gathered RTT information, the RTO is not increased, avoiding possible peaks in the RTO value when the channel seems to improve. Second, the Linux RTO algorithm avoids the RTO estimator to converge into a RTT value after repeatedly measuring constant RTT values [121], which could lead to spurious retransmissions.

The PH-RTO algorithm modifies the original algorithm in such a way that it reacts to sudden increases of the RTT with a quick increase of the RTO using a short term RTT history, which then decays over time towards the value of a long term RTT history. PH-RTO intends to avoid spurious retransmissions by using the long term history, when the channel suffers from sudden delays. RTO dithering is not defined for the Linux and PH-RTO algorithms, which were not designed for IoT scenarios.

Above these two state-of-the-art algorithms used in TCP, a minimalist 'Basic-RTO' (B-RTO) estimator is included in the evaluations as a benchmark and to understand the potential contribution of a simple method to advanced CC solutions. This estimator

always sets the initial RTO for a transmission to a random value between the previously measured RTT and 1.5 times the RTT value. The B-RTO algorithm can use weak RTT measurements. An overview of the features and settings of all six analyzed CC mechanisms is given in Table 6.8.

6.4.1.4 Performance Metrics

In the continuous traffic scenario, the overall throughput as successfully finished transactions per second is chosen as the performance metric. The throughput metric merges several performance aspects, such as delay and PDR, into one single value. A good performing CC mechanism is able to regulate the traffic in such a way that, as the traffic load increases congestion collapse is avoided, while achieving a high throughput.

In the burst traffic scenario, the settling time (ST) of the different CC approaches is analyzed. The ST is defined to be the time in seconds it takes for the clients to finish at least 80% of the transactions that are introduced as part of the burst traffic. The ST serves as an indicator of how fast the network is able to recover to a network state similar to the one before the burst happened. For the operation of IoT networks the ST is an important metric, since bursts of traffic are expected when CoAP transactions are event-based or transmissions from various senders are synchronized.

Furthermore, the behavior of the CC mechanisms is analyzed regarding their fairness in the FlockLab setup, which is challenging given the different path lengths and the tree-like topology of the scenario.

6.4.2 Experiment Results

In this section the evaluation results are provided of the CC mechanisms presented in the setups and the traffic scenarios described in the previous section.

6.4.2.1 Throughput Results

Nearly all RTT-sensitive mechanisms are able to outperform default CoAP independently from the network setup in terms of throughput in the continuous traffic scenario (Fig. 6.8).

Table 6.8: Overview of the features and settings of the different CC mechanisms.

	Strong RTTs	Weak RTTs	Dithering	Backoff method	RTO aging	Use backed-off RTO	RAM usage per client	Main Goal
Default CoAP	No	No	Yes	BEB	No	No	2 Bytes	IoT Traffic
CoCoA	Yes	Yes	Yes	VBF	Yes	No	29 Bytes	IoT Traffic (adaptive)
CoCoA-S	Yes	No	Yes	VBF	Yes	No	19 Bytes	IoT Traffic (adaptive)
Basic RTO	Yes	Yes	Yes	BEB	No	Yes	2 Bytes	Minimalistic RTO Solution
Linux RTO	Yes	No	No	BEB	No	Yes	21 Bytes	TCP RTO Enhancement
PH-RTO	Yes	No	No	BEB	No	Yes	43 Bytes	TCP RTO Enhancement

Table 6.9: Comparison of the average RTT and initial RTO values in milliseconds for different number of clients in the GPRS setup and the FlockLab setup.

	10 GPRS clients		20 GPRS clients		30 GPRS clients		40 GPRS clients		FlockLab	
	RTT	RTO	RTT	RTO	RTT	RTO	RTT	RTO	RTT	RTO
Default CoAP	-	2497	-	2499	-	2499	-	2506	-	2505
CoCoA	661	1505	1437	3379	1936	4119	2796	5431	1507	3710
CoCoA-S	625	1428	1275	2903	1880	4122	2795	4928	656	3227
B-RTO	1025	1152	1962	2198	2983	3272	4733	4441	3172	3266
Linux RTO	682	1325	1550	2801	1863	3345	2931	5797	598	4424
PH-RTO	746	1797	1835	3703	1827	4112	3194	6213	625	4796

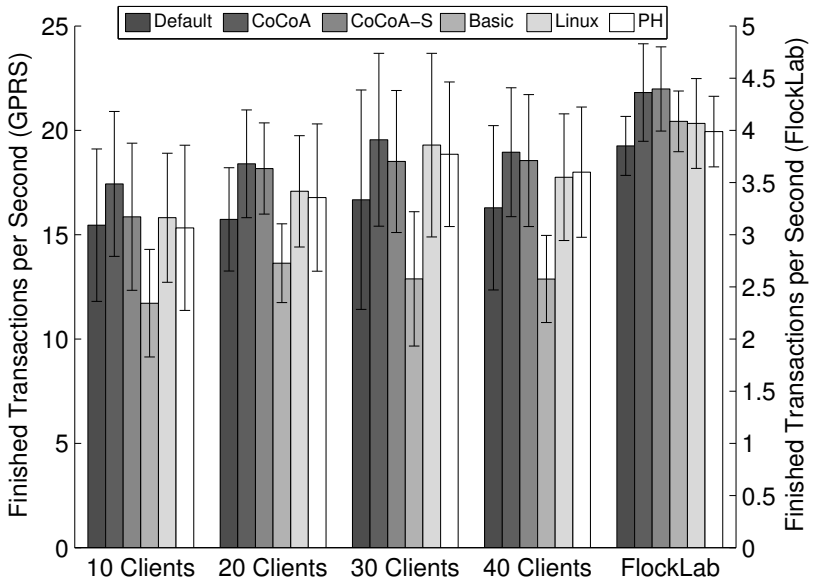


Figure 6.8: Average throughput with 95% confidence intervals achieved by the evaluated CC mechanisms in the GPRS and FlockLab setups.

In the GPRS setup, since the packet loss rate is low, the performance mainly depends on how the RTO algorithms adapt to the RTT of the channel. In this setup, RTT increases with the amount of active clients due to the delay introduced by the queuing of packets in the GPRS modem, which occurs since the overall generated data rate exceeds the uplink capacity. With the increase of the average RTT, the RTT-sensitive RTO algorithms increase their initial RTO values (Table 6.9). In the FlockLab setup, the average RTT is small, yet the average initial RTO for RTT-sensitive algorithms is larger than in the GPRS setup. This can be ascribed to a greater amount of weak RTTs that increase the RTO value and backed-off RTO values due to packet drops as a consequence of overflowing buffers in the areas where traffic mostly concentrates, which are those near to the border router.

Default CoAP underperforms independently from the setup (except in the FlockLab setup comparison with B-RTO) since it uses a fixed range of initial RTO values and does not adapt to the current RTT. If the real RTT is noticeably below the default RTO range, CoAP reacts slowly to losses. If the RTT lies in the RTO range or even exceeds it, spurious retransmissions are likely to happen, as indicated by the increasing percentage of retransmissions with the number of clients in GPRS (Fig. 6.9).

Independently from the number of clients, CoCoA achieves the highest throughput in the GPRS setup. In the same setup, CoCoA-S does not perform as well as CoCoA since it only allows strong RTT measurements, generally resulting in slightly lower RTO values and thus increasing the probability of spurious retransmissions (see Fig. 6.9). In the FlockLab setup, CoCoA and CoCoA-S perform very similarly. Their features allow

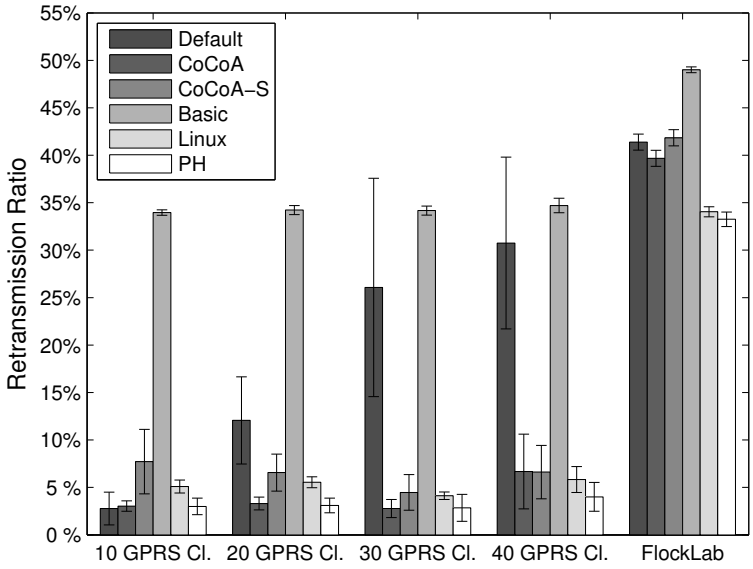


Figure 6.9: Average percentage of CoAP retransmissions over all CoAP transmissions observed for the evaluated CC mechanisms in the GPRS and FlockLab setups.

to benefit especially from links with good connections and small RTTs, but they also provide the necessary mechanisms to adapt the RTO even in a lossy network.

The throughput obtained with the B-RTO algorithm suffers noticeably due to its simplicity. If after measuring a small RTT, the RTT of the following transaction is larger, which is likely given the fluctuations of the RTTs in both network setups, the RTO timer will fire ahead of time with a high probability. In fact, B-RTO exhibits the highest retransmit ratio of all tested algorithms in all settings (Fig. 6.9). On the other hand, when a large RTT is measured, the next RTO used by the B-RTO algorithm can grow very large due to the use of the random multiplier, potentially leading to low throughput.

While the Linux RTO and PH-RTO algorithms perform better than default CoAP, they are not able to outperform CoCoA. The Linux RTO algorithm often calculates smaller RTO values, causing a higher amount of spurious retransmissions (Fig. 6.9), since contrarily to CoCoA it does not increase the RTO when the RTT decreases. The PH-RTO reacts to a sudden RTT increase with a peak in the RTO that then slowly decays in the following transactions. However, given the continuous RTT jittering that is characteristic for both network setups, a sudden RTO increase may not be necessary and can lead to larger idle times if subsequent packets are lost. A disadvantage of both Linux and PH-RTO algorithms is their limitation to using only strong RTTs, while weak RTT measurements could provide additional updates of the RTO estimator. Instead, the old RTO is maintained and backed-off, i.e., large RTO values are reused for new transactions. The reuse of backed-off RTO values by these two algorithms happens

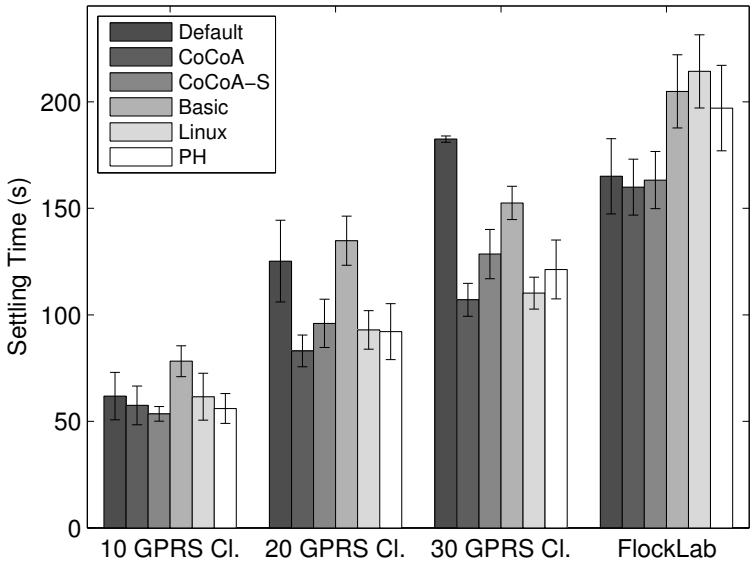


Figure 6.10: Average STs with 95% confidence intervals achieved by the different RTO mechanisms in the burst traffic scenario. For the GPRS experiment results, STs larger than 180 s are valuated as 180 s.

frequently in the FlockLab setup due to packet losses, leading to long idle times that reduce throughput.

6.4.2.2 Settling Time Results

Figure 6.10 shows the average STs obtained by the CC mechanisms in the burst traffic scenario. In the GPRS setup, results reveal that all RTT-sensitive mechanisms, except the B-RTO algorithm, are able to improve the performance of default CoAP when there is congestion. For 10 burst clients (i.e., low congestion), the different CC approaches perform similarly, except for the B-RTO algorithm since it tends to produce spurious retransmissions. When the amount of burst clients increases to 20 and 30, the RTT-sensitive mechanisms adjust their RTOs to higher RTT values. CoCoA does this most efficiently, followed by the Linux RTO, PH-RTO, and CoCoA-S algorithms, which perform slightly worse. While CoCoA-S tends to be too aggressive during the burst, the Linux RTO and PH-RTO need slightly longer time to adjust their RTO timers to the state of congestion, since they do not exploit weak RTT information. With few clients and short RTTs, the B-RTO algorithm deteriorates performance, being too aggressive. However, it gets more conservative with more clients and larger RTTs, eventually increasing performance when compared to default CoAP, which does not adapt the RTO timers at all.

In the FlockLab setup, CoCoA yields an improvement over default CoAP in terms of ST and it shows the most stable behavior with the narrowest confidence intervals.

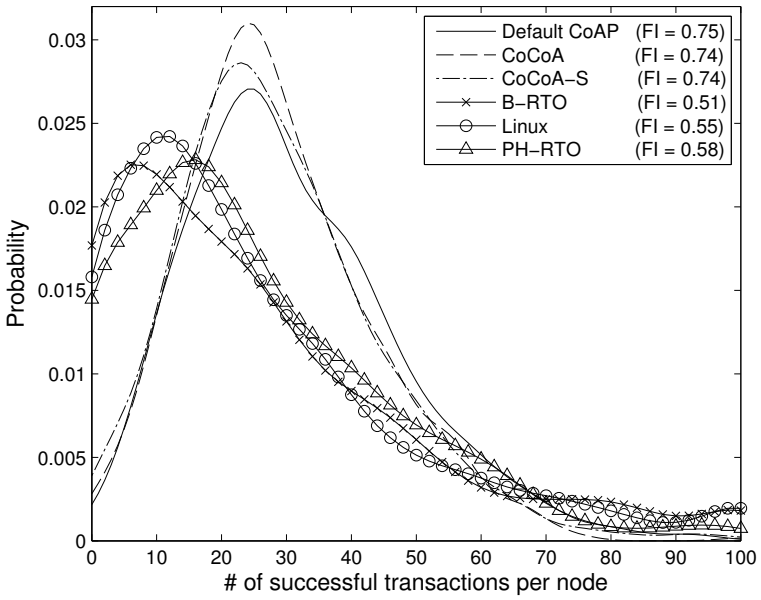


Figure 6.11: Probability Mass Function for the number of finished transactions per node and the Fairness Index (FI) achieved by each of the analyzed algorithms. For illustration purposes, more than 100 finished transactions per node are valuated as 100 finished transactions per node.

Over the course of the tests, CoCoA adapts client RTOs efficiently, so the continuous and burst traffic can be processed in parallel. CoCoA-S behaves similarly and also leads to a minor ST improvement. In contrast, the use of B-RTO, Linux RTO, and PH-RTO lead to larger STs in average. For these algorithms we observe a prevalent behavior of the continuous background traffic that hampers the sudden burst from being processed quickly and abates slowly since few RTT measurements can be made because of packet losses. This behavior has a higher impact when the continuous background traffic is directed towards endpoints with low RTT connections. Hence, fairness between the endpoints is another issue encountered for these three methods.

6.4.2.3 Fairness Evaluations

In terms of fairness, important differences are observed between the different CC algorithms in the FlockLab setup. Figure 6.11 shows the Probability Mass Function (PMF) for the number of finished transactions per destination node for each CC algorithm, measured during the continuous traffic experiments along with the Jain's Fairness Index (FI) [123]. The FI ranges between 0 and 1, where a higher FI indicates a higher fairness.

As seen in the figure, with Linux RTO, PH-RTO and B-RTO, a small group of nodes are served with a very high number of transactions (e.g. more than 80), whereas a large

number of nodes obtain a very low number of finished transactions (e.g. below 10). This results in much lower FI values compared to those of CoAP, CoCoA and CoCoA-S. The reason is that former methods exploit connections with small HCs and small RTTs, setting their RTO to small values, behaving aggressively in case of message losses and increasing throughput. However, connections with multiple hops and larger RTTs do not achieve the same bandwidth fraction, since larger RTO values and consecutive backoffs to these RTO values are applied, reducing throughput.

While CoCoA and CoCoA-S adapt their RTO values as well, the VBF prevents fast retransmissions for good connections with small RTTs and slow retransmissions for bad connections with large RTTs. Moreover, the use of backed-off values when initiating new transmissions is avoided and the aging mechanism prevents from maintaining very small or very large RTO values in idle periods. Thus, as seen in Fig. 6.11, CoCoA(-S) does not sacrifice from fairness when compared to default CoAP, while achieving the performance improvements presented in the previous sections. Default CoAP behaves neutrally in terms of fairness, since its RTO computation algorithm is independent of the specific characteristics of the path between the two communicating endpoints.

6.4.2.4 Memory Footprint Considerations

There exists a trade-off between performance and memory footprint of the CC mechanisms. Improving the behavior of default CoAP CC requires one order of magnitude more memory consumption per CoAP client (Table 6.8). However, and despite the limitations of many IoT devices, the additional state required by an advanced CC mechanism is negligible compared with the state needed for other components in a CoAP implementation such as security support. In fact, Datagram Transport Layer Security (DTLS) is mandatory as per the CoAP specification, and it consumes around 2 kB of RAM [124].

6.5 Conclusions and Future Work

In the first part of this chapter, the details of the implementation and evaluation of CoCoA for the Cf CoAP framework as an optional CC layer are presented for the use of CoAP in Internet cloud services. Experiments are carried out with cloud services that communicate with CoAP servers on real sensor nodes in a testbed and it is analyzed how well the different CC schemes for CoAP perform.

Through the evaluations of three different traffic scenarios, it is determined that the improvements achieved by CoCoA are twofold: the amount of requests that can be processed in parallel increases and the time it takes for clients to complete their tasks decreases. The advanced CC mechanisms achieve this by calculating efficient RTO timers and adjusting the backoff behaviour dynamically. It is also shown that NSTART can be increased to higher values safely, even though it may not always deliver the best performance.

In the second part of this chapter, it is shown that the CoCoA proposal not only performs better than CoAP in almost any case, it also prevails against state-of-the-art CC mechanisms applied in TCP or against simpler solutions. Other approaches to CC for CoAP may be too simple (B-RTO) or do not adapt well to the peculiarities of IoT communications (Linux RTO, PH-RTO). In general CoCoA is capable of delivering higher throughput, shorter RTTs, and shorter STs. In contrast to other mechanisms, CoCoA does not sacrifice from fairness, showing an almost identic FI as neutral CoAP. Moreover, the improvements of CoCoA do not only restrict to IEEE 802.15.4 networks, but also apply to alternative communication technologies like GPRS, which is used in M2M communications.

It can be alluded that there is a need for advanced CC mechanisms like CoCoA to improve CoAP's performance while keeping it stable, even in use cases that have to deal with high degrees of congestion. The release of version 2 of the CoCoA draft [BBGD14], which includes most of the improvements developed in this thesis, has taken the draft a step closer towards its future release as an RFC. With the results presented in this chapter, the suitability of CoCoA as advanced CC mechanisms for CoAP is strongly fortified. In future work, the evaluations of CoCoA therefore should focus on other possible IoT network scenarios, for instance experiments in large scale networks with hundreds or thousands of nodes. Also, the evaluation of CC mechanisms for non-confirmable messages should be considered for scenarios where no end-to-end reliability is necessary.

7

Conclusions and Future Work

In the following, the findings of the investigations carried out in this thesis are recapitulated and final thoughts about the results of the evaluations are shared with the reader. Furthermore, possible future lines of investigation are proposed to extend the presented work and address evaluations that have not been covered so far.

7.1 Conclusions

This thesis develops methods to improve and to evaluate the end-to-end performance of low-power wireless networks. A substantial part of these methodologies involve the adjustment of commonly used parameters and mechanisms of communication protocol stacks for low-power wireless networks. Moreover this thesis contributes to the improvement of end-to-end performance in WSNs by proposing modifications of mechanisms and by designing novel ones. The improvements developed in this thesis have a considerable impact on the end-to-end performance of WSNs. These improvements reflect in a higher reliability, shorter response times, and a higher energy efficiency, to name some of the investigated performance metrics.

The investigations in this thesis cover low-power wireless networks with different application use cases, different network environments, different network scopes (ranging from local WPANs to IoT networks), as well as different communication technologies that are used for end-to-end communications. The overarching experimental and simulative evaluations carried out focus on multiple communication protocol stack layers, including the Application, Transport, Network, and Medium Access layers, providing solutions that lead to performance improvements for each of them. The proposed solutions are generally applicable to networks using the de-facto standard IEEE 802.15.4 for low-power wireless networks, but can also be applied to networks used in specific domains, as shown in the holistic evaluation of ZigBee WHANs. On the other hand, solutions for a very wide application and network scope are developed in the context of the evaluations of performance improvements for IoT networks.

In Chapter 3, it has been shown that the performance achieved in IEEE 802.15.4-based networks when using the default communication protocol stack settings can be improved by adjusting the communication stack parameter settings of several layers. Trivial protocol stack parameters and mechanism settings, such as the number of MAC layer retries, the routing metric applied at the NWK layer, and the congestion window limit of the transport protocol layer prove to have an important impact on the end-to-end performance. Evaluations have been carried out in the Castelldefels indoor testbed composed of 60 TelosB motes, revealing that the default protocol settings do not yield the best performance, often underperforming in terms of PDR and delays when compared to the performance achieved with improved settings that are determined over the course of the evaluations.

The focus of the investigations tilts to WHANs in Chapter 4, where solutions are found to improve the performance in ZigBee WHANs. Via a thorough experimental evaluation of parameter settings and mechanisms from different communication protocol stack layers in a complex HA setup with different node roles and traffic patterns, two improved configurations of the stack are derived. The first of these two configurations is fully compliant with the ZigBee standard, respecting valid ranges of parameter settings and applying the ZigBee protocol mechanisms as defined in the ZigBee specification. The second configuration trespasses the limitations imposed by the ZigBee specification, exceeding the boundaries of the parameters ranges and replacing the default mechanisms with new ones, such as the RTO calculations applied as part of the end-to-end reliability mechanism. Both proposed configurations result in large improvements of performance for metrics that are very important for WHANs such as PDR, end-to-end delays, and energy consumption: With the compliant configuration, an increase of the PDR of up to 31.5%, a delay reduction of up to 60.4%, and an energy efficiency improvement of up to 46.5% can be achieved, while the unrestricted configuration improves the performance to 33.6% (PDR), 66.6% (delay), and 48.7% (energy efficiency), respectively.

In Chapter 5 the scope of investigated networks is widened significantly with the evaluation of CC mechanisms for CoAP, the application layer protocol designed by the IETF for IoT communications. CC is crucial for communications in low-power wireless networks, given the important hardware limitations of constrained devices, such as memory and processing capacities, as well as the limitations of low-power wireless communication technologies. The conservative CC mechanism defined in the CoAP base specification is designed to assure a safe operation of CoAP in the Internet. However, CoAP's CC mechanism lacks dynamicity, completely condoning the communication channel conditions between two communicating devices. This can lead to important drops of the end-to-end performance under certain network conditions.

CoCoA, an advanced CC mechanism for CoAP tries to address the lack of adaptiveness of the default CC mechanism applied by CoAP. Performance evaluations carried out in the Cooja simulator for different network topologies and traffic patterns show that CoCoA can lead to important performance improvements for networks of

constrained devices when compared to default CoAP. Yet, over the course of the evaluations, several shortcomings of the first version of CoCoA are identified that can result in CoCoA underperforming default CoAP. Further investigations are carried out and the shortcomings are addressed by modifying CoCoA's core mechanisms and by adding novel mechanisms to the repertoire. With the redesigned version of CoCoA, the flaws of its initial version are overcome and further performance improvements are observed in Cooja network simulations of IoT application scenarios. Moreover, it is shown that in none of the analyzed scenarios, CoCoA underperforms the default CC mechanism, which is an important prerequisite for the design of an advanced CC mechanism for CoAP.

The evaluations carried out in Chapter 6 validate CoCoA being capable of improving the performance in real IoT setups and that it prevails other state-of-the-art CC mechanisms, like those applied in TCP. In the first part of these validations, the performance of default CoAP and CoCoA is compared in real testbed evaluations, where nodes in the in-/outdoor IEEE 802.15.4 FlockLab testbed located at the ETH in Zürich communicate with an external Internet cloud server. The evaluations reveal that CoCoA is not only able to improve performance noticeably in most of the scenarios, but also that its mechanisms allow a sage loosening of the limitations of parallel transactions towards one destination dictated by the NSTART parameter. In the second part of the validations, the evaluations of CoAP CC in the FlockLab are extended and complemented with experiments in a GPRS setup, where devices using CoAP communicate over a GPRS link. In these evaluations it is shown that CoCoA adapts better to the network conditions of IoT scenarios in comparison with simpler CC mechanisms or those applied in TCP-like protocols. Moreover, the exhaustive experimental evaluations carried out in the FlockLab testbed and a GPRS network setup reveal that CoCoA also achieves a high degree of fairness, further promoting CoCoA as a suitable candidate for an advanced CC mechanism for CoAP.

This thesis therefore presents a wide-scoped investigation of performance improvements for low-power wireless communications, involving exhaustive experimental and simulation-based evaluations. The results of this thesis have been published in international conferences, scientific journals, and have led to the co-authorship of an IETF Internet-Draft. Further, implementations of CoCoA for constrained and unconstrained devices have been publicly released to facilitate further investigations on the improvement of CoCoA by the scientific community.

7.2 Future Work

The results presented in this thesis have been obtained through ample experimental and simulative evaluations of low-power wireless networks with up to 60 nodes and different communication technologies. Given the increasing relevance and the wide application spectrum of other technologies that have not been covered in this thesis, like BLE and Z-Wave, investigations should be carried out to determine the performance of networks

based on these technologies and whether there are ways to improve their performance. For such evaluations, the investigation methodologies introduced in this thesis provide efficient ways to measure and improve the end-to-end performance. Further, the evaluations of low-power wireless networks should be extended to large-scale networks with hundreds or even thousands of constrained devices to validate and strengthen the findings of this thesis for network sizes typical of the IoT. Such large-scale network evaluations should extend the work on the evaluations for CC mechanisms for CoAP, but should include other typical protocols or mechanisms of IoT communication protocol stacks.

Apart from carrying over the evaluations to large-scale networks, it also should be investigated how massive amounts of traffic with up to hundred of thousands of outgoing or incoming requests per second are handled by CoCoA when generated by or destined to certain clients or servers, respectively. In close relation to these evaluations, an aggregate traffic control mechanism should be evaluated, applying CC mechanisms not only on a per destination endpoint basis, but on the entireness of the traffic generated by one device. The analysis of such a mechanism should involve the evaluation of the width of the scope for destination endpoints that is applied for communications relying on CoCoA. In the evaluations carried out in this thesis, a single IP address is considered to be a destination endpoint. Yet, the scope could be narrowed, by defining a destination endpoint to be a single UDP port of an IP address or it could be widened, by defining a destination endpoint to be a whole subnet of devices that use the same IP prefix.

The evaluations of CoCoA carried out so far have proven that when applied in IoT communications its current design results in noticeable performance improvements. To distinguish itself from the basic CC mechanism implemented by CoAP, the CoCoA draft defines several advanced mechanisms along with a set of static parameter settings, such as the weighting of the strong and weak RTO estimators for updates on the overall RTO or the VBFs. While the chosen parameter settings have shown to deliver a good performance in the analyzed network scenarios, no in-depth evaluations have been carried out to determine how alternative settings of these parameters can affect the end-to-end performance. Thus, future research should focus on evaluating different parameter values and it should be determined, whether the use of dynamic values that adapt to the network conditions might improve performance and adaptiveness of CoCoA further.

Moreover, the investigations so far have focused solely on reliable CoAP communications. Yet, the CoCoA draft also details rules for non-reliable CoAP communications, i.e., the exchange of NON messages. The investigation of advanced CC mechanisms for such communications is important for certain types of CoAP communications, for example when using the observe option, where the use of NON messages is common.

Contributions

[BBGD14] C. Bormann, A. Betzler, C. Gomez, and I. Demirkol. CoAP Simple Congestion Control/Advanced (work in progress), July 2014.

[Bet14] A. Betzler. CoCoA Implementation for Californium. https://github.com/eclipse/californium/tree/congestion-control, November 2014.

[Bet15] A. Betzler. CoCoA Implementation for Erbium (to be released in Github). https://sites.google.com/site/augustbetzler/cocoa-er-code, April 2015.

[BGDK14] A. Betzler, C. Gomez, I. Demirkol, and M. Kovatsch. Congestion Control for Reliable CoAP Cloud Services. 2014.

[BGDP12] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells. Should we use the default protocol settings for networks of constrained devices? In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2012 10th International Symposium on*, pages 305 –310, may 2012.

[BGDP13] August Betzler, Carles Gomez, Ilker Demirkol, and Josep Paradells. Congestion Control in Reliable CoAP Communication. In *Proceedings of the 16th ACM International Conference on Modeling, Analysis & Simulation of Wireless and Mobile Systems*, MSWiM '13, pages 365–372, New York, NY, USA, 2013. ACM.

[BGDP14] August Betzler, Carles Gomez, Ilker Demirkol, and Josep Paradells. A Holistic Approach to ZigBee Performance Enhancement for Home Automation Networks. *Sensors*, 14(8):14932–14970, 2014.

[BGDPed] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells. CoCoA+: An Advanced Congestion Control Mechanism for CoAP. *Ad Hoc Networks*, 2015 (accepted to be published).

[BGDPew] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells. Congestion Control for Billions of Nodes in the Internet of Things. *IEEE Communications Magazine*, 2015 (under review).

References

[1] Adam Dunkels. **The ContikiMAC Radio Duty Cycling Protocol**. Technical Report T2011:13, Swedish Institute of Computer Science, 2011.

[2] Freescale Semiconductor. **Freescale - Beestack**. http://www.freescale.com/, 2013. (accessed on 24 March 2015).

[3] C. Gomez, P. Salvatella, O. Alonso, and J. Paradells. **Adapting AODV for IEEE 802.15.4 Mesh Sensor Networks: Theoretical Discussion and Performance Evaluation in a Real Environment**. In *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, WOWMOM '06, pages 159–170, Washington, DC, USA, 2006. IEEE Computer Society.

[4] Matthias Kovatsch, Martin Lanter, and Zach Shelby. **Californium: Scalable Cloud Services for the Internet of Things with CoAP**. In *Proceedings of the 4th International Conference on the Internet of Things (IoT 2014)*, 2014.

[5] IEEE. **802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)**, September 2006.

[6] JP Vasseur. **Terms Used in Routing for Low-Power and Lossy Networks (RFC 7102)**. https://tools.ietf.org/html/rfc7102, January 2014.

[7] ZigBee Alliance. **ZigBee Specification, Document 053474r17**, Jan. 2008.

[8] ZigBee Alliance. **ZigBee Market Leadership**. http://old.zigbee.org/About/AboutTechnology/MarketLeadership.aspx, Jan. 2013. (accessed on 24 March 2015).

[9] S. Deering and R. Hinden. *RFC 2460 Internet Protocol, Version 6 (IPv6) Specification*. Internet Engineering Task Force, December 1998.

[10] Zach Shelby and Carsten Bormann. *6LoWPAN: The Wireless Embedded Internet*. Wiley Publishing, 2010.

[11] Gartner. **Forecast: The Internet of Things, Worldwide, 2013.** http://www.gartner.com/document/2625419?ref=QuickSearch&sthkw=G00259115, December 2013.

[12] ZigBee Alliance. **ZigBee**. http://www.zigbee.org, January 2015. (accessed on 24 March 2015).

[13] **IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications**. *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pages 1–2793, 2012.

[14] C. Bormann, M. Ersue, and a. Keranen. **Terminology for Constrained-Node Networks (RFC 7228)**. https://tools.ietf.org/html/rfc7228, May 2014.

[15] Z. Shelby, K. Hartke, and C. Bormann. **The Constrained Application Protocol (CoAP) (RFC 7252)**, June 2014.

[16] C.J. Fung and Y.E. Liu. **Lifetime Estimation of Large IEEE 802.15.4 Compliant Wireless Sensor Networks**. In *Modeling, Analysis and Simulation of Computers and Telecommunication Systems, 2008. MASCOTS 2008. IEEE International Symposium on*, pages 1 –4, sept. 2008.

[17] Frees. **Compact Integrated Antennas**. Technical Report AN2731, December 2012.

[18] H. Zimmermann. **OSI Reference Model–The ISO Model of Architecture for Open Systems Interconnection**. *Communications, IEEE Transactions on*, **28**(4):425–432, Apr 1980.

[19] Z-Wave Alliance. **Z-Wave**. http://www.z-wave.com/getting_started, 2012. (accessed on 24 March 2015).

[20] Bluetooth. **Specification of the Bluetooth System, Covered Core Package Version: 4.2**, June 2010.

[21] Texas Instruments. **Chipcon Products: CC2420 Datasheet: 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver**, March 2013.

[22] S.B. Eisenman and A.T. Campbell. **E-CSMA: Supporting Enhanced CSMA Performance in Experimental Sensor Networks Using Per-Neighbor Transmission Probability Thresholds**. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1208 –1216, may 2007.

[23] R.C. Carrano, D. Passos, L.C.S. Magalhaes, and C.V.N. Albuquerque. **Survey and Taxonomy of Duty Cycling Mechanisms in Wireless Sensor Networks**. *Communications Surveys Tutorials, IEEE*, **16**(1):181–194, First 2014.

[24] Z. Cao, C. Gomez, M. Kovatsch, H. Tian, and X. He. **Energy Efficient Implementation of IETF Constrained Protocol SuiteEnergy Efficient Implementation of IETF Constrained Protocol Suite**. https://datatracker.ietf.org/doc/draft-ietf-lwig-energy-efficient/, October 2014.

[25] J. Nieminen, T. Savolainen, M. Isomaki, Z. Shelby, and C. Gomez. **Transmission of IPv6 Packets over Bluetooth Low Energy (work in progress)**, February 2015.

[26] T. Winter, P. Thubert, J. Hui, P. Kelsey, P. Levis, K.P Pister, R. Struik, JP. Vasseur, and R. Alexander. **RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks**. Technical Report 6550, RFC Editor, Fremont, CA, USA, March 2012.

REFERENCES

[27] Joanna Kulik, Wendi Heinzelman, and Hari Balakrishnan. **Negotiation-based protocols for disseminating information in wireless sensor networks**. *Wirel. Netw.*, **8**(2/3):169–185, March 2002.

[28] Yong Yao and Johannes Gehrke. **The cougar approach to in-network query processing in sensor networks**. *SIGMOD Rec.*, **31**(3):9–18, September 2002.

[29] Maurice Chu, Horst Haussecker, Feng Zhao, 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**, 2002.

[30] 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.

[31] Kemal Akkaya and Mohamed Younis. **A survey on routing protocols for wireless sensor networks**. *Ad Hoc Networks*, **3**:325–349, 2005.

[32] Internet Engineering Task Force. *RFC 791 Internet Protocol - DARPA Inernet Programm, Protocol Specification*, September 1981.

[33] Rainer Baumann, Simon Heimlicher, Mario Strasser, and Andreas Weibel. **A Survey on Routing Metrics TIK Report 262**. Technical report, 2007.

[34] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. **A high-throughput path metric for multi-hop wireless routing**. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, MobiCom '03, pages 134–146, New York, NY, USA, 2003. ACM.

[35] C. Gomez, A. Boix, and J. Paradells. **Impact of LQI-based routing metrics on the performance of a one-to-one routing protocol for IEEE 802.15.4 multihop networks**. *EURASIP J. Wirel. Commun. Netw.*, **2010**:6:1–6:20, February 2010.

[36] C. Perkins, E. Belding-Royer, and S Das. **Ad hoc On-Demand Distance Vector (AODV) Routing (RFC 3561)**, July 2003.

[37] C. Perkins, S. Ratliff, J. Dowdell, and L. Steenbrink. **Dynamic MANET On-demand (AODVv2) Routing (work in progress)**, May 2015.

[38] T. Clausen, A. Colin de Verdiere, A. Niktash, Y. Igarashi, H. Satoh, U. Herberg, C. Lavenu, T. Lys, and J. Dean. **The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng) (work in progress)**, October 2014.

[39] Ed. J. Macker. **Simplified Multicast Forwarding (RFC 6621)**, May 2012.

[40] T. Clausen, E. Dearlove, J. Dean, and C. Adjih. **Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format (RFC5444)**, Feb. 2009.

[41] T. Clausen and P. Jacquet. **Optimized Link State Routing Protocol (OLSR)**, 2003.

[42] P. Levis, A. Tavakoli, and S. Dawson-Haggerty. **Overview of Existing Routing Protocols for Low Power and Lossy Networks**, Apr. 2009.

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

[44] JP. Vasseur, M. Kim, K. Pister, N. Dejean, and D. Barthel. **Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks**, March 2012.

[45] P. Thubert. **Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL) (RFC 6552)**, March 2012.

[46] O. Gnawali and P. Levis. **The Minimum Rank with Hysteresis Objective Function (RFC 6719)**, June 2012.

[47] M. Goyal, E. Baccelli, M. Phillip, A. Brandt, and J. Martocci. **RFC 6997: Reactive Discovery of Point-to-Point Routes in Low Power and Lossy Networks**, August 2013.

[48] K. Ramakrishnan, S. Floyd, and C. Black. **The Addition of Explicit Congestion Notification (ECN) to IP (RFC 3168)**, Sept. 2001.

[49] Chonggang Wang, Kazem Sohraby, Bo Li, and Weiwen Tang. **Issues of Transport Control Protocols for Wireless Sensor Networks**. In *Proceedings of International Conference on Communications, Circuits and Systems (ICCCAS*, 2005.

[50] M. Allman, S. Floyd, and C. Partridge. **Increasing TCP's Initial Window (RFC 3390)**, Oct. 2002.

[51] P. Karn and C. Partridge. **Improving round-trip time estimates in reliable transport protocols**. *SIGCOMM Comput. Commun. Rev.*, **17**(5):2–7, August 1987.

[52] C. Paxson, M. Allman, J. Chu, and M. Sargent. **Computing TCP's Retransmission Timer (RFC 6298)**, June 2011.

[53] Hari Balakrishnan, Srinivasan Seshan, Elan Amir, and Randy H. Katz. **Improving TCP/IP Performance over Wireless Networks**. In *Proceedings of the 1st Annual International Conference on Mobile Computing and Networking*, MobiCom '95, pages 2–11, New York, NY, USA, 1995. ACM.

[54] Jeonghoon Mo, Richard J. La, Venkat Anantharam, and Jean Walrand. **Analysis and Comparison of TCP Reno and Vegas**. In *In Proceedings of IEEE Infocom*, pages 1556–1563, 1999.

[55] Saverio Mascolo, Claudio Casetti, Mario Gerla, M. Y. Sanadidi, and Ren Wang. **TCP westwood: Bandwidth estimation for enhanced transport over wireless links**. In *Proc. of Mobicom'01*, pages 287–297, 2001.

[56] Luigi A. Grieco and Saverio Mascolo. **Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control**. *SIGCOMM Comput. Commun. Rev.*, **34**(2):25–38, April 2004.

[57] T. Braun, T. Voigt, and A. Dunkels. **TCP support for sensor networks**. In *Wireless on Demand Network Systems and Services, 2007. WONS '07. Fourth Annual Conference on*, pages 162 –169, jan. 2007.

[58] Adam Dunkels, Juan Alonso, Thiemo Voigt, and Hartmut Ritter. **Distributed TCP Caching for Wireless Sensor Networks**. In *Proceedings of the 3 rd Annual Mediterranean Ad-Hoc Networks Workshop*, 2004.

[59] Chieh-Yih Wan, Shane B. Eisenman, and Andrew T. Campbell. **CODA: congestion detection and avoidance in sensor networks**. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 266–279, New York, NY, USA, 2003. ACM Press.

[60] O.B. Akan and I.F. Akyildiz. **Event-to-sink reliable transport in wireless sensor networks**. *Networking, IEEE/ACM Transactions on*, **13**(5):1003 – 1016, oct. 2005.

[61] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. **Hypertext Transfer Protocol – HTTP/1.1 (RFC 2616)**, 1999.

[62] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, 2000. AAI9980887.

[63] Urs Hunkeler, Hong L. Truong, and Andy Stanford-Clark. **MQTT-S: A Publish/Subscribe Protocol ForWireless Sensor Networks**. In *3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08).*, Bangalore, India, January 2008.

[64] K. Hartke. **Observing Resources in CoAP (work in progress)**. https://datatracker.ietf.org/doc/draft-ietf-core-observe/, Dec. 2014.

[65] A. Rescorla and N. Modadugu. **Datagram Transport Layer Security Version 1.2**. https://tools.ietf.org/html/rfc6347, January 2012.

[66] J. Postel. **Transmission Control Protocol (RFC 793)**, September 1981.

[67] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. **TinyOS: An operating system for sensor networks**. In *in Ambient Intelligence*. Springer Verlag, 2004.

[68] 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.

[69] 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, 2006.

[70] INRIA/ University of Washington. **NS-3 Network Simulator**. `http://www.nsnam.org/`, January 2014. (accessed on 24 March 2015).

[71] András Varga et al. **The OMNeT++ discrete event simulation system**. In *Proceedings of the European simulation multiconference (ESM2001)*, **9**, page 65. sn, 2001.

[72] Texas Instruments. **MSP430F1611**. `http://www.ti.com/product/msp430f1611`, 2011. (accessed on 24 March 2015).

[73] David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, and David Culler. **The nesC Language: A Holistic Approach to Networked Embedded Systems**. In *Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation*, PLDI '03, pages 1–11, New York, NY, USA, 2003. ACM.

[74] Crossbow Technology Inc. **TelosB Mote Platform**, 2009. (accessed on 24 March 2015).

[75] F. Cuomo, S. Della Luna, U. Monaco, and F. Melodia. **Routing in ZigBee: Benefits from Exploiting the IEEE 802.15.4 Association Tree**. In *Proc. of ICC '07*, pages 3271 –3276, june 2007.

[76] Dimitrios Lymberopoulos, Quentin Lindsey, and Andreas Savvides. **An Empirical Characterization of Radio Signal Strength Variability in 3-d IEEE 802.15.4 Networks Using Monopole Antennas**. In *Proceedings of the Third European Conference on Wireless Sensor Networks*, EWSN'06, pages 326–341, Berlin, Heidelberg, 2006. Springer-Verlag.

[77] Kannan Srinivasan, Mayank Jain, Jung Il Choi, Tahir Azim, Edward S. Kim, Philip Levis, and Bhaskar Krishnamachari. **The κ factor: inferring protocol performance using inter-link reception correlation**. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, MobiCom '10, pages 317–328, New York, NY, USA, 2010. ACM.

[78] Kannan Srinivasan, Maria A. Kazandjieva, Saatvik Agarwal, and Philip Levis. **The β-factor: measuring wireless link burstiness**. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, pages 29–42, New York, NY, USA, 2008. ACM.

[79] David Moss, Jonathan Hui, Philipand Levis, and Jung Il Choi. **CC2420 Radio Stack**, Jun 2007. (accessed on 24 March 2015).

[80] Kai Chen, Yuan Xue, and Klara Nahrstedt. **On setting TCP's congestion window limit in mobile ad hoc networks**. In *Communications, 2003. ICC'03. IEEE International Conference on*, **2**, pages 1080–1084. IEEE, 2003.

[81] C. Gomez and J. Paradells. **Wireless home automation networks: a survey of architectures and technologies**. *Comm. Mag.*, **48**(6):92–101, June 2010.

[82] A. Brandt, J. Buron, and G. Porcu. **Home Automation Routing Requirements in Low-Power and Lossy Networks (RFC 5826)**, April 2010.

[83] C. Gomez and J. Paradells. **Wireless home automation networks: A survey of architectures and technologies**. *Communications Magazine, IEEE*, **48**(6):92 – 101, june 2010.

REFERENCES

[84] Jin-Shyan Lee, Yuan-Ming Wang, and Chung-Chou Shen. **Performance evaluation of ZigBee-based sensor networks using empirical measurements**. In *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2012 IEEE International Conference on*, pages 58 –63, Bangkok, Thailand, may 2012.

[85] Bilel Nefzi and Ye-Qiong Song. **Performance Analysis and improvement of ZigBee routing protocol**. In *7th IFAC International Conference on Fieldbuses & Networks in Industrial & Embedded Systems - FeT'2007*, Toulouse, France, 2007. IFAC.

[86] Doo Seop Yun, Sung Ho Cho, Dong Wook Seo, and Min Soo Kang. **An efficient and reliable data transmission control method for relaxing congestion problem in ZigBee network**. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, ICUIMC '08, pages 533–537, New York, NY, USA, 2008. ACM.

[87] Mikko Kohvakka, Mauri Kuorilehto, Marko Hännikäinen, and Timo D. Hämäläinen. **Performance analysis of IEEE 802.15.4 and ZigBee for large-scale wireless sensor network applications**. In *Proceedings of the 3rd ACM international workshop on Performance evaluation of wireless ad hoc, sensor and ubiquitous networks*, PE-WASUN '06, pages 48–57, New York, NY, USA, 2006. ACM.

[88] K. Gill, Shuang-Hua Yang, Fang Yao, and Xin Lu. **A ZigBee-Based Home Automation System**. *Consumer Electronics, IEEE Transactions on*, **55**(2):422 –430, may 2009.

[89] Patrick R. Casey, Kemal E. Tepe, and Narayan Kar. **Design and implementation of a testbed for IEEE 802.15.4 (ZigBee) performance measurements**. *EURASIP J. Wirel. Commun. Netw.*, **2010**:23:1–23:2, April 2010.

[90] E.S. Nadimi, H.T. Sgaard, T. Bak, and F.W. Oudshoorn. **ZigBee-based wireless sensor networks for monitoring animal presence and pasture time in a strip of new grass**. *Computers and Electronics in Agriculture*, **61**(2):79 – 87, 2008.

[91] Pablo Suarez, Carl-Gustav Renmarker, Adam Dunkels, and Thiemo Voigt. **Increasing ZigBee network lifetime with X-MAC**. In *Proceedings of the workshop on Real-world wireless sensor networks*, REALWSN '08, pages 26–30, New York, NY, USA, 2008. ACM.

[92] Kwang Koog Lee, Seong Hoon Kim, Yong Soon Choi, and Hong Seong Park. **A Mesh Routing Protocol using Cluster Label in the ZigBee Network**. In *Mobile Ad-hoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 801 –806, oct. 2006.

[93] ZigBee Alliance. **ZigBee Home Automation Public Application Profile**. `http://old.zigbee.org/Standards/ZigBeeHomeAutomation/Overview.aspx`, Feb. 2010. (accessed on 24 March 2015).

[94] A. Brandt and R. Baccelli, E.and Cragie. **Applicability Statement: The use of RPL-P2P in Home and Building Control (work in progress)**, March 2015.

[95] Benedikt Ostermaier, Matthias Kovatsch, and Silvia Santini. **Connecting things to the web using programmable low-power WiFi modules**. In *Proceedings of the Second International Workshop on Web of Things*, WoT '11, pages 2:1–2:6, New York, NY, USA, 2011. ACM.

[96] Geng Wu, S. Talwar, K. Johnsson, N. Himayat, and K.D. Johnson. **M2M: From mobile to embedded internet**. *Communications Magazine, IEEE*, **49**(4):36–43, 2011.

[97] ABI Research. **ZigBee's 2012 Market Share Lead in Home Automation to be Surpassed by Bluetooth in 2015**. `https://www.abiresearch.com/press/zigbees-2012-market-share-lead-in-home-automation-`, August 2013. (accessed on 24 March 2015).

[98] Carles Gomez, Joaquim Oller, and Josep Paradells. **Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology**. *Sensors*, **12**(9):11734–11753, 2012.

[99] ZigBee Alliance. **ZigBee Home Automation FAQ**. `http://old.zigbee.org/Standards/ZigBeeHomeAutomation/FAQ.aspx`, March 2015. (accessed on 24 March 2015).

[100] Freescale Semiconductor. **BeeKit Wireless Connectivity Toolkit**. `http://www.freescale.com/`, Oct. 2009.

[101] Nouha Baccour, Anis Koubâa, Luca Mottola, Marco Antonio Zúñiga, Habib Youssef, Carlo Alberto Boano, and Mário Alves. **Radio link quality estimation in wireless sensor networks: A survey**. *ACM Trans. Sen. Netw.*, **8**(4):34:1–34:33, September 2012.

[102] Bor-Rong Chen, Kiran-Kumar Muniswamy-Reddy, and Matt Welsh. **Ad-hoc multicast routing on resource-limited sensor nodes**. In *REALMAN '06: Proceedings of the second international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 87–94, New York, NY, USA, 2006. ACM Press.

[103] O. Bergmann, K.T. Hillmann, and S. Gerdes. **A CoAP-gateway for smart homes**. In *Computing, Networking and Communications (ICNC), 2012 International Conference on*, pages 446 –450, 30 2012-feb. 2 2012.

[104] A. Betzler. **Experimental Code and Complete Measurement Results**, October 2013.

[105] Qin Wang and Woodward Yang. **Energy Consumption Model for Power Management in Wireless Sensor Networks**. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2007. SECON '07. 4th Annual IEEE Communications Society Conference on*, pages 142–151, 2007.

[106] Mahesh K. Marina and Samir R. Das. **Routing performance in the presence of unidirectional links in multihop wireless networks**. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '02, pages 12–23, New York, NY, USA, 2002. ACM.

[107] Chonggun Kim, Elmurod Talipov, and Byoungchul Ahn. **A reverse AODV routing protocol in ad hoc mobile networks**. In *Proceedings of the 2006 international conference on Emerging Directions in Embedded and Ubiquitous Computing*, EUC'06, pages 522–531, Berlin, Heidelberg, 2006. Springer-Verlag.

[108] Jung Il Choi, Maria A. Kazandjieva, Mayank Jain, and Philip Levis. **The case for a network protocol isolation layer**. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, pages 267–280, New York, NY, USA, 2009. ACM.

[109] Marco Zimmerling, Federico Ferrari, Luca Mottola, Thiemo Voigt, and Lothar Thiele. **pTunes: runtime parameter adaptation for low-power MAC protocols**. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, IPSN '12, pages 173–184, New York, NY, USA, 2012. ACM.

[110] Carsten Bormann, Angelo P. Castellani, and Zach Shelby. **CoAP: An Application Protocol for Billions of Tiny Internet Nodes**. *IEEE Internet Computing*, **16**(2):62–67, March 2012.

[111] L. Eggert. **Congestion Control for the Constrained Application Protocol**, January 2011.

[112] L. Eggert and G. Fairhurst. **Unicast UDP Usage Guidelines for Application Designers**. `https://tools.ietf.org/html/rfc5405`, November 2008.

[113] Zolertia. **Z1 low-power wireless sensor network module**. `http://www.zolertia.com/products/z1`, March 2010. (accessed on 24 March 2015).

[114] Moteiv Corporation. **TMote Sky: Ultra low power IEEE 802.15.4 compliant wireless sensor module**. `http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf`, June 2006. (accessed on 24 March 2015).

[115] A. Betzler. **Extended Simulation Results for CoCoA+ (All Topologies and Traffic Scenarios)**, March 2014.

[116] Matthias Kovatsch, Simon Duquennoy, and Adam Dunkels. **A Low-Power CoAP for Contiki**. In *Proceedings of the 8th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2011)*, Valencia, Spain, October 2011.

[117] Matthias Kovatsch, Martin Lanter, and Zach Shelby. **Californium: Scalable Cloud Services for the Internet of Things**. In *Proc. IoT*, Cambridge, MA, USA, 2014.

[118] Moshe Sidi, Wen-Zu Liu, Israel Cidon, and Inder Gopal. **Congestion Control Through Input Rate Regulation**. In *Proc. GLOBECOM*, pages 1764–1768, Dallas, TX, USA, 1989.

[119] 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 *Proc. IPSN*, Philadelphia, PA, USA, 2013.

[120] Matthias Kovatsch, Simon Duquennoy, and Adam Dunkels. **A Low-Power CoAP for Contiki**. In *Proceedings of the 8th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2011)*, Valencia, Spain, October 2011.

[121] Pasi Sarolahti and Alexey Kuznetsov. **Congestion Control in Linux TCP**. In *Proceedings of the FREENIX Track: 2002 USENIX Annual Technical Conference*, pages 49–62, Berkeley, CA, USA, 2002. USENIX Association.

[122] H. Ekstrom and R. Ludwig. **The peak-hopper: a new end-to-end retransmission timer for reliable unicast transport**. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, **4**, pages 2502–2513 vol.4, March 2004.

[123] Ed. S. Floyd. **Metrics for the Evaluation of Congestion Control Mechanisms (RFC 5166)**, March 2008.

[124] A. Sehgal, V. Perelman, S. Kuryla, and J. Schonwalder. **Management of Resource Constrained Devices in the Internet of Things**. *IEEE Communications Magazine*, **50**(12):144–149, December 2012.

Declaration

I herewith declare that I have produced this paper without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This paper has not previously been presented in identical or similar form to any other Spanish or foreign examination board.

The thesis work was conducted from 2010 to 2015 under the supervision of Carles Gomez Montenegro and Ilker Demirkol at the Universitat Politècnica de Catalunya.

Barcelona,