University of Antwerp
Faculty of Science
Department of Computer Science
Research group IDLab

Adaptive management of Wi-Fi networks in dynamic and heterogeneous environments

Adaptive management of Wi-Fi networks in dynamic and heterogeneous environments

Patrick Bosch

Submitted in fulfillment of the
requirements for the degree of
Doctor in Science: Computer Science
Academical year 2020-2021

University of Antwerp
Faculty of Science
Department of Computer Science
Research group IDLab

Members of the jury:

prof. dr. Maarten Weyn (chair)
University of Antwerp, Belgium

prof. dr. Steven Latré (promotor)
University of Antwerp, Belgium

prof. dr. Pieter Simoens
Ghent University, Belgium
prof. dr. Jeroen Famaey
University of Antwerp, Belgium
dr. Carlos Donato
Zhilabs, Spain
dr. Tim Verdonck
University of Antwerp, Belgium

University of Antwerp
Faculty of Science

Department of Computer Science
Sint-Pietersvliet 7, B-2000 Antwerp, Belgium

Submitted in fulfillment of the
requirements for the degree of
Doctor in Science: Computer Science
Academical year 2020-2021

Acknowledgments

From the moment I worked on my thesis during my studies, I knew I wanted to pursue a Ph.D. What I didn't know was what journey that endeavor would be! Like with many journeys, what makes it memorable is less the destination but the people you meet. Not to say that the research in itself wasn't enjoyable!

The road to my Ph.D. defense started in 2014 with an email, and soon after that, my first day in Antwerp. I had the opportunity to start at a small research group headed by my promoter, prof. dr. Steven Latre, that grew into something extraordinary. I am genuinely thankful that I could be part of it and that I had the privilege to be mentored by such a great researcher and teacher. The open discussions and the great feedback made me grow as a person and improve more than my academic skillset.

I also want to thank the jury for their valuable time and effort. Their feedback was invaluable and ensured that the quality of the thesis is as expected.

I want to thank several people in person for their excellent collaboration. Starting, I want to thank prof. dr. Chris Blondia for his close collaboration for modeling wireless systems. It was a pleasure to work with him over the years, and I hope you enjoy the results of our work together as much as I did! I want to thank Tom as well, with whom I authored multiple papers and had collaborations nearly over the entirety of my journey. It was a pleasure to work with him over the years, and I appreciate his honesty and brilliant ideas. I also want to thank prof. dr. Johann Marquez-Barja for his confidence in me managing an important B-budget project. Within this project, I also enjoyed working intensely with Pedro and Jetmir and creating something great. While not collaborating with him, I also want to thank Bart for his great stories and his strange obsession with windows. And, of course, I can not forget the help that Johan provided over the years, for example, when I broke the network... again. And last but not least, I want to thank Daniel. Not only for his help but also for his patience.

While I could go on for a long time thanking the many people along my journey, I could fill another thesis with it, I would like to thank all of you for the great discussions, stories, and jokes. In no particular order, I would like to thank Glenn, Esteban, Miguel, Ensar, Carlos, Serena, Paola, Anne, Hanne, Lynn, Jakob, Jeroen, Le, Yorick, Le, the many Barts, and everyone else that I might have forgotten.

Not all support came from colleagues, but family and friends helped me during these years immensely. I want to thank my mother for her tireless effort to provide the best possible environment for my brother and I to grow and become the persons we are. I also want to thank my brother for his sometimes help, but mostly the annoyance, as little brothers are. I am grateful for my girlfriend Debora’s love and care during the years and her reminders to eat when I forgot it again. Last but not least, I want to thank all my friends for their encouragement and support.

Antwerp, April 2021
Patrick Bosch

Samenvatting
– Summary in Dutch –

De laatste twee decennia brachten een fenomenale toename van communicatieapparatuur met zich mee. In het begin domineerden pc's en laptops de markt. Dat veranderde in 2007 met de introductie van de eerste smartphone. Sindsdien is de markt voor mobiele en op batterijen werkende apparaten aanzienlijk gegroeid, en tegen het jaar 2017 bedroeg het totale aantal verbonden apparaten 18 miljard. Hiermee groeide ook het aantal beschikbare communicatietetechnologieën en evolueerden bestaande technologieën verder. Twee grote families van draadloze technologie domineren het segment van de draadloze breedbandklanten. Enerzijds is er mobiele connectiviteit, te beginnen met GSM (2G) over UMTS (3G) en LTE (4G) en binnenkort ook 5G. Daarnaast, begon IEEE 802.11 in het 2.4 GHz-spectrum met overdrachtssnelheden van 11 Mbps en bezet nu het 2.4 GHz-, 5 GHz- en 60 GHz-spectrum met overdrachtssnelheden tot enkele Gbps. Er kwamen echter ook nieuwe technologieën buiten de gebruikerstechnologieën, bijvoorbeeld technologieën met een laag energieverbruik zoals LoRa, SigFox en IEEE 802.11ah, die een groter bereik bieden, maar een lagere doorvoersnelheid. Deze nieuwe apparaattypen en technologieën maken nieuwe toepassingen mogelijk, zoals slimme en verbonden auto's, machines en gebouwen of de trend naar gamestreaming, die een hoge bandbreedte en een lage latentie vereist.

Hoewel veel beschikbare technologieën een breed scala aan mogelijkheden bieden, brengt het ook veel uitdagingen met zich mee. Een apparaat heeft vaak meer dan één technologie, maar kan er maar één tegelijk gebruiken, wat leidt tot verspilling van middelen. Veel van deze technologieën delen gebruiksscenario's bijvoorbeeld mobiel internet en IEEE 802.11, die beide dienen voor toepassingen met hoge doorvoer en lage latentie. De momenteel gebruikte technologie kan echter een slechtere gebruikerservaring hebben dan andere beschikbare opties op die moment. Een andere technologie zou een betere gebruikerservaring kunnen opleveren, maar de gebruiker zou dit niet weten zonder deze eerst te testen. Als dit geautomatiseerd zou zijn, zou de ervaring echter aanzienlijk verbeteren. Een andere uitdaging is het delen en samenwerken in het bestaande spectrum. Veel van deze technologieën gebruiken een soortgelijk spectrum en kunnen elkaar storen, wat tot verminderde prestaties leidt. Als we interferentie kunnen vermijden en technologieën kunnen laten samenwerken, kunnen we de prestaties en het comfort voor alle gebruikers en toestellen verbeteren. Desalniettemin kan de samenwerking tussen technologieën niet alle obstakels overwinnen, en is een schatting en modellering van de prestaties onder bepaalde omgevingsomstandigheden noodzakelijk.

Geïntegreerd technologiebeheer bestaat tot op zekere hoogte met oplossingen zoals de IEEE 1905.1-standaard, LTE-LWA en Software-Defined Networking (SDN). Hoewel het geen geïntegreerde oplossing is, biedt MPTCP ook vergelijkbare functionaliteit op basis van koppelingen die beperkt zijn tot TCP. Ze hebben echter beperkingen; ofwel ondersteunen ze alleen specifieke technologieën of transportprotocollen ofwel richten ze zich op beperkte netwerkdomeinen of use cases. Sommige bieden alleen lokale intelligentie, en wanneer ze netwerkbrede intelligentie ondersteunen, is deze alleen op stroom in plaats van pakketniveau. Vaak maken ze alleen specifieke functionaliteit mogelijk voor hun gebruikssituatie, meestal overdrachten. Aan de andere kant komt prestatiemodellering vaker voor. Het bestaat voornamelijk in IEEE 802.11-technologieën, omdat ze gevoelig kunnen zijn vanwege hun manier om toegang te krijgen tot het medium. Hoewel er modellen voor specifieke scenario's bestaan, zoals alleen IEEE 802.11-stations of apparaten met een andere technologie, bestaan er geen generieke modellen die andere technologieën of zelfs elektromagnetische signalen van niet-communicerende apparaten kunnen bevatten. Naarmate het aantal elektrische apparaten toeneemt, wordt dit een groter probleem en kunnen omgevingen die al last hebben van veel apparaten of technologieën, diepgaand worden beïnvloed. Dit proefschrift gaat in op deze uitdagingen en biedt een transparant platform dat connectiviteit tot stand brengt als een service zonder tussenkomst van de gebruiker. Het platform helpt bij de toewijzing van middelen en een optimaal gebruik van middelen door informatie beschikbaar te hebben, monitoring en modellen en mechanismen.

De eerste bijdrage bestaat uit het onderzoeken en modelleren van de prestaties van IEEE 802.11-systemen wanneer er een storende bron aanwezig is die geen IEEE 802.11-apparaat is en vaak zelfs geen communicatietechnologie. In veel gevallen bepalen andere deelnemers en invloeden van buitenaf de prestatie van een draadloos systeem. Voorbeelden zijn magnetrons en babyfoons, die hetzelfde spectrum gebruiken als IEEE 802.11 op 2.4 GHz en alle communicatie volledig kunnen blokkeren. Andere elektrische apparaten kunnen echter invloed hebben op IEEE 802.11 en moeten niet verzenden als het medium bezet is. Voor dit doel onderzoeken we eerst de prestaties van IEEE 802.11 in een uitdagende omgeving via een draadloos mesh-netwerk, waar we een ongelooflijk hoge latentie zien van maximaal 10 s. Verdere metingen met een gecontroleerde opstelling, en een storende bron onder onze controle, bevestigen die resultaten. Daarnaast maakten we ook een simulatieopstelling. We presenteren eerst een rekenkundig snel model voor latentie in het geval van een externe storingsbron die basiswaarden vereist als er geen storing aanwezig is. We breiden dit verder uit door een volledig analytisch model voor te stellen met hogere rekenkosten, maar dat wel het volledige gedrag van een IEEE 802.11-systeem modelleert. Met dit model onderzoeken we vervolgens de prestaties van IEEE 802.11-systemen onder verschillende scenario's en hoe deze de prestaties van het systeem beïnvloeden. Het model, ondersteund door de metingen, vertoont een sterk verschil in prestatie afhankelijk van de storende bron. De prestatievermindering kan zo groot zijn dat een overstap naar een andere technologie de best mogelijke oplossing is.

De tweede bijdrage bestaat uit het ORCHESTRA-framework, dat intertechnologiebeheer mogelijk maakt dat naadloos is voor de gebruiker en operator. Dit raamwerk stelt ons in staat om externe interferentie te verminderen door meerdere technologieën tegelijkertijd te gebruiken. Het bestaat uit de Virtual MAC (VMAC) -laag en de controller, die nauw samenwerken. De VMAC abstraheert onderliggende technologieën op apparaten waarop het is geïnstalleerd en biedt een enkele verbinding met hogere lagen. Het maakt verschillende geavanceerde functionaliteit op pakketniveau mogelijk, zoals handovers, taakverdeling en duplicatie, beheerd door pakketvergelijkingsregels. De centrale controller verzamelt statistieken van elk VMAC-apparaat en biedt een globaal overzicht van het netwerk. Het ondersteunt ook oudere netwerkprotocollen, apparaten en oplossingen zoals SDN-controllers om een geleidelijke uitrol mogelijk te maken. We laten zien dat een prototype, dat IEEE 802.11 en LTE ondersteunt, in alle functionaliteiten beter presteert dan de huidige industriestandaard MPTCP, terwijl het tegelijkertijd willekeurige transportprotocollen ondersteunt.

Als derde bijdrage presenteren we een load balancing-oplossing voor connectecties met verschillende latentie-eigenschappen. Verschillende technologieën, vooral draadloze, vertonen verschillende latentie-eigenschappen vanwege ontwerpkeuzes in die technologie en hardwarebeperkingen. Een IEEE 802.11 netwerk bereikt bijvoorbeeld ongeveer 14 ms onder goede omstandigheden, terwijl een LTE-netwerk ongeveer 60 ms kan bereiken. Bij het gebruik van een enkele technologie zijn deze waarden meestal geen probleem. Het gebruik van load balancing op pakketniveau tussen die verbindingen met TCP kan een probleem worden en de prestaties beïnvloeden. Voor dit doel bieden we een normalisatiemethode die de latentie van een stroom verzacht en vermindert door een korte kunstmatige vertraging tussen pakketten in te voeren. In plaats van bursts van pakketten te verzenden na het opnieuw ordenen, worden pakketten met een korte tijd ertussen verzonden om burst-gedrag te voorkomen. We gebruiken machine learning om de toekomstige aankomstsnelheid van pakketten te voorspellen en, met aanvullende parameters, de kunstmatige vertraging te berekenen.

Summary

The last two decades brought a phenomenal increase in communication devices. In the beginning, PCs and laptops dominated the market. That changed in 2007 with the introduction of the first smartphone. Since then, the market for mobile and battery-operated devices grew significantly, and by the year 2017, the overall amount of connected devices reached 18 billion. With it, the number of available technologies grew as well, and existing technologies further evolved. Two major wireless technology families dominate the wireless broadband customer segment. On the one hand, there is cellular connectivity, starting with GSM (2G) over UMTS (3G) and LTE (4G) and soon 5G. On the other hand, IEEE 802.11 started in the 2.4 GHz spectrum with transfer rates of 11 Mbps and now occupies the 2.4 GHz, 5 GHz, and 60 GHz spectrum with transfer rates up to several Gbps. However, new technologies outside of user technologies became established, for example, low power technologies such as LoRa, SigFox, and IEEE 802.11ah, which offer more extended range, but lower throughput communication. These new device types and technologies enable new applications, such as connected low power sensors or the trend towards game streaming, which requires high bandwidth and low latency.

While many available technologies offer a wide range of opportunities, it also brings many challenges. A device often has more than one technology, but it can only use one simultaneously, leading to wasted resources. Many of these technologies share usage scenarios, for users, for example, mobile Internet and IEEE 802.11, which both serve for high throughput and low latency applications. However, the currently used technology can have a worse user experience than another available one. Another technology might deliver a better user experience, but the user would not know this without testing it first. Having this automated would significantly improve the experience. Another challenge is sharing and cooperating in the existing spectrum. Many of these technologies use a similar spectrum and might interfere with each other, leading to decreased performance. If we can avoid interference and let technologies cooperate, we could improve the overall performance. Nevertheless, cooperation between technologies can not overcome all obstacles, and estimation and modeling of performance under certain environmental conditions is necessary.

Integrated technology management exists to a certain extent with solutions such as the IEEE 1905.1 standard, LTE-LWA, and Software-Defined Networking (SDN). While not an integrated solution, MPTCP offers similar functionality based on links restricted to TCP. They come with restrictions, though; either they only support specific technologies or transport protocols or target limited network domains or

use cases. Some offer only local intelligence, and when they support network-wide intelligence, it is only on flow instead of packet level. Often, they only enable specific functionality for their use case, which is mostly handovers. On the other hand, performance modeling is more common. It exists primarily in IEEE 802.11 technologies as they can be susceptible due to their way of accessing the medium. While models for specific scenarios, like only IEEE 802.11 stations or devices with another technology, exist, generic models that can include other technologies or even electromagnetic signals from non-communication devices do not. As the amount of electrical devices continuously increases, this becomes more of an issue, and environments that already suffer from many devices or technologies can be profoundly affected. This dissertation addresses these challenges and provides a transparent platform that establishes connectivity as a service without user involvement. The platform helps in resource allocation and optimal resource usage by having information available, monitoring and models, and mechanisms.

The first contribution consists of exploring and modeling IEEE 802.11 systems' performance when an interfering source is present that is not an IEEE 802.11 device and often not even a communication technology. In many cases, other participants and external influences determine the performance of a wireless system. Examples are microwaves and baby phones, which use the same spectrum as IEEE 802.11 in 2.4 GHz and can completely block all communication. However, other electrical devices can impact IEEE 802.11 and need to refrain from sending when it considers the medium busy. For this purpose, we first explore the performance of IEEE 802.11 in a challenging environment via a wireless mesh network, where we see incredibly high latency of up to 10 s. Further measurements with a controlled setup and an interfering source under our control confirm those results, as does a simulation setup. We then first provide a computationally fast model for latency in the case of an external interference source that requires base values when no interference is present. We further extend this by providing a fully analytical model with a higher computational cost, but it models an IEEE 802.11 system's entire behavior With this model, we then explore the performance of IEEE 802.11 systems under different scenarios and how they affect the system's performance. The model, supported by the measurements, shows a steep difference in performance depending on the interfering source. The performance degradation can be so vast that a switch to another technology is the best possible solution.

The second contribution consists of the ORCHESTRA framework, which enables inter-technology management that is seamless to the user and operator. This framework enables us to mitigate external interference by using multiple technologies at the same time. It consists of the Virtual MAC (VMAC) layer and the controller, which work closely together. The VMAC abstracts underlying technologies on devices on which it is installed and offers a single connection to higher layers. It enables several advanced packet-level functionalities, such as handovers, load balancing, and duplication, managed by packet matching rules. The central controller aggregates monitoring information from each VMAC-enabled device and provides a global view of the network. It also supports legacy network protocols, devices, and solutions such as SDN controllers to enable a gradual rollout. We show that a prototype, which supports IEEE 802.11 and LTE, can outperform the current industry standard MPTCP in all functionalities while supporting arbitrary transport protocols.

As a third contribution, we present a load balancing solution for links with different latency properties. Different technologies, especially wireless ones, exhibit different latency properties due to design choices in that technology and hardware constraints. For example, an IEEE 802.11 reaches around 14 ms under suitable conditions, while an LTE network might reach around 60 ms. While using a single technology, these values are mostly not a problem. Using packet-level load balancing between those links with TCP can become a problem and affect performance. For this purpose, we provide a normalizing method that smooths and reduces latency on a flow by introducing a short artificial inter-packet delay. Instead of sending out bursts of packets after reordering, packets are sent with a short time in between to avoid burst behavior. We use machine learning to predict the future packet arrival rate and, with additional parameters, calculate the artificial delay.

Contents

List of Figures

List of Tables

Acronyms

Symbols

3GPP Third Generation Partnership Project

A

ACK acknowledgment

ALME Abstraction Layer Management Entity

AODV Ad hoc On-Demand Distance Vector

AP access point

ARP Address Resolution Protocol

B

BATMAN Better Approach To Mobile Adhoc Networking

BGP Border Gateway Protocol

C

CNN Convolutional Neural Network

CoAP Constrained Application Protocol

CS Carrier Sense

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance

CW Contention Window

D

DCF Distributed Coordination Function

DCTCP Data Centre TCP

DHCP Dynamic Host Configuration Protocol

DIFS DCF Interframe Space

DSL Digital Subscriber Line

DSSS Direct-Sequence Spread Spectrum

E

ED Energy Detection

eNB Evolved Node B

EPC Evolved Packet Core

ETSI European Telecommunications Standards Institute

F

FDD Frequency Division Duplex

FEC Forward Error Correction

FHSS Frequency-Hopping Spread Spectrum

FIFO First-In, First-Out

G

GEO Geosynchronous

GPRS General Packet Radio Service

GTP General Packet Radio Service (GPRS) Tunneling Protocol

I

IoT Internet of Things

IP Internet Protocol

ISM Industrial, Scientific, and Medical

K

KDE Kernel Density Estimation

L

LAN Local Area Network

LBO Local Breakout

LBT Listen-Before-Talk

LEO Low Earth Orbit

LIA Linked-Increases Algorithm

LST Laplace-Stieltjes Transform

LTE Long-Term Evolution

LTE-LAA LTE Licensed Assisted Access

LTE-LWA LTE-Wireless Local Area Network (WLAN) Aggregation

LTE-M LTE-Machine Type Communication

LTE-U LTE-Unlicensed

LTE-V LTE-Vehicular

LVAP Light Virtual AP

M

M2M machine-to-machine

MAC Medium Access Control

MEC Multi-access Edge Computing

MIH Media Independent Handover

MIHF Media Independent Handover (MIH) Function

MIMO Multiple-input and Multiple-output

mmWave millimeter-wave

MoCA Multimedia over Coax

MPTCP Multipath Transmission Control Protocol

N

NAT Network Address Translation

NB-IoT Narrowband IoT

NETCONF Network Configuration Protocol

NFV Network Function Virtualization

NR New Radio

O

OF OpenFlow

OFDM Orthogonal Frequency-Division Multiplexing

OSI Open Systems Interconnection

P

PAN Personal Area Network

PDR Packet Delivery Ratio

PER Packet Error Rate

PHY Physical Layer

Q

QBD Quasi Birth-Death

QoS Quality of Service

R

RAN Radio Access Network

RF Radio Frequency

RNN Recurrent Neural Network

RSSI Received Signal Strength Indicator

RTS/CTS Request-To-Send/Clear-To-Send

S

SDN Software-Defined Networking

SDR Software Defined Radio

SINR Signal-to-Interference-plus-Noise Ratio

SIP Session Initiation Protocol

SNR Signal To Noise Ratio

SVG Support Vector Regression

T

TCN Temporal Convolutional Network

TCP Transmission Control Protocol

TDD Time Division Duplex

TDMA Time-Division Multiple Access

U

UDP User Datagram Protocol

UE User Equipment

V

VANET Vehicular AdHoc Network

VMAC Virtual MAC

VNF Virtualized Network Function

VoIP Voice over IP

VoLTE Voice over LTE

W

Wi-Fi IEEE 802.11

WiMAX Worldwide Interoperability for Microwave Access

WLAN Wireless Local Area Network

1 Introduction

1.1 Context

Wireless devices such as smartphones are ubiquitous in our daily life. They are readily available and offer a wide variety of services to consume. This trend started with the emergence of the first smartphone in 2007. It offered a useful and affordable way to use wireless technologies, such as mobile Internet (3G) and IEEE 802.11, more commonly referred to as Wi-Fi [9]. The relatively cheap and easy access, combined with the device's mobility, helped gain wide popularity. Previously, the devices were more expensive and much less mobile due to their size. While pagers and similar existed, they had less functionality. Notebooks were the most common option and offered similar features to current smartphones, but were much larger. The further we go back, the more expensive devices and Internet access were, and the lower the number of users was. Currently, smartphones are available for less than 50 Euro, while devices with similar functionality cost multiples of that price in the past. Additionally, wireless technologies celebrated their advent in the nineties. Before that, only wired and, therefore, stationary solutions were available. The availability of cheap mobile devices leads to an exponential increase in devices over the years. While in 2003, the number of connected devices numbered 500 million, in 2008, at the introduction of the smartphone, the number of devices caught up with humanity's population at around 6.6 billion [9, 10]. This led to a further increase of 12.5 billion in 2012 and reached 18 billion devices by 2017 with around 6.8 billion mobile users in 2019 [9, 11]. For years, the market for stationary computers stagnates around 1.5 billion devices with a further decrease to 1.2 billion devices by 2022 [12, 13]. Other technologies, such as low power solutions, also already

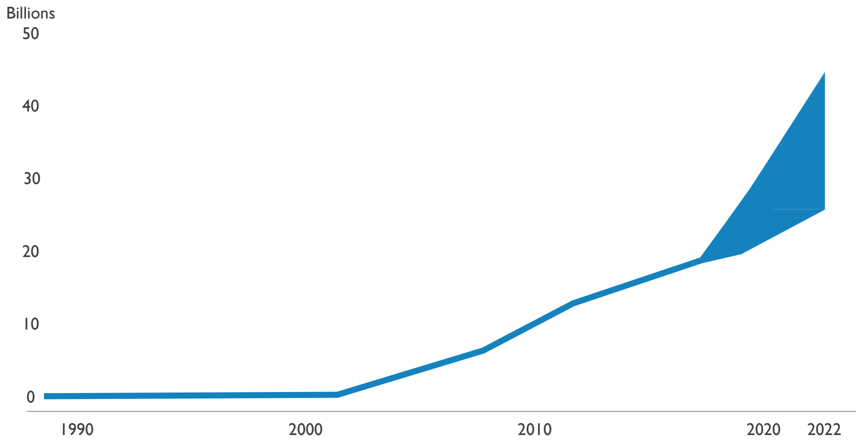


Figure 1.1: Development of connected devices showing an exponential increase over time and significant potential for growth in the future.

compete with existing ones. Internet of Things (IoT) is the primary driver, and with it, the number of connected devices is expected to be in the range from 25.4 billion to 42.6 billion devices by the year 2022 [13, 14, 15].

Besides the number of devices, the number of available technologies also rises steadily, many of them using the same unlicensed spectrum. The main technologies known to users are mobile Internet in the form of Long-Term Evolution (LTE) or 5G and general Internet access through IEEE 802.11 [16, 17, 18, 19, 20]. Both technologies can interfere with each other if LTE is used in the unlicensed spectrum of 5 GHz. There can also be interference from another well-known consumer technology, Bluetooth, in the 2.4 GHz spectrum. However, newer low power solutions, such as IEEE 802.11ah and IEEE 802.15.4, can use the same spectrum and compete for airtime [21, 22, 23]. Other technologies, which are less known to consumers, have similar challenges. Low power solutions, such as SigFox, LoRa, Narrowband IoT (NB-IoT), and LTE-Machine Type Communication (LTE-M), compete in the sub-GHz band while high throughput technologies, such as IEEE 802.11ad/ay and Li-Fi, compete in higher frequencies [24, 25, 26, 27, 28, 29]. Each technology has its properties, advantages, and disadvantages. Nevertheless, not only other communication technologies occupy the limited spectrum. We use myriads of electronic devices that cause interference as well. Among them are interfering sources like microwaves, baby phones, Radio Frequency (RF) sources, or any ill shielded electric device or cable [30, 31, 32, 33, 34]. These sources can impact and further decrease performance, but we currently know little about the quantitative impact. Similarly, we can not estimate or predict a specific source's impact on the performance but require it to use all available resources appropriately. Nearly all communication technologies have the Open Systems Interconnection

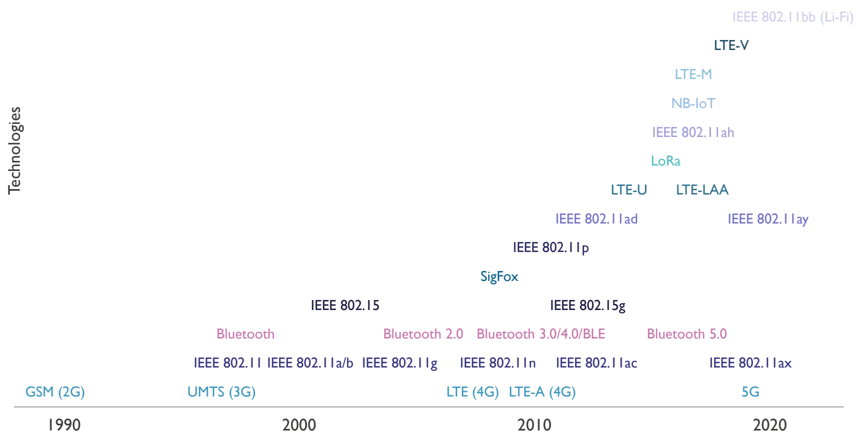


Figure 1.2: Emergence of new and updated technologies over time, increasing the overall number of available technologies.

(OSI) model in common [35]. The OSI model divides communication into different layers with the physical layer as its lowest, followed by the data link layer, usually the Medium Access Control (MAC) layer, and on top of the network and transport layer, and so on. Each technology implements its version of the model, though, and no coordination between technologies takes place. With an increasing number of technologies and an increase in competition for the limited spectrum, this leaves opportunities open to improve resource usage and offer better performance by coordinating the technologies.

The new types of devices and technologies allow a wide range of new applications with different Quality of Service (QoS) requirements. These range from ultra-high bandwidth for 8k video streaming to ultra-low bandwidth for sporadic air quality monitoring in cities. The prediction for video streaming alone is to achieve up to 82 % of the global traffic by 2022 [13]. Additional applications include increased automation in industry 4.0, which requires more focus on latency [36]. Applications can require sub-millisecond latency, while others perform well with several hundred milliseconds of latency. However, other requirements, such as energy consumption for battery-operated devices, jitter for real-time communication, or range, are essential in machine-to-machine (M2M) communication, required by many IoT applications [24]. Guaranteeing these QoS becomes a challenge, but models and new mechanisms combined with more coordination and management between technologies can facilitate a solution.

We can see that the past and current development of devices and technologies provides a challenge to manage them so that they use the available resources efficiently. Additionally, interference between communication technologies or even non-communication interference impacts the performance and can result in a worse user experience by increased latency or decreased throughput. These effects need to be understood and mitigated as well as properly managed.

1.2 Problem Statement

The emergence of new devices and technologies changed the landscape of wireless networks and their management. More and more technologies compete for airtime in the same unlicensed frequency bands and interfere with each other, therefore degrading the performance. Using all kinds of non-communicating electrical devices, many of which are not appropriately shielded, enhances that effect. We currently do not know how much impact such interference can have on the most widely used consumer technology, IEEE 802.11. Until now, it was sufficient that technologies operated without coordination and handled their decisions independently. The increased competitiveness for airtime and therefore increased interference makes it necessary first to understand and describe interfering behavior and then align the technologies' decision-making process. Cooperative management allows for interference mitigation and, therefore, performance improvements. It also achieves higher reliability, mobility, and general performance improvements. These improvements can be realized by cross-technology load balancing or duplication, as well as seamless handovers. Improvements, especially for load balancing, are necessary to adjust for technology differences, such as latency, to better accommodate specific transport protocols. In combination with exact models about factors that are not under the network's influence, we can meet QoS requirements for future applications and services in addition to decreasing costs.

1. Dense IEEE 802.11 deployments in dynamic and heterogeneous environments degrade in performance. Different technologies compete for the same airtime and interfere with each other. Additionally, interference from sources other than communication technologies impact IEEE 802.11 as well and further degrade performance. The degradation manifests in higher latency and lower throughput, but different types of interruptions can have a more severe impact than others. There exists only limited information about these interference types. Similarly, current estimates and models only focus on specific technology interference and do not factor in other possible sources. A fast way of estimating IEEE 802.11 performance in such scenarios is necessary and an accurate model that can be used in network planning and interference mitigation. This model needs to fully describe IEEE 802.11 behavior so that we can derive insights from it.

2. The lack of coordination between different technologies leads to inefficient use of the wireless spectrum and interference between technologies. We need cross-technology management to minimize interfering factors and mitigate performance degradation through cross-technology load balancing and handovers. With this approach, we can maximize airtime and resource usage. This holistic approach needs to work without affecting any technologies,

transport protocols, or applications. Therefore, it has to be transparent to any layer in the OSI stack so that for each layer, it seems that nothing has changed and that it still communicates with the intended layers. Additionally, a management solution is necessary to orchestrate all the devices. We prefer a centralized solution to manage different technologies efficiently with little overhead.

3. While cross-technology load balancing can mitigate and improve performance, it can have its performance penalty if managed poorly. Different latency properties on used links impact the most popular transport protocol, Transmission Control Protocol (TCP). Packets might arrive out of order, or the time between two arriving packets might be too long. TCP considers the packet lost, which in turn leads to a readjustment of the throughput. The mitigation and improvement of this effect are essential to allow using any links together and provide a management solution with much-needed flexibility.

1.3 Hypothesis

From the three problems we identified, we can derive our hypothesis on how to approach future heterogeneous wireless networks:

Quantifying and modeling interference of wireless networks is necessary to mitigate it through cross-technology management with new mechanisms to fully support future networks and the increasing QoS requirements of users.

We live in a world where electrical and wireless devices are standard and use many different technologies. Many technologies use the unlicensed bands as it is open to anyone, and therefore, it is easy for a user to set up a device. This accessibility leads to interference between technologies and regular RF transmissions, microwaves, or ill shielded electrical devices. It is currently unknown how much impact such interference has, and we require both measurements and models to describe it accurately. These models need to be fast in computation and accurate to use them for network planning. While these models can give an overview of the expected performance, they can not mitigate the interference. For this purpose, cross-technology management is necessary that can switch between technologies if the connection is too bad with the current one. We require a framework that can offer this kind of management and offer additional mechanisms to increase reliability and improve throughput. Seamless handovers, load balancing, and duplication are all mechanisms that can improve resource usage in heterogeneous environments but need to be transparent to transport protocols and applications. This way, we can avoid changes in the network infrastructure and facilitate a faster transition. Transport protocols or applications might be affected by the use of different technologies simultaneously due to the technologies' properties. For example, TCP requires packets to arrive in order, and using two technologies with different latency requires reordering and still adds a latency overhead. We need to mitigate this effect by controlling and normalizing the reordered network flow and ensuring that while load balancing, the two technologies appear as one link with only one set of properties.

1.4 Research Questions

From the hypothesis, we can derive four research questions that this thesis will answer:

1. How does IEEE 802.11 behave in heterogeneous environments with a non-IEEE 802.11 interfering source? Moreover, can latency be estimated from the base performance when no such source is present? The behavior of IEEE 802.11 with an interfering source is not known yet, but for guaranteeing latency, it is necessary. The knowledge of the behavior is essential for larger and denser networks. They are already operating under high stress as the number of clients also significantly impacts latency and throughput. Any additional interference can disrupt the network and significantly decrease performance for all other clients in the same network. For active management, a fast and reliable way to estimate latency is necessary as conditions change over time. For example, the number of clients changes through mobility, or an interferer is only sometimes available.

2. Can IEEE 802.11 behavior and latency with a non-IEEE 802.11 interfering source be fully modeled, and does it allow for explanations? Additionally to the previous question and an applicable estimation of performance, a full model is required to research various interfering sources and their effect on IEEE 802.11. Most interferers do not have the same characteristics; some are more frequent but very short, some are more sporadic but longer. On average, they may occupy the medium for the same amount of time, but their impact on the performance of IEEE 802.11 can differ significantly.

3. How can one manage heterogeneous wireless networks and introduce solutions that allow for more functionality while still maintaining legacy compliance? The full availability of devices and technologies allows for a wide variety of opportunities in services they can realize. In the limited spectrum, technologies compete with each other instead of cooperating and using the spectrum as efficiently as possible. A significant requirement for such a solution is full transparency and that it works with existing technologies and networks. Hiding the solution from any participant, may it be a unique technology of a device, other network components that do not support the solution, or end-users and applications is necessary. This approach also helps in keeping the required changes to devices and infrastructure minimal and increasing acceptance.

4. Can a mechanism be defined that allows for load balancing over several links with different latency properties while maintaining or increasing throughput with TCP? One technology alone cannot always solve the throughput requirements. Using several technologies can increase throughput, but it might also decrease TCP throughput if the latency difference of the links is too high as packets arrive out of order or packets are considered lost. A solution needs to restore the order of packets and normalize the time between packets so that it appears as one link with one latency property.

1.5 Research Contributions

Four different research contributions address each research question from the previous section. Each focuses on a specific question. They are closely linked and build on top of each other.

1. Studies and tests of IEEE 802.11 in large-scale, dense, and heterogeneous environments with interfering source and a fast way to estimate the latency in such a scenario (Chapter 3).
    - We present studies from a large scale event that show end to end latency in a multi-hop scenario reaching up to several seconds.
    - Studies in an environment with a non-IEEE 802.11 interfering source are presented that show the impact of such an interferer on IEEE 802.11.
    - We use an estimation algorithm to assess latency with an interferer.
    - We use the base performance when no interferer is present to derive the performance when an interferer is present. This choice allows for fast computation.
2. Full analytical model that describes the behavior of IEEE 802.11 when interfering source present (Chapter 4)
    - Based on previous findings and aims to describe the whole process of IEEE 802.11 if an interfering source is present.
    - Based on a Markov chain to fully describe the back-off mechanism of an IEEE 802.11 station.
    - Interferer is described as a Poisson process to allow for a wide range of interfering sources.
    - We validate the model by comparing it to real measurements as well as simulation.
    - The model is also used to explore the behavior of IEEE 802.11 with different types of interference, may it be frequent and short or sporadic and more prolonged.

3. To address the challenge of heterogeneous wireless networks, we present the ORCHESTRA framework (Chapter 5). The ORCHESTRA framework was developed in collaboration with Tom De Schepper and Ensar Zeljković. My main contribution was the lower-level functionality and management, while Tom and Ensar focused on higher-level management.
    - A two component-design consisting of an on-device Virtual MAC (VMAC) layer and a centralized controller.
    - The virtual layer is fully transparent between the MAC and network layer of a device and helps in abstracting different technologies.
    - The design allows for additional functionality like inter-technology handovers and packet-level load balancing and duplication.
    - The centralized controller collects monitor information from all devices that are VMAC enabled and provides a global view over the network, independent of technology. The placement of the controller can be in the network or the cloud.
    - The controller also manages and reconfigures network flows by using instructions provided by the VMAC.
    - We present and evaluate a prototype implementation with support for IEEE 802.11 and LTE.
4. A machine learning based load balancing mechanism capable of using links with different latency properties while maintaining high throughput for TCP (Chapter 6).
    - Packets on end or intermediate nodes are not directly sent further but are placed in a queue first.
    - To normalize the time between packets when they are forwarded, we use an artificial delay.
    - The delay is computed based on the current arrival of packets on each technology and other information in the system, like the number of packets currently waiting for reordering.
    - This approach requires sub-millisecond prediction times and periodic recomputation intervals in the range of two-digit milliseconds.

1.6 Dissertation outline

Including this introduction, the thesis consists of seven chapters. While Chapter 1 includes the introduction, Chapter 2 follows up with related work and current state of the art solutions and frameworks. After that, we follow the outline presented in Figure 1.3 with a focus on dense and heterogeneous IEEE 802.11 networks that suffer from interference in Chapter 3. This chapter introduces measurements and simulations of such networks and a fast estimation method to derive the latency

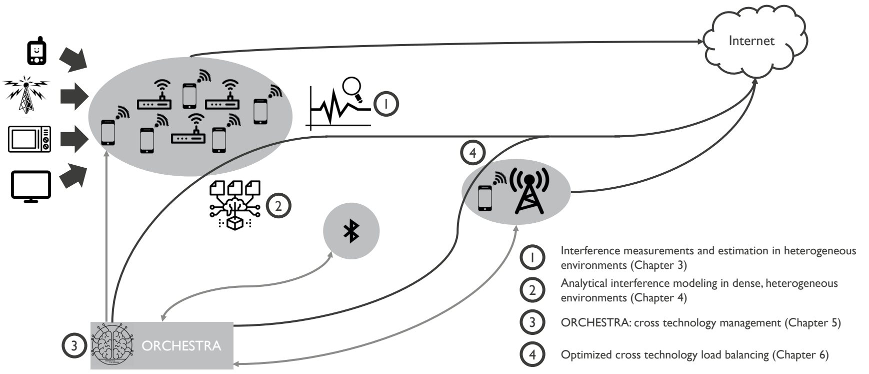


Figure 1.3: Dense and heterogeneous wireless networks need accurate performance models combined with cross-technology management.

in a system with an interferer. In the following chapter, Chapter 4 extends on this and provides a full analytical model for such interference. While computationally more expensive, it has higher accuracy and can fully describe IEEE 802.11 stations' behavior. Chapter 5 introduces the ORCHESTRA framework and describes its components and functionality. ORCHESTRA is designed to elevate the issues stemming from interference and resource scarceness through cross-technology management. Chapter 6 follows up by introducing a load balancing mechanism for links with different latency properties. The ORCHESTRA framework can use this mechanism to improve performance and use more flexible weights when load balancing and reduce the performance impact of cross-technology load balancing. The thesis is concluded in Chapter 7 with a summary of how the research questions and hypothesis were fulfilled.

1.7 Publications

The research results obtained during this Ph.D. research have been published in scientific journals and presented at international conferences. The contributions to the ORCHESTRA framework and the work on channel allocation resulted in two submitted patent applications. The following list provides an overview of the publications and patent applications during Ph.D. research.

1.7.1 A1: Journal publications indexed by the ISI Web of Science "Science Citation Index Expanded"

1. Tom De Schepper, **Patrick Bosch**, Ensar Zeljković, Farouk Mahfoudhi, Jetmir Haxhibeqiri, Jeroen Hoebeke, Jeroen Famaey, and Steven Latré. *ORCHESTRA: Enabling Inter-Technology Network Management in Heterogeneous Wireless Networks.* Published in IEEE Transactions on Network and

Service Management (TNSM), vol. 15, no. 4, pp. 1733-1746, December 2018. doi: 10.1109/TNSM.2018.2866774. [Impact factor: 3.286]

2. **Patrick Bosch**, Tom De Schepper, Ensar Zeljković, Jeroen Famaey, and Steven Latré. *Orchestration of Heterogeneous Wireless Networks: State of the Art and Remaining Challenges.* Published in Computer Communications, vol. 149, pp. 62-77, January 2020. doi: 10.1016/j.comcom.2019.10.008. [Impact factor: 2.766]

3. **Patrick Bosch**, Steven Latré, and Chris Blondia. *An Analytical model for IEEE 802.11 with non-IEEE 802.11 interfering source.* Published in Computer Networks, vol. 172, pp. 107154, February 2020. doi: 10.1016/j.comnet.2020.107154 [Impact factor: 3.030]

4. Tom De Schepper, **Patrick Bosch**, Jakob Struye, Carlos Donato, Jeroen Famaey, and Steven Latré. *ORCHESTRA: Supercharging wireless backhaul networks through multi-technology management.* Accepted by Springer Journal of Network and Systems Management, January 2020. [Impact factor: 1.676]

5. **Patrick Bosch** and Steven Latré. *A machine learning approach for optimizing latency and inter-packet arrival rate in TCP multi-path load balancing.* Submitted to International Journal of Network Management, February 2021. [Impact factor: 1.338]

1.7.2 P1: Proceedings included in the ISI Web of Science "Conference Proceedings Citation Index - Science"

1. Ensar Zeljković, Tom De Schepper, **Patrick Bosch**, Ian Vermeulen, Jetmir Haxhibeqiri, Jeroen Hoebeke, Jeroen Famaey, and Steven Latré. *ORCHESTRA: virtualized and programmable orchestration of heterogeneous WLANs.* In proceedings of the International Conference on Network and Service Management (CNSM), Tokyo, Japan, pp. 1-9, November, 2017. doi: 10.23919/CNSM.2017.8255999.

2. Ian Vermeulen, **Patrick Bosch**, Tom De Schepper, and Steven Latré. *DiMob: Scalable and seamless mobility in SDN managed wireless networks.* In proceedings of the International Conference on Network and Service Management (CNSM), Tokyo, Japan, pp. 1-6, November, 2017. doi: 10.23919/CNSM.2017.8256048.

3. **Patrick Bosch**, Tom De Schepper, Ensar Zeljković, Farouk Mahfoudhi, Yorick De Bock, Jeroen Famaey, and Steven Latré.*A demonstration of seamless inter-technology mobility in heterogeneous networks.* In proceedings of the IEEE International Symposium on a World of Wireless, Mobile, and Multimedia Networks (WoWMoM), Chania, Greece, pp. 1-3, June, 2018. doi: 10.1109/WoWMoM.2018.8449788.

4. **Patrick Bosch**, Steven Latré, and Chris Blondia. *Latency Modelling in IEEE 802.11 Systems with non-IEEE 802.11 Interfering Source.* In proceedings of the International Conference on Network and Service Management (CNSM), Rome, Italy, pp. 275-279, November, 2018.

5. Olivier Jeunen, **Patrick Bosch**, Michiel Van Herwegen, Karel Van Doorselaer, Nick Godman, and Steven Latré. *A Machine Learning Approach for IEEE 802.11 Channel Allocation.* In proceedings of the International Conference on Network and Service Management (CNSM), Rome, Italy, pp. 28-36, November, 2018.

1.7.3 C1: Other publications in international conferences

1. **Patrick Bosch**, Bart Braem, and Steven Latré. *A Network-Driven Multi-Access-Point Load-Balancing Algorithm for Large-Scale Public Hotspots.* In proceedings of the IFIP International Conference on Autonomous Infrastructure, Management and Security (AIMS), Ghent, Belgium, pp. 30-42, June, 2015. doi: 10.1007/978-3-319-20034-7_3.

2. **Patrick Bosch**, Jeroen Wyffels, Bart Braem, and Steven Latré. *How is your event Wi-Fi doing? Performance measurements of large-scale and dense IEEE 802.11n/ac networks.* In proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM), Lisbon, Portugal, pp. 701-707, May, 2017. doi: 10.23919/INM.2017.7987362.

3. Tom De Schepper, **Patrick Bosch**, Ensar Zeljković, Koen De Schepper, Chris Hawinkel, Steven Latré, and Jeroen Famaey. *SDN-based transparent flow scheduling for heterogeneous wireless LANs.* In proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM), Lisbon, Portugal, pp. 901-902, May, 2017. doi: 10.23919/INM.2017.7987404.

4. **Patrick Bosch**, Steven Latré, and Chris Blondia. *IEEE 802.11 Latency Modeling with Non-IEEE 802.11 Interfering Source.* In proceedings of the IFIP International Conference on Wired/Wireless Internet Communications (WWIC), Bologna, Italy, pp. 40-50, June, 2019. doi: 10.1007/978-3-030-30523-9_4.

1.7.4 Patent applications

1. **Patrick Bosch**, Tom De Schepper, Ensar Zeljković, Jeroen Famaey, and Steven Latré. *Network stack for a plurality of physical communication interfaces.* European patent application EP17171131.0, submitted May 2017.

2. Olivier Jeunen, **Patrick Bosch**, Ensar Zeljković, Karel van Doorselaer, and Nick Godman. *A method for allocating frequency channels to a plurality of*

neighboring access points. European patent application 17305724.1-1875, submitted June 2017.

2
Related Work

2.1 Introduction

Heterogeneous wireless networks became more common over the last decade, and many parties proposed solutions to handle them. Throughput and latency were always an essential aspect for wireless networks, and significant research accompanied it. In the following chapter, we explore research concerning IEEE 802.11 performance and behavior modeling to give an overview of existing studies and models. We will also give an overview of different cooperative management solutions, standards, and frameworks and how they tackle heterogeneous wireless networks. Additionally, we give an overview of TCP throughput prediction and load balancing.

2.2 IEEE 802.11 performance and modeling

IEEE 802.11 has been the focus of a significant amount of research, as it is a widely accessible technology that everybody can deploy. Its use of the Industrial, Scientific, and Medical (ISM) band provides easy deployments and requires handling transmission and occupation of airtime from other networks and devices. The Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol handles this part. However, the number of other stations present and non-IEEE 802.11 interference impact its performance heavily. Following, we give an overview the performance of IEEE 802.11 in different scenarios and the state of models to capture the effects of different factors on CSMA/CA.

2.2.1 Performance of large-scale deployments

With IEEE 802.11 being one of the most crucial consumer-oriented wireless technologies, the IEEE 802.11 infrastructure mode is heavily used for large-scale deployments, offering public Internet connectivity to the users. Characterizing the performance of these deployments is critical in understanding their challenges. Again, most studies focus on analytical models to estimate the load and corresponding performance in large-scale hotspots. Ghosh et al. present a model to estimate traffic in large-scale deployments [37]. They use collected data to create a model, which can accurately model the traffic and session distribution. Paul and Ogunfunmi's work is more focused on link performance [38]. Their focus is on a complete analytical model for the IEEE 802.11n standard to model different parameters' behavior. While Zhang et al.'s primary focus is a Signal To Noise Ratio (SNR) based rate adaption, they provide a good overview of the impact of SNR and interference on the Packet Delivery Ratio (PDR) [39]. Focusing on the Packet Error Rate (PER), Ramachandran et al. present measurements for saturated and non-saturated channels [40]. Their setup is relatively dense, and it gives a good overview of the impact of density on wireless networks. Gummadi et al.'s work focuses on interference outside IEEE 802.11, while Rayanchu et al. present a system to detect it with commodity IEEE 802.11 hardware [41] [30]. They show significant vulnerability regarding latency and throughput, with only a low power output on the interfering device.

Experimental performance characterizations of public hotspot deployments are rare. McHenry et al. present spectrum analysis measurements for Chicago [42]. Their IEEE 802.11 measurements show that the band is already well used with Received Signal Strength Indicator (RSSI) up to around -65dB. Their work is, however, limited to a medium-sized hotspot and focuses only on lower layer measurements. Biswas et al. present many network measurements from many deployments, including data about channel utilization, delivery ratio, spectrum analysis, and usage of operating systems and devices [43]. This work is one of the first large-scale experimental performance studies that investigate a real deployment. They focus on the performance of medium-sized hotspots targeting hundreds of potential users.

While the IEEE 802.11 infrastructure mode is still dominant, IEEE 802.11-based mesh setups are also used in production networks with latency-sensitive applications. In this setup, layer two or layer three routing protocols such as Ad hoc On-Demand Distance Vector (AODV), Better Approach To Mobile Adhoc Networking (BATMAN), or BATMAN advanced are being used [44]. As such, ample examples exist of such deployments (e.g., in office environments to handle backhaul traffic [45], as a city-wide community IEEE 802.11 [46]). Vural et al. provide a survey of experimental evaluations of such wireless city-based mesh networks [47]. The work identifies the impact of external interference effects as one of the most critical challenges in setting up a wireless mesh network. Moreover, Vural et al. also present guidelines in terms of node location, directional antennas, and other aspects.

The studies of very high-density deployments have mainly been limited to analytical or simulation models. With this respect, Michaloliakos et al. provide a model for characterizing the performance through simulation of a medium-density deployment such as a conference [48]. Abinader et al. showed decreased throughput in simulations for high-density environments [49]. These simulations show that (i) the density introduces considerable performance costs and (ii) that the performance is reverse proportional to the hop count in the wireless mesh. However, it does not accurately reflect external interference effects, which is crucial in a real deployment.

Interference in wireless systems has been studied for various interfering sources, and it is known that different types of interference can have a detrimental effect. Simple probe requests can already significantly affect the network's performance, although the station does not actively participate in the channel [50]. Ever denser network deployments increase the amount of interference between networks [43]. This effect can also be passively monitored [51]. Additionally, interference can come from adjacent channels, as the frequency bands overlap [52]. Although adjacent channels can improve throughput if used correctly [53]. Because of this, more and more managed tools attempt to detect the presence of non-IEEE 802.11 interference. Airshark proposes a solution to detect non-IEEE 802.11 (Wi-Fi) interference with commodity hardware [30]. The work also shows that the impact on User Datagram Protocol (UDP) throughput is severe, over 90 %, reduced by different interfering sources such as a video camera, an analog phone, or a microwave. Similarly the effect of digital cordless phones, baby monitors, and frequency hopping Bluetooth was explored. While Bluetooth has a minimal effect on throughput, cordless phones and baby monitors can altogether drop the connection [31]. The general occupancy of the 2,4 GHz band can reach up to 34 % during busy hours [54]. IEEE 802.11 networks also interfere with other network technologies like ZigBee and Bluetooth Low Energy, where the latter performs slightly better [32]. Yi et al. provide accurate deployment guidelines for ZigBee to coexist with IEEE 802.11 based on interference avoidance and distance [55]. Introducing LTE into the unlicensed spectrum has a severe impact on the performance of IEEE 802.11 [33, 34]. The throughput of IEEE 802.11 can decrease up to 98 % while LTE is barely affected.

2.2.2 Modeling IEEE 802.11

With the rise of IEEE 802.11 since the late 90s, modeling the network performance became more urgent to predict the network's behavior. One of the most notable contributions for modeling IEEE 802.11 behavior is Bianchi's work [56, 57]. It models the throughput in the saturated case, using a Markov chain to describe the process of the IEEE 802.11 transmission mechanism. This work has some shortcomings, however. It does not consider the retry limit for sending a packet and, therefore, also does not consider dropped packets. An extension was proposed to correct the retry limit and take dropped packets into account, while also proposing an improved access method [58]. This model, especially the Markov chain, serves as a basis for a model describing the average packet latency in an IEEE 802.11 network [59]. Chatzimisios et al. describe how to include the bit error rate simply and straightforwardly [60]. A new Markov chain for the back-off algorithm is proposed and evaluated against an extended version of the existing one [61]. It also corrects an inaccuracy in the work of Chatzimisios et al., where it is assumed that the transmission slot time duration is equal to the average time duration of deferred slots [59]. An extended model is presented with latency and drop probability, average drop time, and the retry limit [62]. A latency distribution model follows, and the Distributed Coordination Function (DCF) is analyzed, concluding that it is prone to long latency [63]. A model for channel latency and jitter is presented for the saturated case by Li et al. [64]. This work also includes a discussion about the initial Contention Window (CW), the maximal back-off stage, and the packet size on the latency. A more advanced latency model for a saturated channel, including jitter, latency distribution, and drop probability, was proposed [65]. A three-dimensional Markov chain, which includes heterogeneous node transmit power levels, is proposed to describe throughput performance under saturation condition [66]. Pham and Tickoo and Sikdar and Challa et al. present extensions of the models of Bianchi and Haitao Wu et al. for the unsaturated case [7, 8, 57, 58, 67]. These include latency and throughput and are based on a queuing model for the packet buffer. Another extension for the unsaturated case is presented by Daneshgaran et al. [68]. This model includes an error-prone channel, while previously, an ideal channel was assumed. The analysis for the channel is limited to the bit error probability. A queue state-driven analysis for ad-hoc networks was proposed, which focuses on end-to-end latency [69]. Saturated and unsaturated cases do not have to be disconnected from each other, as shown by Felemban and Ekici [70]. Here an extension to both cases is presented, focusing on more accuracy in latency and throughput. Similarly, with a focus on both cases, a model for QoS metrics, especially for real-time applications, is proposed [71]. Xu et al. also focus on QoS by maximizing throughput in their model [72]. Kuo et al. show that computation time is also a factor to consider [73]. The authors show a trade-off between accuracy and speed and present one fast but less accurate model and one model that is accurate but slower. While the previously presented articles focus on direct communication, Xie et al. explore the performance for latency, jitter, and packet loss in a multi-hop ad-hoc network [74]. Mehrnoush et al. extend the model of Bianchi for LTE coexistence experiments by including the Energy Detection (ED) threshold so that it can be fine-tuned.

2.2.3 Summary

Many researchers inspected IEEE 802.11 performance in many scenarios that range from larger deployments, over ad-hoc networks, to external interference. The decrease in performance can be as significant as 98 % with LTE in the unlicensed spectrum, but also a high number of devices cause a severe performance decrease. A wide range of analytical models exists that consider many QoS parameters, including throughput, latency, jitter, and others. They focus on pure IEEE 802.11 deployments without considering the external influence and, therefore, can not accurately describe the performance in such a scenario.

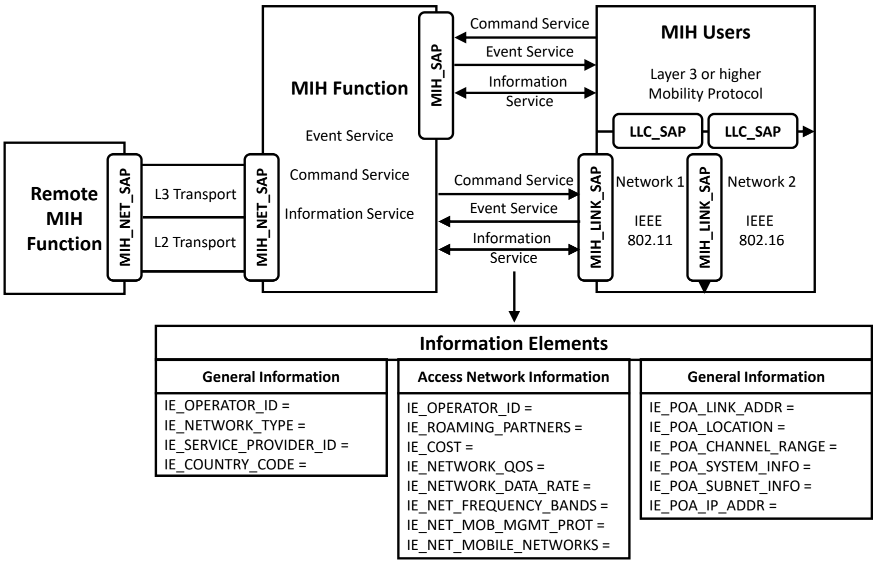


Figure 2.1: The architecture of IEEE 802.21 depicting all included functionality. [1]

2.3 Inter technology management

Managing different technologies in one solution is an essential aspect of research and industry alike. Several standards exist, mostly concerning specific technologies or application scenarios.

2.3.1 Media Independent Handover (IEEE 802.21)

Handover mechanisms have been defined or proposed for roaming across access points (APs) or base stations within single technologies such as IEEE 802.11, IEEE 802.16, or 3G/4G [76, 77, 78, 79, 80]. To offer similar seamless mobility across those different networks (in particular LAN-WAN), and to speed up mobile IP handovers, the Media Independent Handover (MIH) standard was proposed in 2009 [81, 82, 83]. Figure 2.1 shows the general architecture of IEEE 802.21. This standard allows for the continuation of IP sessions across different technologies and networks by introducing the exchange of inter-layer messages through the MIH Function (MIHF). This function is located between layer 2 and layer 3 of the corresponding wireless technology. It can use various Internet Protocol (IP) based protocols, including Session Initiation Protocol (SIP) and Mobile IP, to facilitate handovers. Event notifications, commands, and information services handle the communication between MIHFs of different wireless technologies. An event notification can include a warning about dropping signal quality, while a command issues the initiation of a handover between technologies. Information services are used to exchange information between higher and lower layers as well as the MIHF.

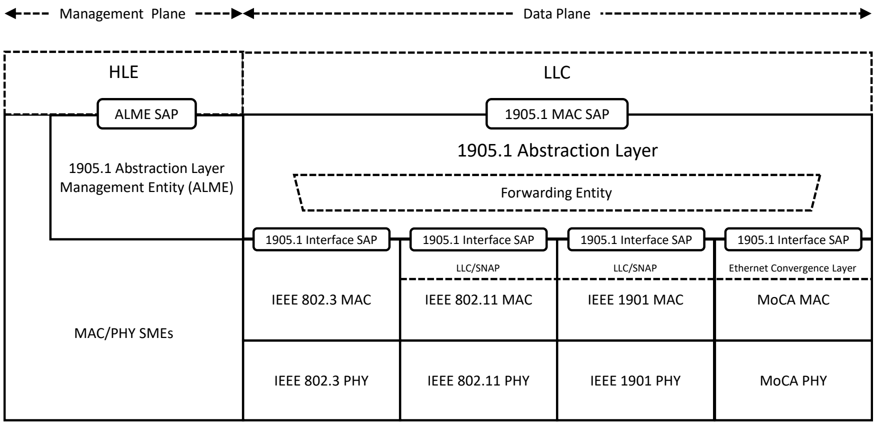


Figure 2.2: The abstraction layer of IEEE 1905.1 [2].

However, this requires adaptations to the underlying technology. Additionally, not only end-devices but edge nodes as well need to support this standard. For centralized management, intermediate network nodes, which do not have a wireless connection, that implement MIHF are necessary. While the focus is on handovers between on the one hand IEEE 802.11 and Worldwide Interoperability for Microwave Access (WiMAX), and, on the other hand, WiMAX and LTE, it is extendable to other technologies [81, 82]. Currently, IEEE 802.21 is being used in Mobile IPv6 to facilitate handovers [84].

The standard was heavily reworked in 2012 and 2017, focusing on security and support for IoT networks and edge and fog computing [85, 86]. It also includes new technologies that only support downlink traffic, like typical broadcasting networks. As the standard does not give any guarantees for handover times, many authors tried to improve handovers times, as summarized by Ghahfarokhi and Movahhedinia [87]. Additional research includes implementation and actual deployment, extending the standard to support a broader range of commands, and handover strategies to improve user experience [87, 88, 89].

2.3.2 IEEE 1905.1

The IEEE 1905.1 standard from 2013 also tries to address the inter-technology handover and management problems, especially in Local Area Networks (LANs) [2]. As shown in Figure 2.2, IEEE 1905.1 compliant devices have an abstract layer hiding the underlying diversity in supported technologies (i.e., Ethernet, IEEE 802.11, Powerline HomePlug, and Multimedia over Coax (MoCA)). This abstract layer is key regarding user-friendliness and QoS, as users do not want to struggle with the low-level specifics of each network technology [90, 91, 92]. It allows for the easy installation of new devices as it is, in essence, plug-and-play. Both users and service providers benefit. It is also compatible with legacy hardware. A unique virtual

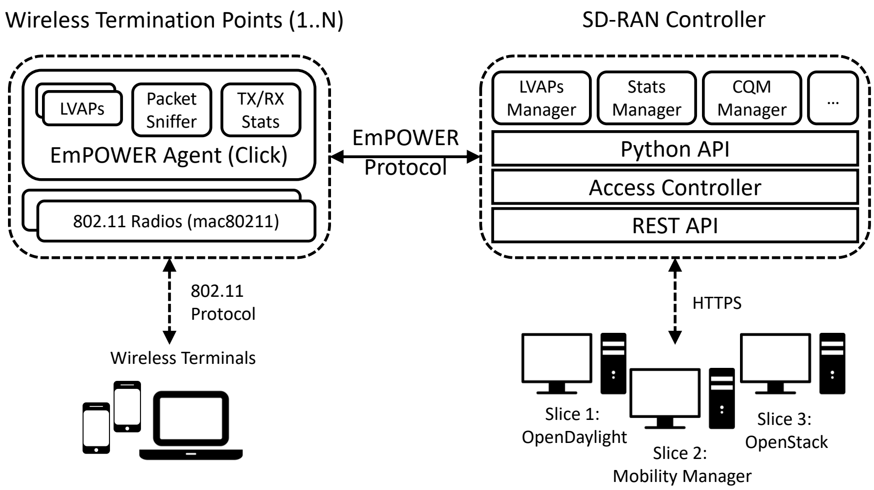


Figure 2.3: The 5G-EmPOWER architecture as an example of SDN in wireless networks [3].

MAC address is required to represent each device on the network. To detect IEEE 1905.1 enabled neighbors, it uses the unique virtual address and communicates with them to create topology information and link metrics.

The Abstraction Layer Management Entity (ALME) service access point enables management of the abstract layer, which serves as a point of contact to higher layers. Besides topology and link metrics, it also offers a way to set flow forwarding rules based on MAC addresses. These packet header matching rules can be used to transparently handover flows and to load balance different flows across the different interfaces. While products exist that support this standard (e.g., Qualcomm Hy-Fi), the standard was never widely adopted by the industry. Research interest is limited to, for example, applying and making use of the standard in a framework for network management [93].

2.3.3 SDN-based solutions

The well-known paradigm of Software-Defined Networking (SDN) can also be transferred from the wired domain to the wireless domain. The splitting of control and data plane allows better management of large deployments by abstracting difficulties, such as handovers, in wireless networks. SDN was mainly deployed in IEEE 802.11 networks, as they had the most need for better management. Much of the decision-making process was either concentrated on the AP or client, which led to wasted resources. Most of the solutions presented in this section follow a similar principle of abstracting functionalities of IEEE 802.11 and centralizing them in a controller (e.g., Figure 2.3). However, each solution has a specific approach and adjustments.

2.3.3.1 SDN@Home

SDN@Home proposes an alternative for an abstract MAC layer to bring SDN into LANs [94, 95]. SDN@Home transforms the gateway into an SDN controller, which is ultimately controlled by a network administrator. In addition to SDN in wired networks, the gateway configures forwarding tables and takes wireless network conditions into account, such as radio configuration, mobility, and interference. There is no need, though, for specialized hardware such as Software Defined Radios (SDRs). A programmable MAC engine allows for wireless devices' configuration without modifying the underlying physical hardware [96]. The channel, transmission power, priority, and other parameters can be modified. While this approach allows using legacy hardware, it still requires modification on a software level to enable modification of parameters.

2.3.3.2 ODIN

To make (dense) wireless networks more manageable and increase IEEE 802.11 experience and QoS, ODIN is proposed as one of the first wireless SDN controllers [97, 97, 98, 99]. Essential in its design is the introduction of the Light Virtual AP (LVAP) abstraction, as an addition to the default virtualization of APs. The association state is virtualized and separate from the physical AP. Each station is assigned to a unique LVAP instead of a physical AP. In the case of a handover between two physical APs, the LVAP is transferred to the new AP, and the station experiences seamless mobility. The ODIN architecture comprises the ODIN master, the controller of the network, and the ODIN agent, which runs on the physical APs. The controller is implemented on top of the Floodlight OpenFlow (OF) controller. This setting enables full OF capabilities and thus allowing for a global network view.

2.3.3.3 5G-EmPOWER

A more recent wireless SDN contribution is the 5G-EmPOWER networking framework [1] [3, 99, 100]. This framework builds on top of the same principles as ODIN, especially the notion of LVAP to handle mobility and handovers in the network. The main extension compared to ODIN focuses on the network's programmability through Virtualized Network Function (VNF) by using either Python interfaces or a REST API [3, 100]. This focus leads to increased control and monitoring information of different aspects in the network, including, but not limited to, available bandwidth or load on a physical AP. The framework's main goals are client state management, resource allocation, and network reconfiguration [3]. The framework focuses not only on IEEE 802.11 networks but also on managing cellular networks and devices through a specialized interface, which is currently being implemented [3, 100].

[1] https://5g-empower.io/

2.3.3.4 Wi-5

Within the context of the European Horizon 2020 program, the Wi-5 project focuses on managing IEEE 802.11 APs more efficiently [101]. Instead of deploying more hardware, it aims to evolve APs into more intelligent network nodes, enabling inter-provider cooperation and seamless user experience. Instead of letting APs decide on their own, they exist in a framework with a centralized controller. The controller then tries to minimize interference and maximize throughput between different AP deployments. Further, it allows seamless handovers between providers and, therefore, a better user experience. Additionally, QoS management and Network Function Virtualization (NFV) enable low latency or high throughput services. It also intends to reduce operational costs by reducing the management costs of each service provider.

Research in the Wi-5 project covers a broad spectrum to achieve the goals of the project. It ranges from flow optimization of small packets [102], over frame aggregation to support either lower latency or higher throughput [103], to being more flexible in moving wireless clients [104].

2.3.4 3GPP based solutions

The ever-growing bandwidth and traffic speed demands have urged the Third Generation Partnership Project (3GPP) community to explore the wireless spectrum outside of the traditional licensed 3G/4G bands. In order to offload traffic, the use of unlicensed spectrum (i.e., LTE Licensed Assisted Access (LTE-LAA)/LTE-Unlicensed (LTE-U)) has been proposed [105, 106, 107, 108, 109, 110, 111]. Both proposals define the use of LTE in unlicensed spectrum, specifically the 5 GHz band. LTE-U was defined outside the 3GPP standardization body first. Afterward, it was standardized in the 3GPP release 12. In this version, downlink traffic could be offloaded to the unlicensed spectrum, while uplink traffic still used the licensed spectrum. To speed up the technology launch, mainly in countries such as the United States and China, no Listen-Before-Talk (LBT) protocol was specified. The lack of such a protocol led to researching the effect of LTE on IEEE 802.11 and vice versa [33, 34, 112]. The common conclusion is that LTE transmissions can heavily affect IEEE 802.11 performance, while this effect is very minimal the other way around. Unlicensed spectrum also allows for other types of services, such as device-to-device communication [113].

The complications led to a more refined version with a mandatory LBT protocol with ED [114]. It also employs a so-called freeze period, where LTE leaves free airtime that IEEE 802.11 can use. While the specification in 3GPP release 13 only allows for downlink traffic in the unlicensed spectrum, besides dynamic channel selection, the extended version of 3GPP release 14 allows for uplink traffic in the unlicensed spectrum. LTE-LAA with LBT leads to better coexistence than LTE-U, and with the mandatory LBT, it can also be used worldwide [111, 115]. The throughput per AP while using LTE-LAA as coexistence can be increased compared to IEEE 802.11 sharing spectrum with other IEEE 802.11 devices.

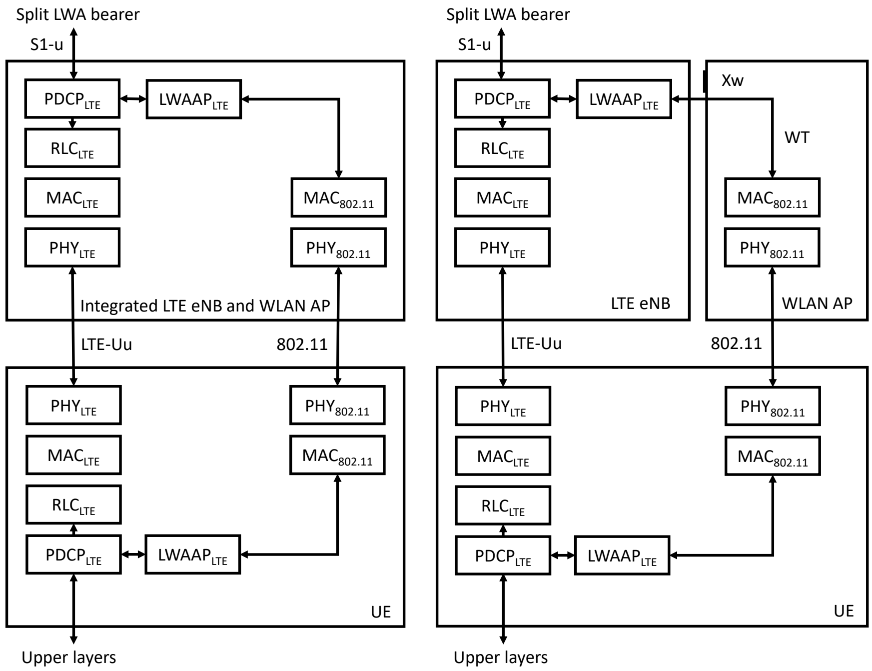


Figure 2.4: The LWA architecture provides an integrated access point or an external one [4].

Based on LTE-LAA, but specified outside of the 3GPP standardization body, MulteFire, specified by the MulteFire Alliance in version 1.0 in 2017, aims to fill the market for small cells and local deployment [116, 117]. It supports an LBT protocol and private deployments and mainly works in the unlicensed and shared spectrum. Contrary to standard LTE deployments, no service provider is necessary, but it allows it to connect to a public network as a neutral host. Deployment and operation work similar to IEEE 802.11, where a company can manage its network. The use of the LTE protocol promises similar advantages of a centralized scheduled network with voice and data services alike.

In addition to specifying LTE in unlicensed spectrum, 3GPP also defines the use of IEEE 802.11 in combination with LTE [4, 118, 119]. LTE-Wireless Local Area Network (WLAN) Aggregation (LTE-LWA), first presented in 3GPP release 13, proposes the use of an IEEE 802.11 AP over which LTE traffic is encapsulated in the standard IEEE 802.11 MAC frame (Figure 2.4). This combination requires either a physical integration of an IEEE 802.11 AP into an Evolved Node B (eNB) or that the AP is externally connected through a network interface. The LTE-LWA approach introduces fewer coexistence issues than LTE-U or LTE-LAA and does not require hardware changes on the infrastructure, except support for the new interface, which can be done in software [120]. From a user perspective, both LTE and IEEE 802.11 appear seamlessly as mobile traffic flows are tunneled over the

IEEE 802.11 connection, and a handover between both technologies is possible. The main focus of research for LTE-LWA lies in achieving high performance and low latency handovers. Therefore, most research focuses on decreasing the overhead of handovers and scheduling them properly, reducing the handover time in both cases [121, 122].

3GPP Release 15 in 2018 specifies a new generation of mobile network technology. It specifies the fifth generation of mobile networks, more generally known as 5G. Officially called 5G phase 1, this release introduces the first standards for 5G technology and specifies, among others, the New Radio (NR) paradigm [19, 20]. This radio interface is meant to unify and replace all existing 3G / 3G technologies, including LTE-U, LTE-LAA, and LTE-LWA, and enable future development and progress.

The NR interface specifies from the start operation frequencies that start from below 1 GHz and reach up to 52.6 GHz [20]. The major change focuses on the spectrum above 6 GHz, which introduces millimeter-wave (mmWave) communications to mobile technologies [19]. This spectrum range is needed to support future high bandwidth applications and enables small cell deployments, exploiting the frequency band's limited range. MmWave technologies pose a challenge, though, as connections can quickly drop, or a small amount of interference can cause performance degradation. Beams between multiple directed antennas and Multiple-input and Multiple-output (MIMO) overcome these challenges. Additionally, beamforming and handovers between higher and lower frequencies are under development [19, 20].

2.3.5 Multipath TCP

In order to maximize resource usage and increase redundancy in multi-technology networks, MPTCP has been proposed. This TCP extension offers multiple regular TCP connections (denoted as sub-flows) as one to an application while allowing each sub-flow to follow different paths through the network (Figure 2.5) [123]. A scheduler can thus divide or duplicate application data across these sub-flows, based on the ever-changing network characteristics (e.g., increased RTT), to attain a higher throughput or increased reliability [5]. Additionally, the scheduler can keep one sub-flow idle and only use it when the main sub-flow breaks. In this case, the controller already established the fallback sub-flow, meaning the handover can occur very quickly and fully transparent to upper layers.

While MPTCP aims at improving QoS and network resource utilization, it focuses only on the alternative paths between two hosts and not network-wide optimization [124]. It can also have degraded performance if the receive buffer is too low or if the network paths are heterogeneous [125, 126, 127, 128]. In both cases, the available throughput drops. Android and iOS devices (e.g., by Siri) use MPTCP actively on a large scale [129, 130]. Furthermore, telecom operators use MPTCP to split traffic across both wired and wireless backbone networks (called hybrid access networks). This type of use is, in particular, the case for Digital Subscriber Line (DSL) and LTE solutions to circumvent the limited capacity of

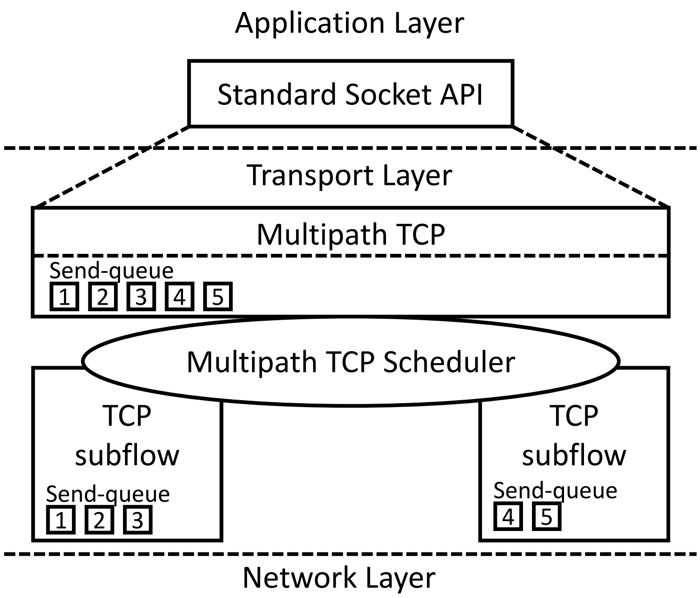


Figure 2.5: The Multipath Transmission Control Protocol (MPTCP) architecture [5].

DSL wires (also known as DSL-LTE bonding). This technology is, for instance, commercially available as Hybrid Access Solution [2].

2.3.6 Application layer and operating system based solutions

While the previous focus lay on lower layer solutions, the application layer also offers inter-technology or intra-technology handover solutions. The Border Gateway Protocol (BGP) offers a decentralized routing protocol based on TCP [131, 132]. Each routing device opens a TCP port and listens and sends keep-alive messages, which show which links are alive and which not. While BGP is most famous for its use in the routing of the Internet, it can be used in smaller independent networks, making it also suitable for wireless networks. It is not directly usable for seamless handovers; however, it can identify multiple routes that traffic can take. SIP, on the other hand, with its extension, focuses on Session Mobility [133, 134, 135]. Each device registers with a registrar that manages the current reachability of the device through its identifier. When the network or technology changes, the devices updates its IP address with its registrar, which in turn forwards it to registrars of currently connected devices. This mechanism allows for fast handovers, but there is a short downtime until the IP address is updated. SIP is currently used by Voice over LTE (VoLTE) to allow voice calls over the mobile data connection.

Operating systems continue technology integration as well, especially in the

[2] https://www.tessares.net/

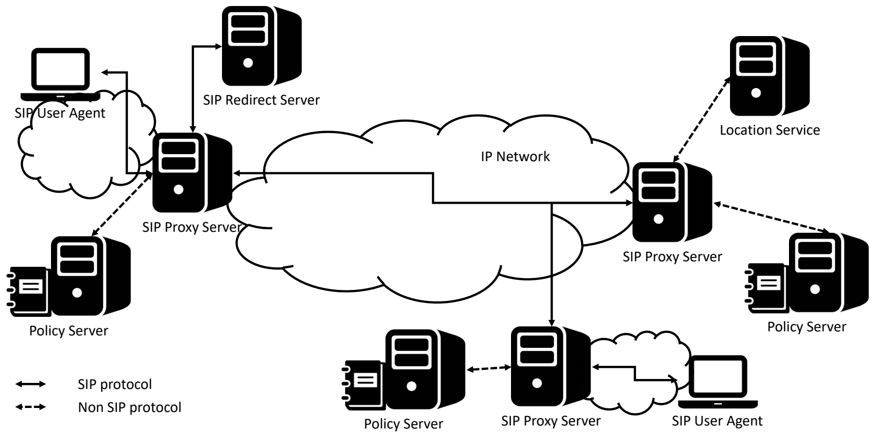


Figure 2.6: The SIP architecture is an example of application-based solutions [6].

mobile market segment. By monitoring IEEE 802.11 and LTE parameters, iOS from version 12 can nearly seamlessly hand over connections between technologies. This behavior is mainly achieved by reacting early on and preferring the more stable technology.

2.3.7 Low power based technologies

The IoT promises billions of wireless devices for monitoring, information gathering, and low power wireless communication. Many technologies offer this functionality. They range from low throughput of hundreds of bytes per second with long-range (e.g., LoRa [21], Sigfox [25], and NB-IoT [26]) to high throughput of hundreds of kilobytes per second but shorter ranges (e.g., IEEE 802.15.4g [22], IEEE 802.11ah [23], and DASH7 [136]). Similar to other technologies, these also operate independently from each other.

Current solutions to provide unified management are mostly limited to research with limited products available for select technologies like Wizzilab's D7A::LoRa::SigFox gateway [3]. The European Telecommunications Standards Institute (ETSI) defined a M2M service layer that abstracts the technology on the service layer and allows interoperability [137]. In research, there are mainly two approaches to manage different technologies. The first is based on SDN, web services utilizing REST, and Constrained Application Protocol (CoAP) [138, 139]. While the second is based on a multimodal approach that integrates multiple technologies into the same hardware [140, 141]. In both cases, energy efficiency is the most pressing concern, which results in lightweight solutions that require little power to operate. The first approach allows for simplified management by using established methods to manage the network. The second approach can reduce deployment costs

[3]http://wizzilab.com/

while also further reducing energy requirements due to singular hardware.

2.3.8 Summary

Table 2.1 provides an overview of the different solutions, standards, and frameworks. The main thing in common between all of them is that they can not provide a complete solution. Most are lacking in technology support, apart from MPTCP and application-based solutions. Those who do not have limited use cases like MPTCP are limited to a specific transport protocol. Coordination efforts are also limited, as is the control-level, which is mostly flow-based. All in all, none of the presented solutions can tackle heterogeneous wireless networks thoroughly, but they always provide a trade-off.

2.4 Multi-path load balancing

Multi-path load balancing is an essential aspect of heterogeneous networks to allow for flexibility in network management. It allows us to use multiple links simultaneously but comes with the caveat that differences in latency across the links can negatively impact performance. The work for multi-path load-balancing is limited and mostly focuses on avoiding high latency differences. This section focuses on two aspects of related work. First, it considers TCP throughput optimization in general and second, time series forecasting, which is vital to arrive at accurate predictions for future performance.

2.4.1 TCP throughput optimization

Optimizing TCP throughput has a long history, mainly because it is a reliable protocol and is widely used, but it reacts considerably to packet loss. Predicting and optimizing the performance was done early and on a wide variety of networks. Dong Lu et al. analyze throughput for large TCP flows and develop a model that can predict throughput in such a case [142]. Mathematical models and history-based approaches also describe the throughput; although history-based approaches are path-dependent, they provide better results [143]. Similar work was done for mobile networks where the focus lay on measuring and modeling throughput and latency and their effect on web services [144]. However, analysis of more specific use cases, such as video calls over wireless networks, was done [145]. This work's main focus was the packet latency and varying available bandwidth, where a historic data prediction was used. Sun et al. did similar work on bit-rate adaption, where data sets were analyzed to adapt bit-rate to improve video quality [146]. Dynamic network throughput prediction for live video streaming was also proposed using simple moving averages to ensure low latency [147]. Identifying repeating patterns for traffic measurements and traffic engineering for machine-to-machine communication in an IoT environment was also conducted to optimize performance [148]. Improving performance can also be done by avoiding IEEE 802.11 outages through congestion control to improve the performance with a central controller [149]. Liu and Lee conducted a broader study on throughput prediction, where seven algorithms were analyzed with data from mobile networks [150]. It turns out that throughput in mobile networks is predictable, and the prediction algorithm does not need to be too complex to achieve high accuracy.

To make use of multiple interfaces and links, MPTCP was proposed [123]. It is based on TCP but allows us to load-balance a TCP flow with sub-flows through multiple interfaces and schedules packets depending on the available link qualities. However, it does not have any central intelligence and therefore has to compete with other active solutions in a network. MPTCP can also impact other TCP flows negatively while not adding any benefit to MPTCP flows [124]. This effect is mainly due to the Linked-Increases Algorithm (LIA) that MPTCP uses by default. Measurements on MPTCP with an IEEE 802.11 and LTE network show a viable solution to keep latency in an acceptable range [151]. Scheduling is essential to achieve high performance with MPTCP. Xue et al. provide one of the most recent scheduling algorithms that take packet loss into account and take care of out-of-order packets that inevitably occur when using multiple paths [152].

However, not only mobile networks use TCP, but data centers can also massively profit from it. An adjusted version of TCP, Data Centre TCP (DCTCP), was developed to cope with the challenges of cloud data centers [153]. The use of MPTCP is not limited to wireless networks either. Data centers benefit from using it if there are multiple paths available, as it load balances flows [154]. However, also, SDN is heavily used to achieve flow-based load balancing with multi-path support [155].

2.4.2 Time series forecasting

Recent trends in machine learning tend to use neural networks for time series forecasting. Especially Recurrent Neural Network (RNN) are popular due to their ability to connect information over a long time [156]. However, RNNs are not the only option; Convolutional Neural Networks (CNNs) show promising results to efficiently deal with time series information [157, 158]. Temporal Convolutional Networks (TCNs), a modification of CNNs, were first used by Google in their Wavenet topology, and Bai et al. demonstrated that they could outperform traditional RNN algorithms [159, 160]. However, more classical approaches can be used, such as a genetic algorithm, to estimate TCP throughput based on historical data points [160]. It shows that high accuracy can be achieved. Alternatively, an approach based on Support Vector Regression (SVG) that allows for accurate prediction of TCP throughput in multi-path environments can be used [161]. Another well-known solution that is widely used in practice is tree boosting [162]. One of the most widely used frameworks for tree boosting is XGBoost [163]. It is highly optimized in terms of resource usage and prediction speed and therefore used in production.

2.4.3 Summary

TCP throughput optimization and packet arrival prediction have been thoroughly researched, but current approaches mainly focus on specific use cases or specific networks. The work on providing a solution for heterogeneous wireless networks focusing on latency differences and their effect on load balancing TCP has not been done yet.

Features		Technologies
Network domain	LAN	IEEE 1905.1, SDN@Home, ODIN, 5G-EmPOWER
	LAN-WAN	IEEE 802.21, Wi-5, MulteFire, LTE-LWA
	Any	LTE-U/LAA, 5G New Radio, BGP, SIP
Coordination	None	IEEE 802.21
	Local	SDN@Home, ODIN, 5G-EmPOWER, MulteFire, LTE-LWA, MPTCP, BGP, SIP
	Global	IEEE 1905.1, Wi-5, LTE-U/LAA, 5G New Radio
Control-level	Flow-based	IEEE 802.21, IEEE 1905.1, SDN@Home, ODIN, 5G-EmPOWER, Wi-5, LTE-U/LAA, MulteFire, LTE-LWA, 5G New Radio, BGP, SIP
	Packet-based	MPTCP
Transport protocols	Any	IEEE 802.21, IEEE 1905.1, SDN@Home, ODIN, 5G-EmPOWER, Wi-5, LTE-U/LAA, MulteFire, LTE-LWA, 5G New Radio, BGP, SIP
	TCP	MPTCP
Vertical handovers	Yes	IEEE 802.21, IEEE 1905.1, Wi-5, LTE-LWA, MPTCP, SIP
	No	SDN@Home, ODIN, 5G-EmPOWER, LTE-U/LAA, MulteFire, 5G New Radio, BGP
Load balance single flow	Yes	MPTCP
	No	IEEE 802.21, IEEE 1905.1, SDN@Home, ODIN, 5G-EmPOWER, Wi-5, LTE-U/LAA, MulteFire, LTE-LWA, 5G New Radio, BGP, SIP
Packet duplication	Yes	MPTCP
	No	IEEE 802.21, IEEE 1905.1, SDN@Home, ODIN, 5G-EmPOWER, Wi-5, LTE-U/LAA, MulteFire, LTE-LWA, 5G New Radio, BGP, SIP
Client changes required	Yes	IEEE 802.21, IEEE 1905.1, SDN@Home, MPTCP
	No	ODIN, 5G-EmPOWER, Wi-5, LTE-U/LAA, MulteFire, LTE-LWA, 5G New Radio, BGP, SIP
Infrastructure changes required	Yes	IEEE 802.21, IEEE 1905.1, SDN@Home, MPTCP, ODIN, 5G-EmPOWER, Wi-5, LTE-U/LAA, MulteFire, LTE-LWA, 5G New Radio
	No	BGP, SIP

Table 2.1: Feature comparison of existing and upcoming solutions.

3

Estimating latency in IEEE 802.11 system with interfering source

The contributions presented in this chapter are based on the publications titled *"How is your event Wi-Fi doing? Performance measurements of large-scale and dense IEEE 802.11n/ac networks"*, *"Latency Modelling in IEEE 802.11 Systems with non-IEEE 802.11 Interfering Source"*, and *"IEEE 802.11 Latency Modeling with Non-IEEE 802.11 Interfering Source"*.

3.1 Introduction

Many of today's and future services, like Voice over IP (VoIP), game streaming, or real-time monitoring, rely on low latency. However, achieving low latency in wireless technologies is not an easy task, especially under environmental conditions that include other devices, other wireless communication technologies, and even influences from non-communication technologies. The wireless spectrum is a shared medium, and while technologies can be placed in separate frequency bands, the space of usable frequency, physical or regulatory, is limited. In this chapter, we will focus on the performance of IEEE 802.11 technology, which is the most widely used consumer technology for wireless Internet access.

The main reason for its popularity is the use of the unlicensed spectrum, which makes it easy for everyone to deploy their wireless network. The ease of deployment comes with a cost, though, as it uses CSMA/CA as its medium access, which is an LBT protocol, to ensure that multiple devices can use the spectrum without blocking each other [164]. Each device listens to the medium for IEEE 802.11

traffic with each Carrier Sense (CS) mechanism as well as to any energy above a certain threshold with its ED mechanism. Contrary, managed networks use a centralized resource allocation mechanism, such as LTE with scheduling. CS and ED both sense the medium. The first detects and decodes IEEE 802.11 traffic and estimates how long the channel will be busy by reading the preamble of the packet. The latter detects energy that is above a threshold, specified by the hardware vendor.

If the device detects any signal, it does not transmit, but backs off with the value of the back-off timer randomly chosen from its current CW. Similar, if it tries to send and a collision occurs because an external interference source becomes active during the transmission or another station tries to send as well, the station backs off. In both cases, the size of the CW doubles each time until it reaches the maximum size. If it reaches the maximum number of retries, the device drops the packet. Collisions with packets of other IEEE 802.11 sources always lead to packet loss. However, collisions with external sources can happen unnoticed, thanks to the Forward Error Correction (FEC) mechanism that can recover from the collision, in particular when the device uses a conservative data rate.

It becomes apparent that the more devices use the wireless spectrum and the more sources transmit any type of energy that is above the threshold, the higher the degradation of performance is. As long as the medium has unused airtime, this is less of a problem, but as soon as devices mostly use up the airtime, the performance drops sharply. The use of a communication protocol gives each node the possibility to successfully transmit as they back off after a collision, which leaves an open window for transmission. However, this behavior is completely different with a source that does not follow a communication protocol as it can become active at any point in time and does not necessarily leave a window for transmission.

For proper network planning, it is essential to know the performance of each device and link, depending on its environment. Performance modeling is not an entirely new topic. We presented different performance measurements, approaches, and models in Section 2.2.1 and Section 2.2.2, which highlight the importance of IEEE 802.11 performance. What is currently missing in research is the consideration of external influences on the IEEE 802.11 system performance. In this chapter, we will explore performance impacts with different experiments and will also present an approach to determine the expected performance quickly. We will see in the initial performance measurements in Section 3.2 that IEEE 802.11 performance suffers significantly from interference and primarily external influences. This measurement is essential and, we can draw conclusions from it and build a fast and accurate model that gives us the performance under different conditions without performing measurements. We do require a baseline performance, though, which we can derive from the measurements.

The rest of this chapter is structured as follows. First, we present performance results from a real deployment on a large-scale event consisting of an ad-hoc network. Second, we show the results of a more controlled environment and the behavior of IEEE 802.11 on different types of external interference. Third, we provide a fast way to compute latency with external interference from the latency when no interference is present.

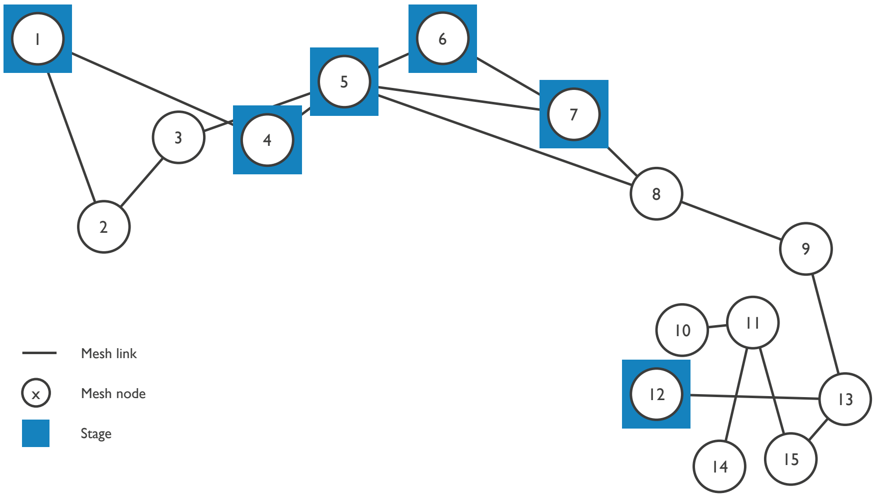


Figure 3.1: The topology of the mesh deployment at the event.

3.2 IEEE 802.11 performance in large-scale event deployment

In this section, we explore the performance impact of external influence on an IEEE 802.11 mesh deployment at a large-scale music festival.

3.2.1 Experimental setup

We deployed an IEEE 802.11 based mesh network of 15 nodes with up to four antennas per node across the festival area. As the network should work as a backhaul network, we placed the nodes close to where other systems need access. Those locations include six stages, and usually, there are hundreds to several thousand people around them. We chose directional antennas to improve performance, as our links were point-to-point, and the maximum distance was close to one kilometer. We used the Ubiquiti NanoStation M5 as antennas [165]. Each device ran OpenWrt 14.07 and used the IEEE 802.11n standard in the 5 GHz frequency band [166]. To be able to perform mesh-based routing, we used the BATMAN advanced protocol [167].

Figure 3.1 illustrates the planned topology of the deployed IEEE 802.11 mesh network. The locations where network access is required include bars and food stalls, which means they attract a large number of people. Therefore, the nodes were close to other electronic equipment, which often transmits RF signals as well. The height of the nodes did vary, as some could be placed very high, up to 20 meters high, and others were close to ground level. Most of the nodes we deployed were on ground level at about one to three meters height. This height means that

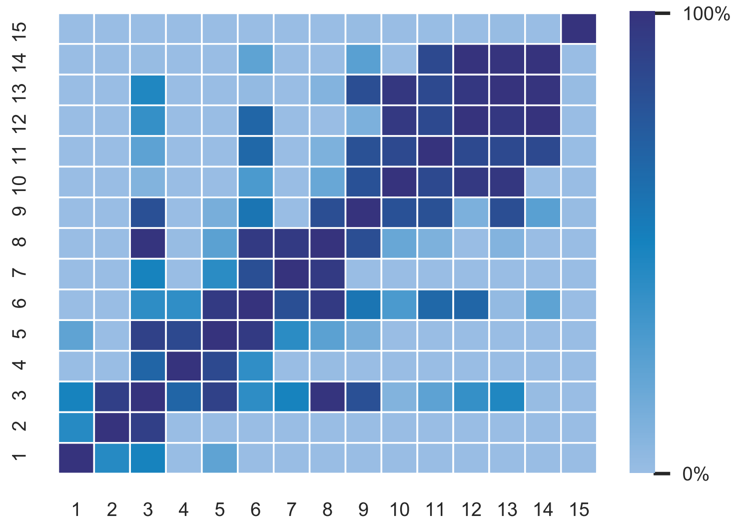


Figure 3.2: Available links in the mesh network.

for most links, both people and user devices such as smartphones potentially cause interference as they were moving between the nodes.

3.2.2 Performance results

Following, we discuss the performance of the IEEE 802.11 mesh network throughout the field trial. We use several metrics to describe the performance. First, we use the percentage of the available and used links, derived from the BATMAN advanced information. Second, we report the RSSI values of different node types. Third, we show the latency and loss of continuously transmitted pings.

Figures 3.2 and 3.3 show the available and used links of the BATMAN advanced protocol as a fraction of the 3-day time zone in which the nodes were online, respectively. When we compare the available links with the initially planned topology in Figure 3.1, we can see significantly more available links than were planned. Especially in the denser clustered area with nodes 10 to 15, most of the nodes can see each other during the entire festival. Although directional antennas are by nature focused in one direction, their angle of radiation usually is wide enough to allow for not completely aligned links. While the availability of our chosen links is rather high, we have a high number of links that have availability from 20 % to 70 %. The availability of a link itself is not sufficient to explain the quality of it. If we have a look at the usage of the links and compare it to the availability, we can see links with high availability, for example, between node 3

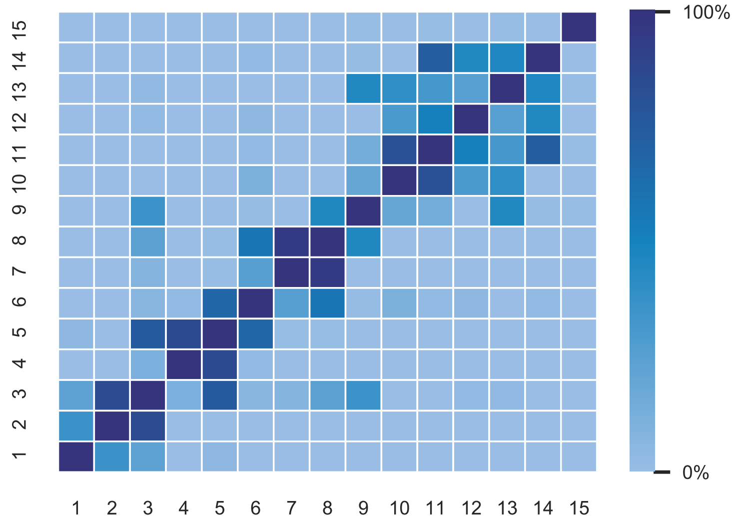


Figure 3.3: Used links in the mesh network.

and node 8, but a low usage (Figure 3.3). The link was simply not reliable enough to be chosen by the BATMAN advanced protocol, although it was available most of the time. The overall medium usage of specific links indicates frequent changes in the topology. The protocol chose other links because the quality of the link decreased. This unreliability is mainly due to too much interference and, therefore, too much packet loss on these links.

We can distinguish between three different types of nodes, depending on where we placed them: (i) Edge nodes, (ii) Indoor nodes, or (iii) Outdoor nodes. The location of edge nodes is close to the edge of the festival ground, and they had, therefore, less interference and fewer visitors around them. Outdoor nodes describe nodes that are placed outside, mainly on top of stages or other high places. They had, therefore, sufficient line of sight with other nodes and did not suffer from the Faraday effect, for example. Indoor nodes mark nodes that are insides stages or other buildings. They usually do not have a line of sight but need to transmit through structures. Moreover, as a stage is typically a massive metal construct, it can act as a Faraday shield.

Figure 3.4 displays the average RSSI for the three node types (Edge, Outdoor, and Indoor) with the standard deviation. There is a clear distinction in the average values, while the standard deviation is very similar for all groups at around 10.5 dBm. Edge nodes have an average of around -66 dBm, which is an acceptable value for IEEE 802.11 systems. Home setups can achieve better values, but the performance of IEEE 802.11 is still satisfactory with -66 dBm. The value for Outdoor nodes is

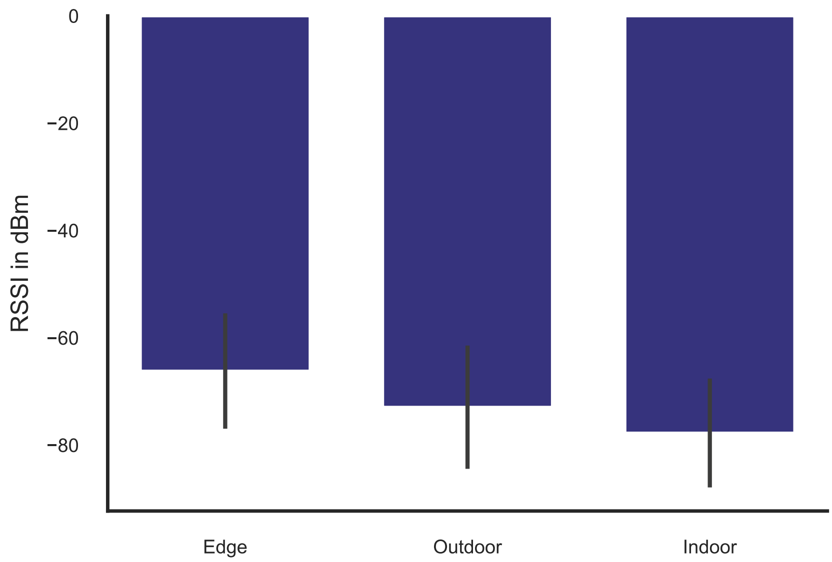


Figure 3.4: The signal strength of different type of nodes.

about 7 dBm less compared to Edge nodes. -73 dBm is already close to the (vendor-specific) threshold, where most IEEE 802.11 systems consider the medium occupied. The significant standard deviation of 10.5 dBm shows that the RSSI is unstable and that often it drops below the threshold, which can lead to a back-off and longer latency as well as packet loss. Indoor nodes have an additional decrease of 5 dBm, which results in an average of -78 dBm. Combined with the high standard deviation of 10.5 dBm, the average value means frequent back-offs and retransmissions due to lost packets. The deviation also shows that the interference was not continuous, but continually changing.

Figure 3.5 shows the latency of continuous pings. As expected, the latency increases with each hop count, but flattens with more than three hops, likely because the last hops are very close by and have a good connection between each other. The high latency of 2.5 s for a single hop shows that the effects of interference through the environment have a significant impact. The latency makes any type of bidirectional communication nearly impossible. For larger hop counts, the latency increases to up to 10 seconds and more. The standard deviation is in all cases very high, which creates extreme outliers but also partially acceptable latency. For one-hop distance, around 50 % of the pings were below 100 ms, which is still acceptable for most applications. This percentage decreased down to 30 % for two hops and further to 10 % for more than two hops. Figure 3.6 shows the accompanying loss to the latency of the ping. The loss starts for direct neighbors at around 50 %, which increases to nearly 80 % for two hops and tops out at 90 % for more than two hops.

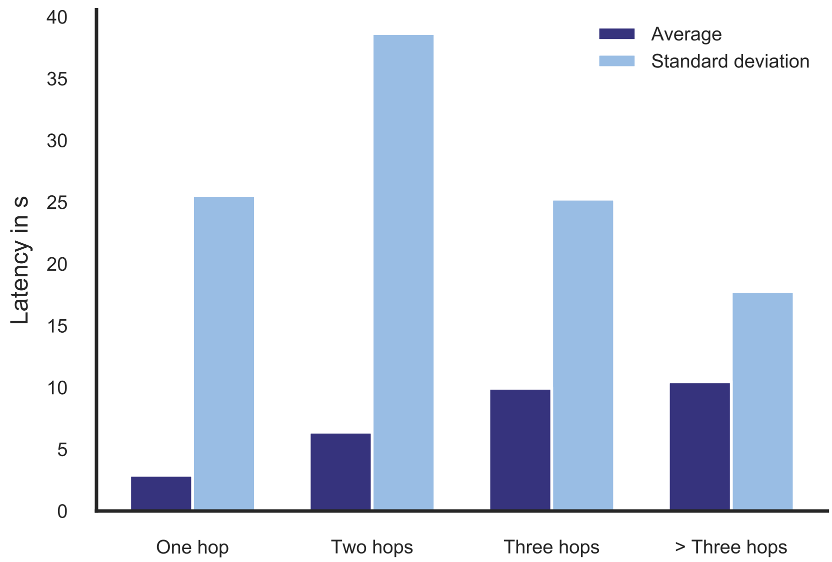


Figure 3.5: The latency obtained through pings for different hop counts.

The high loss indicates that significant interference (RF or otherwise) is present, and packets collide with this interference. Collisions cause retransmissions and back-offs in the CSMA/CA scheme, which results in the high latency that we have seen. The extremely high values indicate that there are many interfering sources. While other IEEE 802.11 devices are present and can cause interference with probe requests, other sources are not directly accounted for [50].

3.3 IEEE 802.11 performance with varying external influence

After we presented the large-scale deployment, we will go into more detail about the influence of external interference in this section. To be more precise, we consider a setup in a lab environment with generated interference and where we can assume that additional interference from other devices is as minimal as possible. First, we characterize the behavior of the interfering source, and then we present the setup and its results.

3.3.1 Characterizing an interfering source

To correctly model the interfering source, we take an on/off process with exponentially distributed on and off periods as a basis. The interruptions of the medium

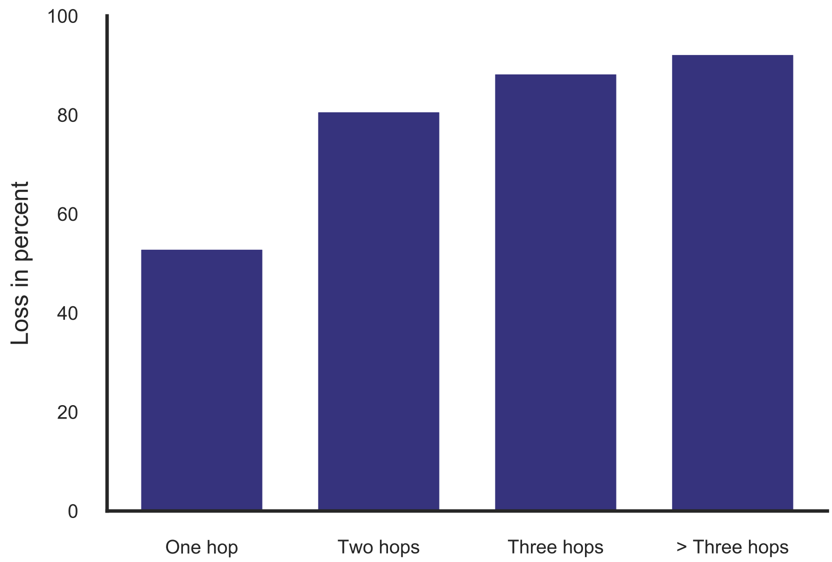


Figure 3.6: The packet loss of pings for different hop counts.

access by the interfering source, which generates energy above the ED threshold, occur according to a Poisson process with rate ν. We can model different sources in this form, for example, a microwave oven or a baby phone [168]. We assume that during an ongoing interruption, no new interruptions occur. The variable u denotes the random variable representing the length of the interruptions. We assume that u is exponentially distributed with mean $E[u]$. With the Poisson assumption in mind, the average time that the interfering source is inactive is given by $\frac{1}{\nu}$. In contrast, the fraction of the time the interfering source is active is given by:

$$p_a = \frac{E[u]}{E[u] + \frac{1}{\nu}} \tag{3.1}$$

3.3.2 Experimental setup

For our experimental study, we use the w-ilab.t [1] lab facility, a large-scale emulation platform with wireless nodes allowing extensive experiments. We use configurations of 15, 20, and 25 stations on the 5 GHz band with IEEE 802.11a consisting of Zotac Zbox ID10 with IEEE 802.11n capable wireless cards. We connected all stations to a single AP (PC Engines APU 1d4 with an IEEE 802.11ac enabled wireless card), and a test includes transmitting packets for 60 seconds with a repetition of 5 times for each configuration. To generate interference according to the previously defined model, we installed an SDR of type USRP N210. Figure 3.7 displays the setup.

[1] http://doc.ilabt.imec.be/ilabt-documentation/index.html

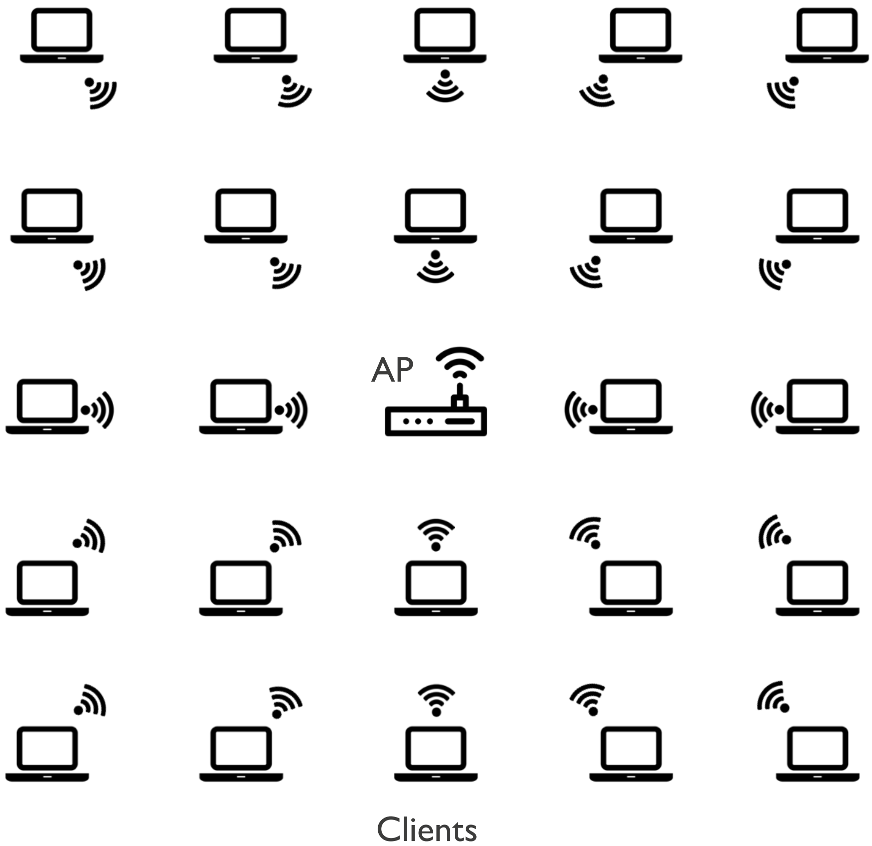


Figure 3.7: Setup of experiment with central access point and clients around it.

We use two modes of interference occurrence in the experiment: low occurrence with $\frac{1}{\nu}$ equal to $9 \cdot 10^{-4}$ s and high occurrence with $\frac{1}{\nu}$ equal to $1.8 \cdot 10^{-4}$ s. We set the duration of interference $E[u]$ to three different modes: low ($9 \cdot 10^{-5}$ s), medium ($4.5 \cdot 10^{-4}$ s), and high ($9 \cdot 10^{-4}$ s). The packets have a size of 1500 bytes, and we sent them at a fixed bit rate of 54 Mbps. The sending rate of packets per second had a minimum of 25 and a maximum of 200 packets per second with a step size of 25 packets per second. We used a continuous packet source to generate packets on the MAC layer according to a Poisson process. The queue length is given by $K = 64$.

3.3.3 Performance results

As discussed in Section 3.3.1, this combination of ν and $E[u]$ leads to different levels of airtime usage of the interfering source. We chose these values in such a way that the airtime usage varies from minimal (9.1 %) to a significant amount

Table 3.1: Airtime occupation of interfering source for different probabilities and timeslots used in the presented experiments.

$\frac{1}{\nu}$ \ **E[v]**	$\mathbf{9 \cdot 10^{-5}}$	$\mathbf{4.5 \cdot 10^{-4}}$	$\mathbf{9 \cdot 10^{-4}}$
$\mathbf{9 \cdot 10^{-4}}$	9.1 %	33.33 %	50 %
$\mathbf{1.8 \cdot 10^{-4}}$	33.33 %	71.4 %	83.33 %

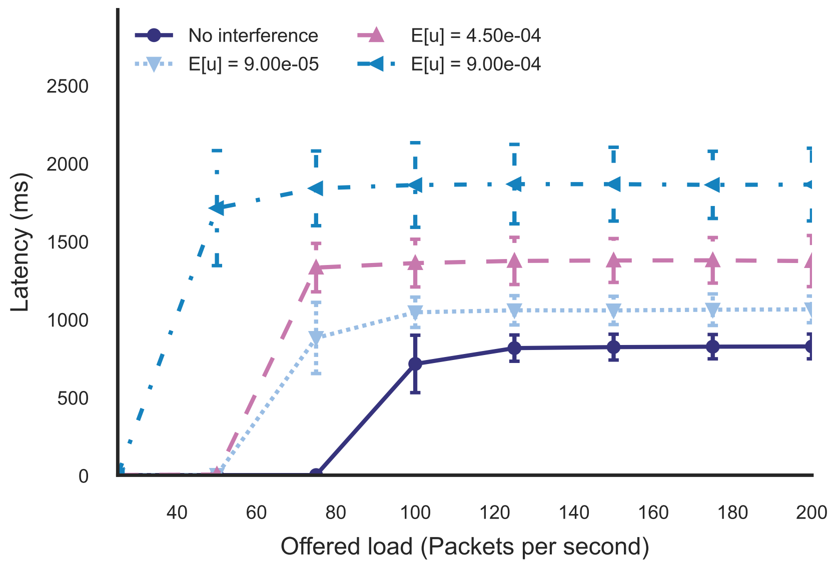


Figure 3.8: Latency without and with interference with low occurrence interference and 25 stations.

(83.33 %). Table 3.1 illustrates the relationship between occurrence and duration.

In the following graphs, we consider three parameters: the duration of the interference $E[u]$, the arrival rate of interference ν, and the number of stations N. All three different parameters have an impact on the saturation point, the point where the medium is fully used and therefore all participants are heavily competing for airtime, and the maximum delay.

Figure 3.8 shows the latency for variable $E[u]$. We fixed the interfering source to low interference occurrence, while we fixed the number of stations to $N = 25$. For once, we observe that for a fixed value of interference occurrence ν and an increase in the duration of the interference, the system reached saturation point earlier as the interfering source occupies an increasing amount of the airtime. At the same time, we observe an increase of the average packet latency under saturation as well as stations have to wait longer until they can transmit.

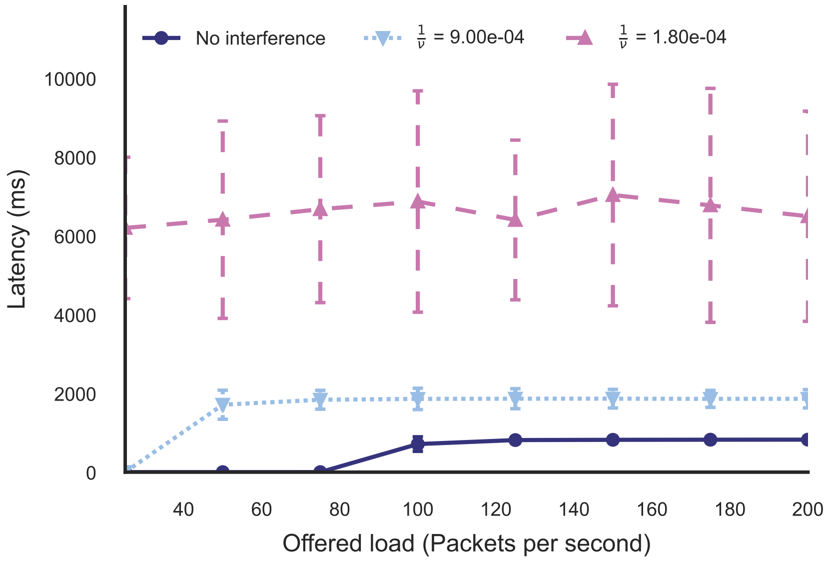


Figure 3.9: Latency without and with interference with high interference duration and 25 stations.

Figure 3.9 shows the impact of an increased packet rate and different values for the interference occurrence ν. In this case, we set the number of stations to 25 and the interference duration $E[u]$ to high. Similar to the previous case, an increase in occurrence shifts the saturation point towards lower load value due to a higher airtime usage of the interfering source. At the same, the average packet latency under saturation increases as well, in this case, much more drastically than previously. Remark the large confidence interval for high interference occurrence. This test yielded a meager number of successfully transmitted packets, and therefore the variation is much higher.

In the last figure, Figure 3.10, the number of stations N varies from 15 to 20 to 25 while we set the interference duration $E[u]$ to high and the interference occurrence ν to low. Even when no interference is present, the saturation point as well as the maximum latency shift with an increasing number of stations, which is expected behavior due to the back-off mechanism. We can observe a similar behavior when interference is present. The interference occupies a part of the airtime, and the existing stations have to compete for the remaining airtime.

Summarizing, we can see that a non-IEEE 802.11 interference source can heavily degrade performance. The system reaches saturation much earlier, and latency increases significantly due to competition in airtime usage. The latency can reach up to a maximum of 10 s, which means every type of connection suffers greatly. The decrease gets worse with more interference present in general. However, there

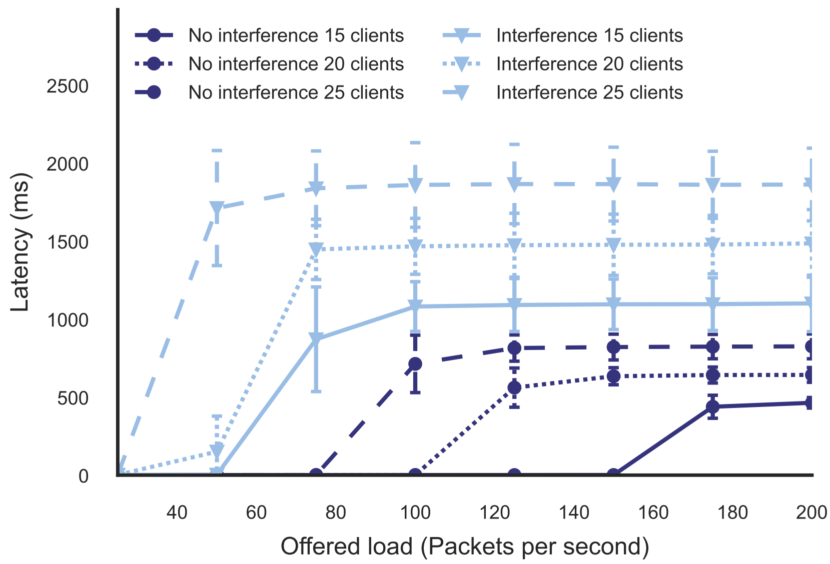


Figure 3.10: Latency without and with interference with low interference occurrence and high interference duration.

is an indication that different pairs of values of occurrence ν and duration $E[u]$, may lead to the different behavior of the average packet latency (both in absolute value and saturation point). However, the fraction of time the source of the interference is active is the same for these pairs.

3.4 Estimation of delay with interference

In this section, we explore the possibility to derive the latency of an IEEE 802.11 system with interference from an IEEE 802.11 system without interference. The base system without interference can be measurements from a real deployment or based on analytical models, as presented in Section 2.2.2. We use the same model for the interference source as in Section 3.3.1, as well as the same validation setup as in Section 3.3.2.

3.4.1 Deriving average latency in a system with interference from a system without interference

An interfering source has three major effects on the operation of the IEEE 802.11 MAC protocol.

The first is based on the ED function of an IEEE 802.11 station, which senses for energy on the channel before it tries to send a packet. When the interfering source becomes active, the station detects energy on the channel and defers from transmitting a packet for $E[u]$ seconds on average. Next, when the interfering source becomes active at the time the device transmits a packet, the packet collides with the signal of the interfering source and is lost. Not only a retransmission of the packet is the result, but it also adds additional latency in the form of a doubled contention window for the next back-off phase. Third, stations with a packet at the head of the queue during the time the interfering source is active will sense the medium busy. As soon as the interfering source stops transmitting, the stations will enter in a back-off phase. If a station is in the back-off phase during the activity of the interfering source, it has to stop the process. It then needs to wait until the medium is considered free again.

We will first take the unavailability of the medium and the increased CW into account. Consider an IEEE 802.11 network with N stations, which are equally loaded. We model a station as a finite capacity single server queue with Poisson input with rate λ where the service time equals the sum of the IEEE 802.11 access latency and the transmission time itself. We will model the activity of the interfering source as service interrupts. Computing the latency in this M/M/1/K queue with service interruptions will result in the average packet latency of a station.

For a random variable d, we denote $D(t)$ its cumulative distribution, respectively $D^*(s)$ its Laplace-Stieltjes Transform (LST). $E[d]$ denotes its mean value. The service time of a packet consists of two major parts, access latency and the transmission time of the packet itself. We denote the service time by b_{ni}, respectively, b_{wi}, in the system without interference, respectively, the system with interference. Let b denote the transmission time of a packet. We make the additional assumption that both b_{ni} and b_{wi} are exponentially distributed. This approach turns the model of a station in a system with and without interference into an M/M/1/K queue. Let d_{ni} and d_{wi} be the packet latency respectively in the system without interference and in the system with interference.

First, we derive a formula for the LST of the service time in a system with interference $B^*_{wi}(s)$, as a function of the LST of the packet latency in a system without interference $B^*_{ni}(s)$. We follow reasoning similar to the one by Fiems et al., where the service interruptions are the active periods of the interfering source [169]. We consider three cases, depending on the start of the first time the interfering source becomes active after a packet starts its service concerning the different components of this service time.

First case: The interfering source does not become active during the service time. In this case, the service time in the system with interference is the same as without interference.

Second case: The interfering source becomes active during the access latency of a packet. In this case, the time the interfering source is active needs to be added to the service time in a system without interference.

Third case: The interfering source becomes active during the transmission time of a packet. In this case, not only the time the interfering source is active needs to be added to the service time, but also an additional time since the MAC protocol reacts to this interrupt of the transmission by doubling the contention window. Let a be the random variable representing the additional access latency of a packet whose transmission was interrupted by the interfering source. Note that this case includes the added latency of the paused back-off mechanism when a packet is at the head of the queue.

Assume that the service time in the system without interference is given by x.

1. No interrupt by the interfering source occurs during the service time, which happens with probability $e^{-\nu x}$. In this case
$$B^*_{wi}(s|x) = e^{-(\nu+s)x} \tag{3.2}$$
where $B^*_{wi}(s|x)$ denotes the LST of b_{wi}, given that the service time in the system without interference is x.

2. An interrupt by the interfering source occurs during the access latency (i.e., during $[0, x-b[$). This interrupt happens with probability $1 - e^{-\nu(x-b)}$. In this case
$$B^*_{wi}(s|x) = \frac{\nu}{\nu+s} \cdot V^*(s) \cdot B^*_{wi}(s) \cdot (1 - e^{-(\nu+s)(x-b)}) \tag{3.3}$$

3. An interrupt by the interfering source occurs during the transmission time (i.e., during $[x-b, x[$). This interrupt happens with probability $e^{-\nu(x-b)} - e^{-\nu b}$. In this case
$$B^*_{wi}(s|x) = \frac{\nu}{\nu+s} \cdot V^*(s) \cdot A^*(s) \cdot B^*_{wi}(s) \cdot (e^{-(\nu+s)(x-b)} - e^{-(\nu+s)x}) \tag{3.4}$$

Combining the three cases, we obtain

$$\begin{aligned} B^*_{wi}(s|x) = e^{-(\nu+s)x} &+ \frac{\nu}{\nu+s} \cdot V^*(s) \cdot B^*_{wi}(s) \cdot (1 - e^{-(\nu+s)(x-b)}) \\ &+ \frac{\nu}{\nu+s} \cdot V^*(s) \cdot A^*(s) \cdot B^*_{wi}(s) \cdot (e^{-(\nu+s)(x-b)} - e^{-(\nu+s)x}) \end{aligned} \tag{3.5}$$

Integrating over all possible service times x leads to

$$\begin{aligned} B^*_{wi}(s) = B^*_{ni}(\nu+s) &+ \frac{\nu}{\nu+s} \cdot V^*(s) \cdot B^*_{wi}(s) \cdot (1 - e^{(\nu+s)b} \cdot B^*_{ni}(\nu+s)) \\ &+ \frac{\nu}{\nu+s} \cdot V^*(s) \cdot A^*(s) \cdot B^*_{wi}(s) \cdot (e^{(\nu+s)b} \cdot B^*_{ni}(\nu+s) - B^*_{ni}(\nu+s)) \end{aligned} \tag{3.6}$$

Since
$$E[b_{wi}] = -\frac{d \cdot B^*_{wi}(s)}{ds}|_{s=0} \tag{3.7}$$

we obtain that

$$E[b_{wi}] = \frac{1}{\nu \cdot B^*_{ni}(\nu)} \cdot (1 - B^*_{ni}(\nu)) \cdot (1 + \nu \cdot E[v]) + E[a] \cdot (e^{\nu b} - 1) \tag{3.8}$$

Given the assumption that b_{ni} is exponentially distributed, we obtain

$$E[b_{wi}] = E[b_{ni}] \cdot (1 + \nu \cdot E[v]) + E[a] \cdot (e^{\nu b} - 1) \tag{3.9}$$

Let us compute $E[a]$. The probability that the interfering source becomes active while a packet is being transmitted is given by

$$\int_0^b \nu e^{-\nu b} dt = 1 - e^{-\nu b} \tag{3.10}$$

with b being the time needed to transmit a packet.

Hence, the probability that the interfering source becomes active during each of the consecutive i retransmissions of a packet and not during the $(i+1)^{\text{th}}$ is given by

$$(1 - e^{-\nu b})^i \cdot e^{-\nu b} \tag{3.11}$$

Assuming $CW_{min} = 63$, $CW_{max} = 1023$ and for the 5$^{\text{th}}$ and 6$^{\text{th}}$ retransmission $CW = 1023$, the value of $E[a]$ is given by

$$E[a] = \sum_{i=1}^{5} 2^{4+i} \cdot (1 - e^{-\nu b})^i \cdot e^{-\nu b} + 2^9 \cdot [1 - \sum_{i=0}^{5} (1 - e^{-\nu b})^i \cdot e^{-\nu b}] \tag{3.12}$$

To take the increased latency of the paused back-off mechanism into account when the packet is at the head of the queue, we consider the behavior of stations when the interfering source becomes active similar to their behavior when other stations transmit packets. Therefore, we express the activity of the interference source in terms of packet transmissions. The fraction of the time, the interference source is active is given by p_a. Let n_{if} be the number of packets per second that a device could send during an active period of the interference source. Then

$$n_{if} = \frac{p_a}{b + c} \tag{3.13}$$

with c being the transmission time of an acknowledgment. For the latency computation with interference, the packet arrival rate is given by

$$\lambda_a = \lambda + \frac{n_{if}}{N} \tag{3.14}$$

and will be used to derive the performance measures for the system with interference.

We assume that the number of stations N and the packet arrival rate λ are given To compute the average packet latency in a system with interference $E[d_{wi}]$, given the average packet latency in the system without interference $E[d_{ni}]$, using the relationship between $E[b_{wi}]$ and $E[b_{ni}]$ as established above. Let K be the length of the MAC queue in a station. The random variables l_{ni} and l_{wi} denote the number of packets in a station without and with interference. Let

$$\rho_{ni} = \lambda \cdot E[b_{ni}] \tag{3.15}$$

respectively,

$$\rho_{wi} = \lambda_a \cdot E[b_{wi}] \tag{3.16}$$

be the load of a station without, respectively, with interference. Assume that we know the average latency $E[d_{ni}]$ of a packet in a system without interference and that a station in the system without interference is modeled as an M/M/1/K queue. Given Little's law, the average number of packets in the station is given by

$$E[l_{ni}] = \lambda_{eff_{ni}} \cdot E[d_{ni}] \tag{3.17}$$

where

$$\lambda_{eff_{ni}} = \lambda_a \cdot (1 - \frac{1 - \rho_{ni}}{1 - \rho_{ni}^{K+1}} \cdot \rho_{ni}^{K}) \tag{3.18}$$

is the effective packet arrival rate in the system without interference.

Then, the average number of packets, $E[l_{ni}]$, is also given by the formula

$$E[l_{ni}] = \rho_{ni} \cdot \frac{1 - (K+1) \cdot \rho_{ni}^{K} + K \cdot \rho_{ni}^{K+1}}{(1 - \rho_{ni}) \cdot (1 - \rho_{ni}^{K+1})} \tag{3.19}$$

From Equation 3.17 and Equation 3.19, we derive the value of ρ_{ni}. This derivation leads to

$$E[b_{ni}] = \frac{\rho_{ni}}{\lambda_a} \tag{3.20}$$

Now it is possible to compute $E[b_{wi}]$ using Equation 3.9. Once $E[b_{wi}]$ is known, it is possible to compute

$$\rho_{wi} = \lambda \cdot E[b_{wi}]. \tag{3.21}$$

Using Equation 3.19 for the system with interference, we derive $E[l_{wi}]$ and applying Little's formula leads to the average latency in a system with interference

$$E[d_{wi}] = \frac{E[l_{wi}]}{\lambda_{eff_{wi}}} \tag{3.22}$$

with

$$\lambda_{eff_{wi}} = \lambda \cdot (1 - \frac{1 - \rho_{wi}}{1 - \rho_{wi}^{K+1}} \cdot \rho_{wi}^{K}) \tag{3.23}$$

3.4.2 Evaluation

Similar to the initial analysis of the impact of an interfering source in the previous section, there are two significant elements to assess the accuracy of the model, the saturation point and the maximum average latency when the system is saturated.

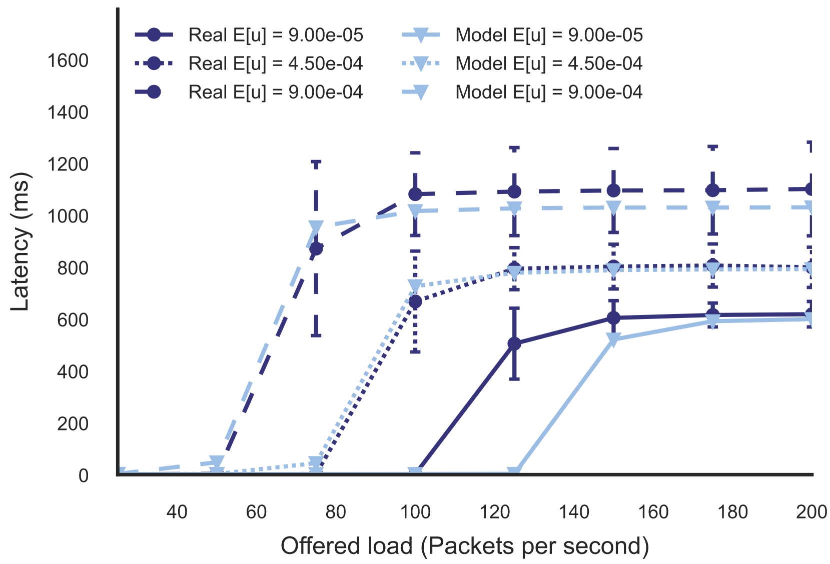


Figure 3.11: Latency with low interference occurrence and 15 stations.

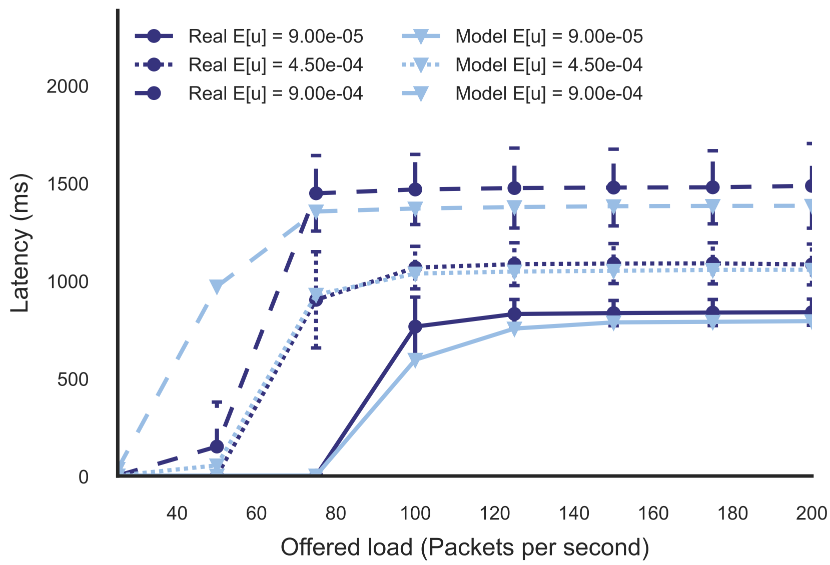


Figure 3.12: Latency with low interference occurrence and 20 stations.

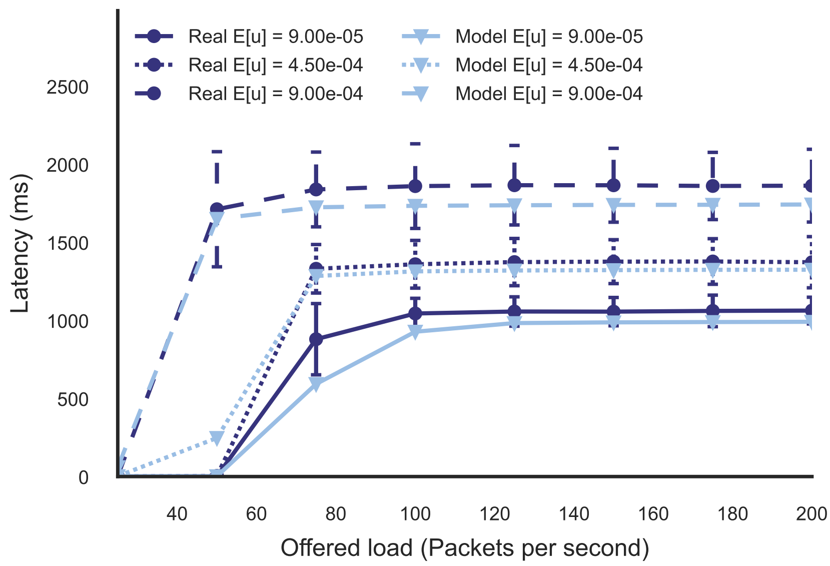


Figure 3.13: Latency with low interference occurrence and 25 stations.

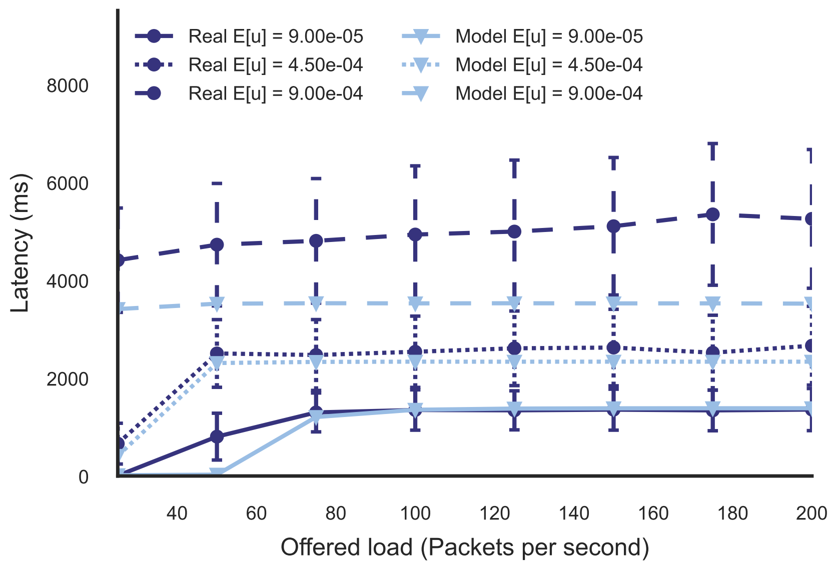


Figure 3.14: Latency with high interference occurrence and 15 stations.

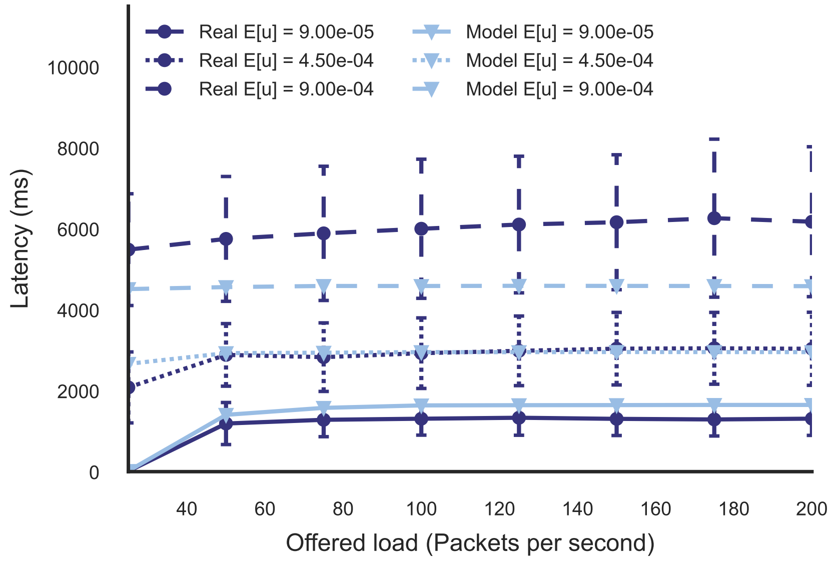


Figure 3.15: Latency with high interference occurrence and 20 stations.

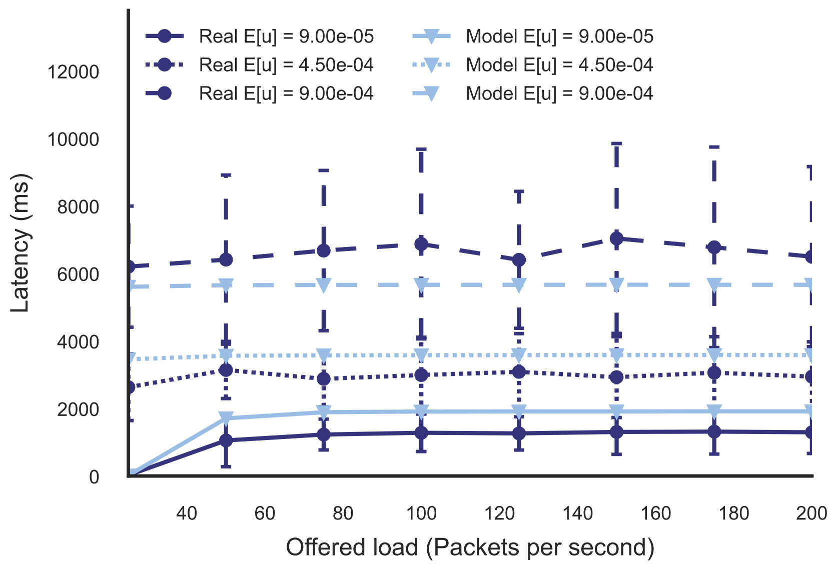


Figure 3.16: Latency with high interference occurrence and 25 stations.

Saturation is the state when the wireless network reaches its maximum capacity and depends on the number of stations, the number of packets per station, and the characteristics of the interfering source. Note that, as we only have measurement results with steps of 25 packets per second, we used interpolation when applying Equation 3.17.

Low occurrence: Figures 3.11, 3.12, and 3.13 present the graphs for 15, 20, and 25 stations and low interference occurrence. In 8 out of 9 cases, the model matches the saturation point accurately. Only in the case of 15 stations and with the shortest duration, the model predicts the saturation point too early. The latency prediction at saturation is within 6-7 % of the average latency, while a higher number of stations lead to higher accuracy.

High occurrence: Figures 3.14, 3.15, and 3.16 show similar results for high interference occurrence. The model matches the saturation point accurately in 8 out of 9 cases, again except for 15 stations and the shortest duration. The accuracy of the latency at saturation is within 13-50 %. The high duration case is an outlier with any number of stations. The airtime usage of the interfering source amounts to 83 % of the available airtime in this case, and the difficulty of prediction stems from the low amount of successful packets, as can be seen by the confidence interval.

The results show that the accuracy is high and that one can use our proposed method in further network management.

3.5 Conclusion

In this chapter, we first presented the results of a mesh deployment on a large-scale event that provides a challenging environment. The results show that IEEE 802.11 systems can be profoundly affected by interference sources, such as other IEEE 802.11 devices, but also from external influence, such as RF equipment. Following, we explored the latter option in more detail, showing that external interference can have a significant impact on IEEE 802.11 systems, which leads to high latency. We then provided a fast and convenient computation to derive the latency in an IEEE 802.11 system with interference from a system without interference that can reach high accuracy. The computation requires information from a base system, though, and we will explore more possibilities to describe the performance in the next chapter.

4 Latency prediction with an analytical model in IEEE 802.11 systems with interfering source

The contributions presented in this chapter are based on the publication titled *"An Analytical model for IEEE 802.11 with non-IEEE 802.11 interfering source"*.

4.1 Introduction

In the previous chapter (Chapter 3), we explored the behavior of IEEE 802.11 systems under the influence of external interference and showed that it could have a significant impact. We also provided a fast way to compute the expected performance from a base system without interference. This requirement limits the model in its expressiveness of the complete behavior of an IEEE 802.11 system and can not give answers for all possible behaviors. In this chapter, we extend the expressiveness by providing a full model that describes all behavior of an IEEE 802.11 system based on a Markov chain.

As we showed in Section 2.2.2, modeling the performance of IEEE 802.11 systems is as old as the technology itself. Nearly all latency and throughput models for IEEE 802.11 base themselves on the model of Bianchi [56]. The model of Pham [7], on which we base our model, is also based on the model of Bianchi [56]. It extends the original model from the saturated to the unsaturated case by introducing several new states. There are two main differences, both represented in Figure 4.1. First, it includes the case that after the last re-transmission,

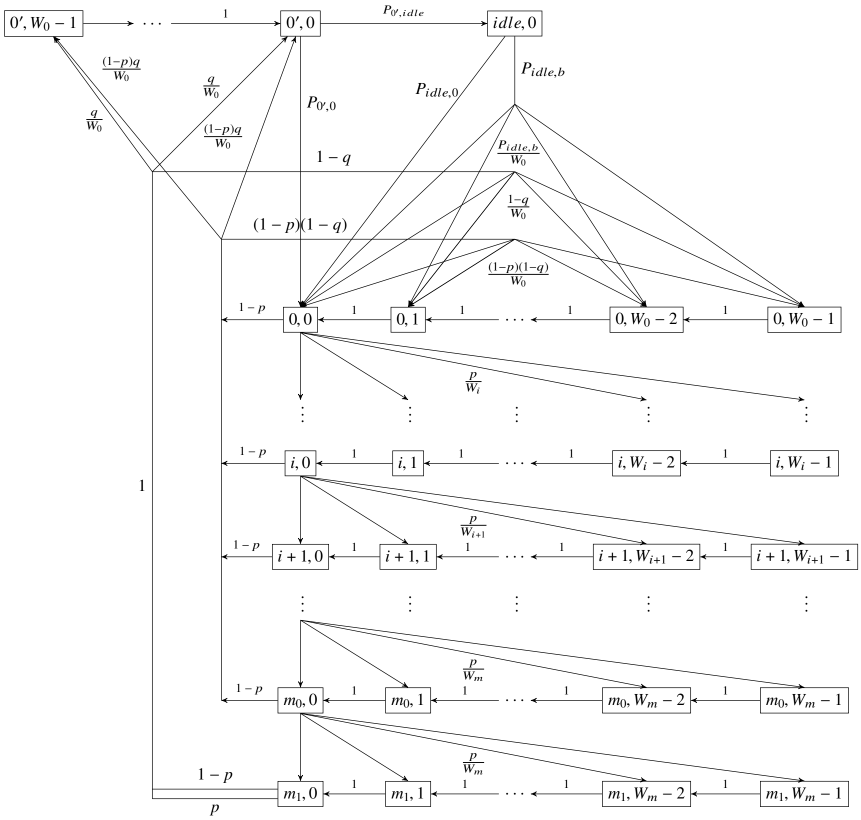


Figure 4.1: Transition probabilities $\{s(t), b(t)\}$ of the Markov Chain.

the packet is dropped and not transmitted, including the post-back-off, which also consumes service time. Second, the unsaturated case, i.e., where the packet queue may be empty, is modeled as well.

Introducing a transition from the last back-off stage to the post-back-off mechanism covers the first extension. The transition is similar to the standard back-off mechanism but is handled with the minimum contention window and is only applied when the channel has been idle before, and no packet is ready to transmit. This approach preserves the DCF algorithm and avoids a longer back-off as soon as a packet is available for transmission. In Figure 4.1, these are the states $(0', j)$, with $j = 0, \ldots, W_0 - 1$. Introducing an idle state, $S_{idle,0}$, that can be entered if there is no packet at the head of the queue handles the unsaturated case. If a packet becomes available, depending on the state of the channel, it can be transmitted immediately, with probability $P_{idle,0}$, or it needs to enter the normal back-off phase with $P_{idle,b}$.

We will then use the same interfering source as presented in Section 3.3.1, but with a slightly different formulation in Section 4.2. With Pham's model as a

Table 4.1: Overview of possible interfering devices with probability and duration for available parameters and devices

Device	p_{if}	T_{if}
Microwave Gollakota et al.	0.0015	1111
Microwave Hithnawi et al.	0.0006	555
Bluetooth	0.05	41
Cordless phone Gollakota et al.	1.0	Call duration

basis, we introduce several cases that cover the influence of the interfering source in Section 4.3 as well as use a more advanced model for the packet queue in Section 4.4. We verify the model through a real deployment as well as the exploration of different parameters in Section 4.5.

4.2 The interfering source and its motivation

Consider a slotted system where the length of a slot corresponds to the time slot of the IEEE 802.11 system under study, denoted by σ. To make the model more manageable, we assume that the interfering source only becomes active at the start of a slot. Later experiments do not follow this assumption but still achieve high accuracy. We use the following model for the interfering source. Let

- p_{if} : the probability that at the start of a slot, the interfering source becomes active.
- T_{if} : the average duration in multiples of σ that the interfering source is active above the threshold (and hence detected by the stations). We assume that this duration expressed in multiples of the interval σ is geometrically distributed. As a consequence of these assumptions, there is always at least one time-interval σ between two active periods of the interfering source.

The fraction of the time the interfering source is active is given by:

$$P_a = \frac{T_{if}}{T_{if} + \frac{1}{p_{if}}} \tag{4.1}$$

This general and straightforward model of an interference source allows us to model different sources of interference, independent of the specific communication technology, RF interference, or devices such as microwaves. The parameterization entirely depends on which type of source one wants to model.

Gollakota et al. show that a microwave exhibits a periodical ON-OFF pattern, where the ON period typically lasts for about 10 ms and the OFF period for 6 ms [31]. Hithnawi et al. confirm this behavior, in which case the respective durations are 5 ms ON and 15 ms OFF [170]. Gummadi et al. also confirms the results, but do not provide a specific model for the interfering source besides a simple Signal-to-Interference-plus-Noise Ratio (SINR) based one [41].

Also, the presence of a Bluetooth network may interfere with an IEEE 802.11 network. Bluetooth uses a *Frequency Hopping* scheme, and interference may only occur when there is an overlap in time and frequency. Several papers study this behavior, among which is the work of Conti et al. and Jung-Hyuck Jo and Jayant [171, 172]. The duration of a Bluetooth slot is 625 μs, while the transmission time within a time slot is 366 μs. In order to have interference, there should be a frequency overlap. When, e.g., using IEEE 802.11b, the probability that the Bluetooth piconet hops into the IEEE802.11 Direct-Sequence Spread Spectrum (DSSS) passband is 0.25. If the Bluetooth piconet carries a telephone conversation, then 33 % of the time slots are used to transmit a packet. Hence the probability that an ON period starts is given by 0.05, and the duration is 366 μs.

According to Gollakota et al., digital cordless telephones continuously transmit packets, and therefore the channel is never free for the duration of the call (hence OFF = 0 ms) [31].

Some video monitoring systems use several channels in the 2.4 GHz band. If the camera does not transmit continuously, but one image every x ms, then the interference will exhibit an ON-OFF pattern. Hithnawi et al. describe a wireless camera that hops within the frequency range [2.42 GHz - 2.45 GHz] [170]. Figure 3e, in their work, clearly shows the ON-OFF pattern of this camera. The digital Frequency-Hopping Spread Spectrum (FHSS) cordless phone in that paper exhibits a similar ON-OFF pattern.

Shuaib et al. investigate the interference of Zigbee (IEEE 802.15.4) on IEEE 802.11g (and vice versa) [173]. There is no model given for the Zigbee interferer, so we do not know whether we can use the ON-OFF model. Huo et al. investigate the cross-interference of Zigbee and IEEE 802.11 as well, but based on a PER model and not a signal interference model [174]. In both cases, a further literature study is required.

Bicakci and Tavli present different physical layer attacks that impact the performance of IEEE 802.11 [175]. While our model can not describe all of them, some require sensing capabilities; our formulation can describe others. Especially the *Hit and Run* tactic they present can be described accurately by our model.

The activity of the interfering source is entirely independent of the operation of the IEEE 802.11 MAC. When the interfering source becomes active while a packet is transmitted, the packet is considered lost, similar to a collision between two packets. If a station has a packet ready for transmission and the interfering source is active, the station will refrain from sending and react as if another station would be active. Like in the case of another IEEE 802.11 station, when the interfering source becomes active, the countdown process of the back-off counter stops until the station does not consider the channel busy any longer when a station is in a back-off phase. This freezing of the counter is the default behavior of an IEEE 802.11 station.

4.3 Analytical model for delay and throughput with interference

In this section, we model the behavior of a tagged station employing a Quasi Birth-Death (QBD) process, namely a finite capacity queue with Poisson arrival process and hyperexponential service time [176, 177]. The input rate is the arrival rate of packets at the tagged station. The service rate should take into account both the packet transmission and the access delay due to the DCF access mechanism of the IEEE 802.11 MAC. Similar to previous works, the DCF access mechanism is modeled as a Markov Chain, as depicted in Figure 4.1. We use an adaptation of the IEEE 802.11 DCF MAC model proposed by Pham [7]. Consider an IEEE 802.11 network with n stations and let for a tagged station at time t

- $b(t)$: the back-off time counter
- $s(t)$: the back-off stage $(0, \ldots, m+1)$

We use similar notations as Pham, which can be seen in Table 4.2 [7].

4.3.1 Derivation of parameters in a system with interfering source

We consider the 2-dimensional stochastic process $\{s(t), b(t)\}$, of which Figure 4.1 shows the transition probabilities. The states (i, j), with $i = 0, \ldots, m+1$ and $j = 0, \ldots, W_i - 1$, correspond to those of the saturated case (see Bianchi and Raptis et al.) [57, 65]. The states $(0', j)$, with $j = 0, \ldots, W_0 - 1$, represent the post-back-off mechanism, and the state $(idle, 0)$ represents the state where the queue of the tagged station is empty. These additional states are needed to model an unsaturated network.

We define the first key parameter for our model, τ. Let τ be the probability that a station starts sending a packet in a randomly chosen time slot. Additionally, we define ω being the probability that during an ongoing packet transmission, the interfering source becomes active, but this does not lead to the loss of a packet, as the FEC mechanism of the Physical Layer (PHY) can correct the corrupted bits due to the collision. The derivation of these first two parameters is based on the steady-state of the Markov chain $s(t), b(t)$, and is computed in the next section. Once p and τ are known it is possible to define the following parameters: $P_{tr(n)}$ and $P_{tr(n-1)}$ being the probabilities that at least one out of n or $n-1$ stations transmits respectively and $P_{s(n)}$ and $P_{s(n-1)}$ being the probabilities that there is a successful transmission out of n or $n-1$ stations respectively. It is clear that

$$\begin{aligned}
P_{tr(n)} &= 1 - (1-\tau)^n \\
P_{tr(n-1)} &= 1 - (1-\tau)^{n-1} \\
P_{s(n)} &= n \cdot \tau(1-\tau)^{n-1} \\
P_{s(n-1)} &= (n-1) \cdot \tau \cdot (1-\tau)^{n-2}
\end{aligned} \tag{4.2}$$

Table 4.2: Summary of parameters

Parameter	Description
p	Probability of packet not received
τ	Probability of station sending in randomly chosen slot
ω	Probability that FEC mechanism can correct the packet
q	Probability of empty queue
$b(i, j)$	Probability of node in back-off state $S_{i,j}$
$b(idle, 0)$	Probability of node in idle state $S_{idle,0}$
$P_{0',0}$	Probability of sending immediately after post-back-off
$P_{idle,0}$	Probability of sending immediately after idle state
$P_{idle,b}$	Probability of sending after back-off after idle state
$P_{0',idle}$	Probability to go into idle state after post-back-off
$P_{tr(n)}$	Probability that at least one out of n nodes transmits
$P_{s(n)}$	Probability that one out of n nodes transmits successfully
$P_{tr(n-1)}$	Probability that at least one out of $n-1$ nodes transmits
$P_{s(n-1)}$	Probability that one out of $n-1$ nodes transmits successfully
P_l	Probability that packet is lost due to overflow in queue
$S_{i,j}$	State of node when $s(t) = i$ and $b(t) = j$
$S_{idle,0}$	State of node when no packet ready for transmission
W	Minimal back-off window size
W_i	Back-off window size at stage $s(t) = i$
σ	Channel idle slot (system slot)
$\widetilde{\sigma}$	Average channel slot time
λ	Average packet arrival rate
μ	Packet processing rate
Q_l	Queue length
δ	Propagation latency
U	Average throughput of station
$E[A]$	Average latency of packet

$P_{0',0}$ describes the probability of moving from state $S_{0',0}$, after the post-back-off, to state $S_{0,0}$, meaning a direct transmission, if there is a packet ready for transmission at the station. If there is no packet, the station moves to state $S_{idle,0}$ with probability $P_{0',idle}$. Similarly, if the station is in state $S_{idle,0}$, a packet arrives, and the channel has been idle for more than DCF Interframe Space (DIFS), the packet is transmitted immediately with probability $P_{idle,0}$. Otherwise, the node enters into a back-off state $S_{0,i}$, $0 \leq i \leq W_0 - 1$ with probability $P_{idle,b}$.

The probabilities for $P_{0',0}$ and $P_{0',idle}$ stay the same as in the work of Pham [7]:

$$P_{0',0} = 1 - e^{\frac{-\lambda W_0 \widetilde{\sigma}}{2}} \tag{4.3}$$

and

$$P_{0',idle} = e^{\frac{-\lambda W_0 \widetilde{\sigma}}{2}} \tag{4.4}$$

Both $P_{idle,0}$, and $P_{idle,b}$ need to be changed to include the interfering source. When the station is in the idle state, then a transition towards state $S(0,0)$ occurs in two cases:

- There is an arrival before the first DIFS period ends, and during that DIFS period, there are no other stations that start to transmit, and the interfering source is not active. I.e., the source was not active at the start of the DIFS period and will not become active during the DIFS period. The probability for this event equals

$$\left(1 - e^{-\lambda DIFS}\right) \cdot \left[(1 - p_{if})(1 - \tau)^{n-1}\right]^{\frac{DIFS}{\sigma}} \tag{4.5}$$

- The first arrival at the tagged station occurs after the first DIFS period ends, and during the time interval, the station was idle. The medium has been sensed idle. I.e., no other station has started to transmit, and the interfering source has not been active. I.e., the source was not active at the start of the idle period and had not become active during that period. In that case, the packet is sent immediately, i.e., a transition to state $S(0,0)$ occurs. The probability for this event equals

$$\begin{aligned} &e^{-\lambda DIFS} \left[(1 - p_{if})(1 - \tau)^{n-1}\right]^{\frac{DIFS}{\sigma}} \Big(1 \\ &- e^{-\lambda\sigma}\Big) \sum_{t=0}^{\infty} e^{-\lambda\sigma t} \left[(1 - p_{if})(1 - \tau)^{n-1}\right]^{t} \end{aligned} \tag{4.6}$$

Hence

$$\begin{aligned} P_{idle,0} = &\left(1 - e^{-\lambda DIFS}\right) \left[(1 - p_{if})(1 - \tau)^{n-1}\right]^{\frac{DIFS}{\sigma}} \\ &+ e^{-\lambda DIFS} \left[(1 - p_{if})(1 - \tau)^{n-1}\right]^{\frac{DIFS}{\sigma}} \Big(1 \\ &\qquad - e^{-\lambda\sigma}\Big) \sum_{t=0}^{\infty} e^{-\lambda\sigma t} \left[(1 - p_{if})(1 - \tau)^{n-1}\right]^{t} \end{aligned} \tag{4.7}$$

and

$$P_{idle,b} = 1 - P_{idle,0} \tag{4.8}$$

The states $S(m, j)$ and $S(m + 1, j)$ with $0 \le j \le W_m - 1$ are the last two stages of the back-off mechanism. In stage $S(m + 1, j)$, the packet is either transmitted or discarded. The CW of the last two stages, W_m and W_{m+1}, both equal the maximum window size of $2^m W_0$.

We denote by q the probability that the transmission queue in the tagged station is empty. We determine this value in Section 4.4.

4.3.2 Derivation of the steady-state probability

In what follows, we describe the steady-state probabilities of the Markov chain as a function of $b_{(0,0)}$. We first define the second key parameter for our model, p. Let p be the probability that the transmission of a packet is not successful. We follow the same reasoning as Pham and obtain the same equation [7]

$$D_1 = \frac{1}{2} \cdot \left[W \cdot \left(\sum_{i=1}^{m} (2p)^i + p \cdot (2p)^m \right) + \frac{p \cdot (1 - p^{m+1})}{1 - p} \right] + q \cdot \left(\frac{2 \cdot P_{0'0} + W + 1}{2} \right) \tag{4.9}$$

$$D_2 = \frac{q \cdot P_{0',idle}(P_{idle,0} + 1)}{P_{idle,0} + P_{idle,b}} + \frac{(W+1) \cdot \left(\frac{q \cdot P_{0',idle} P_{idle,b}}{P_{idle,0} + P_{idle,b}} + (1-q) \right)}{2} \tag{4.10}$$

$$b(0,0) = \frac{1}{D_1 + D_2} \tag{4.11}$$

4.3.3 Derivation of channel slot time with an interfering source

As a next step, we need to calculate $\widetilde{\sigma}$, the average channel slot time. This time includes the average time the channel is idle, or busy with a packet transmission or collision. Besides the fixed times for DIFS and SIFS, we need three other times. The average channel slot time σ, the time for a successful transmission T_s, and the time for a collision T_c. We consider two cases: the basic scheme and the scheme using Request-To-Send/Clear-To-Send (RTS/CTS). These can be derived as follows:

$$T_s = P_L + SIFS + \delta + ACK + DIFS + \delta \tag{4.12}$$

and

$$T_c = P_L + DIFS + \delta \tag{4.13}$$

with

$$P_L = T_{preamble} + T_{signal} + N_{SYM} \cdot \left\lceil \frac{16 + 8 \cdot (P_B + H) + 6}{N_{DBPS_{DATA}}} \right\rceil \tag{4.14}$$

and

$$ACK = T_{preamble} + T_{signal} + N_{SYM} \cdot \left\lceil \frac{16 + 8 \cdot 14 + 6}{N_{DBPS_{CON}}} \right\rceil \tag{4.15}$$

where the transmission time for the preamble, $T_{preamble} = 16\,\mu$s, the transmission time for the signal, $T_{signal} = 4\,\mu$s, and the transmission time for a symbol, $N_{SYM} = 4\,\mu$s in a 20 MHz channel when using the Orthogonal Frequency-Division Multiplexing (OFDM) PHY, which we use later on in our experiments. $N_{DBPS_{DATA}}$, the transmitted bytes per symbol for data, depends on the data rate used as well as $N_{DBPS_{CON}}$, the transmitted bytes per symbol with the control rate. P_B is the packet length in bytes, while H is the length of the header.

When the RTS/CTS mechanism is used, T_s and T_c are the following

$$T_s = RTS + SIFS + \delta + CTS + SIFS + \delta + P_L + \delta + ACK + DIFS + \delta \quad (4.16)$$

and

$$T_c = RTS + DIFS + \delta \quad (4.17)$$

with

$$RTS = T_{preamble} + T_{signal} + N_{SYM} \cdot \left\lceil \frac{16 + 8 \cdot 20 + 6}{N_{DBPS_{CON}}} \right\rceil \quad (4.18)$$

and

$$RTS = T_{preamble} + T_{signal} + N_{SYM} \cdot \left\lceil \frac{16 + 8 \cdot 14 + 6}{N_{DBPS_{CON}}} \right\rceil \quad (4.19)$$

Let parameter k be:

$$k = \left\lceil \frac{T_s}{\sigma} \right\rceil \quad (4.20)$$

and l respectively:

$$l = \left\lceil \frac{T_c}{\sigma} \right\rceil \quad (4.21)$$

denoting the number of time slots for T_s and T_c.

To calculate the average channel slot time $\widetilde{\sigma}$, we need to take the presence of the interfering source into account. The average channel slot time $\widetilde{\sigma}$ consists of the following components:

- No other packet is sent, and the interfering source does not become active at the start of this slot. In that case, the contribution to $\widetilde{\sigma}$ equals:

$$(1 - P_{tr(n-1)}) \cdot (1 - p_{if}) \cdot \sigma \quad (4.22)$$

- The interfering source becomes active at the start of this slot, and the tagged station considers the channel busy. In that case, the contribution to $\widetilde{\sigma}$ equals:

$$p_{if} \cdot (T_{if} + \sigma) \quad (4.23)$$

- A packet is sent successfully; thus, the interfering source does not become active during $k + 1$ time-intervals of length σ or does become active, but the FEC solves the problem. In that case, the contribution to $\widetilde{\sigma}$ equals:

$$\begin{aligned} &P_{s(n-1)} \cdot [(1 - p_{if})^{k+1} \cdot (T_s + \sigma) \\ &+ \sum_{j=0}^{k-1} (1 - p_{if})^j \cdot p_{if} \cdot \omega \cdot (T_s - j \cdot \sigma + T_{if} + \sigma)] \end{aligned} \quad (4.24)$$

This results into

$$\begin{aligned} &P_{s(n-1)} \cdot [(1 - p_{if})^{k+1} \cdot (T_s + \sigma) \\ &+ \omega \cdot ((1 - (1 - p_{if})^{k+1}) \cdot (T_s + T_{if} + \sigma) \\ &- \sigma \cdot (1 - p_{if}) \cdot \frac{1 - (1 - p_{if})^{k-1} \cdot (1 + (k-1) \cdot p_{if})}{p_{if}})] \end{aligned} \quad (4.25)$$

- A packet is sent successfully, and the interfering source does not become active during transmission. However, it becomes active in the time-interval immediately following the successful transmission, and the tagged station considers the channel busy. In that case, the contribution to $\widetilde{\sigma}$ equals:

$$P_{s(n-1)} \cdot (1 - p_{if})^k \cdot p_{if} \cdot (T_s + T_{if} + \sigma) \tag{4.26}$$

- A packet is sent but involved in a collision with another packet we assume that the interfering source does not become active during the interval of length T_c. The tagged station considers the channel busy. In that case, the contribution to $\widetilde{\sigma}$ equals:

$$P_{tr(n-1)} \cdot \left(1 - \frac{P_{s(n-1)}}{P_{tr(n-1)}}\right) \cdot (1 - p_{if})^{l+1}(T_c + \sigma) \tag{4.27}$$

- A packet is sent but involved in a collision with another packet; we assume that the interfering source does not become active during the interval of length T_c. Moreover, the interfering source becomes active in the time-interval immediately following the time-interval T_c. The tagged station considers the channel busy. In that case, the contribution to $\widetilde{\sigma}$ equals:

$$P_{tr(n-1)} \cdot \left(1 - \frac{P_{s(n-1)}}{P_{tr(n-1)}}\right) \cdot (1 - p_{if})^l \cdot p_{if} \cdot (T_c + T_{if} + \sigma) \tag{4.28}$$

- A packet is sent but involved in a collision with the interfering source that can not be solved by the FEC mechanism (the interfering source does not become active at the start of the time-interval because Equation 4.23 covers this). The tagged station considers the channel busy. In that case, the contribution to $\widetilde{\sigma}$ equals (remark that Equation 4.26 covers the case $n = k$):

$$P_{s(n-1)} \cdot \left(\sum_{j=1}^{k-1}(1 - p_{if})^j \cdot p_{if} \cdot (1 - \omega) \cdot (j \cdot \sigma + T_{if} + \sigma)\right) \tag{4.29}$$

- A packet is sent but involved in a collision with another packet and in a collision with the interfering source (the interfering source does not become active at the start of the time-interval because Equation 4.23 covers this), and the tagged station considers the channel busy. In that case, the contribution to $\widetilde{\sigma}$ equals (remark that Equation 4.28 covers the case $n = l$):

$$\begin{aligned} &P_{tr(n-1)} \cdot \left(1 - \frac{P_{s(n-1)}}{P_{tr(n-1)}}\right) \\ &\cdot \left(\sum_{n=1}^{l-1}(1 - p_{if})^n \cdot p_{if} \cdot (n \cdot \sigma + T_{if} + \sigma)\right) \end{aligned} \tag{4.30}$$

Subsequently, $\widetilde{\sigma}$ is the sum of all those components. Note that this leads to a different formula compared to Pham, which leads to different results between our model and the original one from Pham (see Section 4.5) [7].

4.3.4 Derivation of p and τ

The values for p and τ can only be derived numerically. For this purpose, we need two independent formulas for τ. We can formulate the first one by following the Markov chain by using the state $b(0,0)$, defined in Equation 4.11:

$$\tau = \sum_{i=0}^{m_1} b(i,0) = \frac{b(0,0)\cdot(1-p^{m+2})}{1-p} \tag{4.31}$$

The second one can be derived directly from the definition of p, which is equal to all possibilities of at least two stations transmitting and the probability that the interfering source is active:

$$p = 1-(1-\tau)^{n-1}[(1-p_{if})^k + (1-(1-p_{if})^k)\cdot\omega] \tag{4.32}$$

which we can rewrite as:

$$\tau = 1 - \sqrt[n-1]{\frac{p-1}{\omega(1-p_{if})^k - (1-p_{if})^k - \omega)}} \tag{4.33}$$

Using Equation 4.31 and Equation 4.33, we can derive $p$and τ numerically.

4.3.5 Throughput

The throughput of a station can be derived directly from a successful transmission by a station. Three different probabilities compose a successful transmission. First, the probability that only one station transmits, $\tau(1-\tau)^{n-1}$. Second, the probability that the interfering does not become active during the transmission or that it becomes active but does not lead to packet loss due to the presence of the FEC mechanism, being $(1-p_{if})^{k+1} + \omega\cdot(1-(1-p_{if})^{k+1})$. Third, the probability that the station has a packet at the head of the queue $(1-q)$. As the throughput is defined as the fraction of the channel slots in use, we can write the channel throughput U as:

$$U = \frac{P_B\cdot\tau(1-\tau)^{n-1}(1-q)((1-p_{if})^{k+1} + \omega\cdot(1-(1-p_{if})^{k+1}))}{\widetilde{\sigma}} \tag{4.34}$$

4.4 The M/HEXP/1/Q queuing model

In what follows, we model a station as a queue with Poisson input, hyperexponential service time, and finite buffer capacity [177].

Packets arrive according to a Poisson process with rate λ. Upon arrival, they join a finite queue with capacity Q. Packets arriving at a full queue are lost. Accepted packets are transmitted in a First-In, First-Out (FIFO) order.

We distinguish between $m+2$ types of packets:

- Type i packets $0 \leq i \leq m+1$ are transmitted after i unsuccessful transmissions: given the definition of the parameter p, a packet belongs to type i, $0 \leq i \leq m-1$, with probability $a_i = p^i \cdot (1-p)$. Its mean service time, i.e., the time between the packet is at the head of the queue and starts competing for accessing the medium and the time instant the transmission starts, is given by

$$E[A]_i = T_s + i \cdot T_{col} + \widetilde{\sigma} \cdot \sum_{j=0}^{i} \frac{W_i - 1}{2} \tag{4.35}$$

  with T_{col} given by

$$T_{col} = (1-p_{if})^l \cdot T_c + \sum_{b=1}^{l} (1-p_{if})^{b-1} \cdot p_{if} \cdot (\sigma \cdot (b-1) + T_{if}) \tag{4.36}$$

- Type $m+2$ packets are lost due to the contention process (collision with packets of other stations or with the interfering source). A packet belongs to type $m+2$ with probability $a_{m+2} = p^{m+2}$. The mean access delay of this type of packet is given by

$$E[A]_{m+2} = (m+2) \cdot T_{col} + \widetilde{\sigma} \cdot \sum_{j=0}^{m+1} \frac{W_j - 1}{2} \tag{4.37}$$

  Remark that type $m+2$ packets are not successfully transmitted but contribute to the latency of successfully transmitted packets.

These assumptions turn a station into a finite capacity queue with Poisson input and hyperexponential service time, denoted by $M/HEXP/1/Q/$ (see Equation 4.29).

This queue is a continuous-time QBD process. The rate matrix G is determined as follows:

Define the following matrices:

B_0 is an $1 \times (m+3)$ matrix with entries

$$(B_0)_{1,i} = a_{i-1} \cdot \lambda \quad , i = 1, \ldots, m+3 \tag{4.38}$$

B_1 is a 1×1 matrix with entries

$$(B_1)_{1,1} = -\lambda \tag{4.39}$$

B_2 is an $(m+3) \times 1$ matrix with entries

$$(B_2 1)_{i,1} = \mu_{i-1} \quad , i = 1, \ldots, m+3 \tag{4.40}$$

with

$$\mu_i = \frac{1}{E[A]_i} \tag{4.41}$$

A_0 is an $(m+3) \times (m+3)$ diagonal matrix with diagonal entries

$$(A_0)_{i,i} = \lambda \quad , i = 1, \ldots, m+3 \tag{4.42}$$

A_1 is an $(m+3) \times (m+3)$ diagonal matrix with diagonal entries

$$(A_1)_{i,i} = \lambda + \mu_{i-1} \quad , i = 1, \ldots, m+3 \tag{4.43}$$

A_2 is an $(m+3) \times (m+3)$ matrix with entries

$$(A_2)_{i,j} = a_{j-1} \cdot \mu_{i-1} \quad , i, j = 1, \ldots, m+3 \tag{4.44}$$

The rate matrix G, with dimensions $Q_l \cdot (m+3) + 1 \times Q_l \cdot (m+3) + 1$, is given by

$$\begin{pmatrix} B_1 & B_0 & & & & & \\ B_2 & A_1 & A_0 & & & & \\ & A_2 & A_1 & A_0 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \ddots & \ddots & \ddots \\ & & & & & A_2 & A_1 & A_0 \\ & & & & & & A_2 & A_1 + A_0 \end{pmatrix} \tag{4.45}$$

The stationary distribution of the number of packets in a station is given by

$$\bar{v} = (\bar{v_0}, \ldots, \bar{v_{Q_l}}) \tag{4.46}$$

where we can calculate the row vector $\bar{v}$ by solving the linear system

$$\bar{v}G = \bar{o} \tag{4.47}$$

with normalization condition

$$\bar{v}\bar{e} = 1 \tag{4.48}$$

where $\bar{e}$ is a column vector with all entries equal to 1.

The probability that there are i packets in a station, $i \leq i \leq Q_l$, is given by

$$\begin{aligned} \pi_0 &= \bar{v_0} \\ \pi_i &= \bar{v_i}\bar{e_i} \end{aligned} \tag{4.49}$$

where $\bar{e_i}$ is a column vector of dimension $m+3$ with all entries equal to 1. Remark that $\bar{v_0}$ is a scalar. De Nitto Personè and Grassi discuss solutions of finite QBD processes [176]. Remark that $q = \pi_0$ (i.e., the probability that the queue is empty).

The mean number of packets in a station is given by

$$E[S] = \sum_{i=0}^{Q_l} i \cdot \pi_i \tag{4.50}$$

moreover, the mean number of packets in the queue of a station is given by

$$E[QL] = \sum_{i=1}^{Q_l} (i-1) \cdot \pi_i = E[S] - (1 - \pi_0) \tag{4.51}$$

The mean time a packet (successfully transmitted or not) spends in the queue is, using Little's formula, given by

$$\frac{E[QL]}{\lambda(1 - \pi_Q)} \tag{4.52}$$

Remark that in this queue, both packets of type i, $0 \leq i \leq m+1$, and packets of type $m+2$ are present. However, to derive the total mean access latency a packet experiences (i.e., both queuing and service time), only successfully transmitted packets need to be taken into account. Therefore, the mean packet access latency is given by

$$E[A] = \frac{E[L]}{\lambda \cdot (1 - \pi_Q)} + \sum_{i=0}^{m+1} E[A]_i \cdot \frac{p^i \cdot (1-p)}{1 - p^{m+2}} \tag{4.53}$$

4.5 Evaluation

In this section, we present the validation of the model. First, we explain the experimental setup. Then we show the validation for latency, after which we present the validation for throughput. Afterward, we provide an asymptotic analysis for a low and full load. Last, we explore the behavior of IEEE 802.11, depending on different parameters.

4.5.1 Experimental setup

For our results, we have three different scenarios, simulation, measurements in a real testbed, and the model.

For our real measurements, we use the same setup as in Section 3.3.2, with up to 25 stations and one AP in the 5 GHz band. The AP is set to IEEE 802.11a, enforcing the use of the OFDM PHY, and IEEE 802.11n capabilities are disabled. The packets have a size of 1530 bytes, and we fix the bit rate to 54 Mbps. The number of packets per second starts at 25 and increments by steps of 25 packets per second up to 400 packets per second. We repeated each configuration five times.

Additionally, we installed an SDR to generate interference according to the model of an interfering source defined in Section 3.3.2. The same SDR as in Section 3.3.2 is in use, and it is not time-aligned with the slots of the IEEE 802.11 system and, therefore, wholly decoupled, which is a more realistic scenario but is contrary to one assumption for the model.

Additionally, we used simulation as a second comparison besides real measurements. We implemented the interfering source in *ns3*, version 3.28, and simulated the same scenarios as for the real setup to stay comparable. The interfering source

Table 4.3: Summary of values

Parameter	**Value**
W	32
Q_l	64
m	5
$Bitrate$	54 Mbps
P_B	1530 byte
H	28 byte
δ	10^{-6}s
σ	$9 \cdot 10^{-6}$s
$T_{preamble}$	$16 \cdot 10^{-6}$s
T_{signal}	$4 \cdot 10^{-6}$s
$DIFS$	$34 \cdot 10^{-6}$s
$SIFS$	$16 \cdot 10^{-6}$s
N_{SYM}	$4 \cdot 10^{-6}$s
$N_{DBP_{DATA}}$	216
$N_{DBP_{ACK}}$	96

Table 4.4: Airtime occupation of interfering source for different probabilities and timeslots used in the presented experiments.

$\mathbf{T_{if}}$ / $\mathbf{p_{if}}$	**10**	**50**	**100**
0.01	9.1 %	33.33 %	50 %
0.025	20 %	55.55 %	71.43 %

works similar to the already existing waveform generator. Instead of periodic transmission, it works as an on/off source. Each time slot, it turns on with given probability and remains active for the specified amount of time slots. As in the real setup, the interfering source in simulation is not time-aligned to the IEEE 802.11 system. We set the ED threshold to -74 dBm, while we use the Friis propagation loss model and the constant speed propagation delay mode. We set the data rate to 54 Mbps and the control rate to 6 Mbps, both using OFDM and both at a constant rate.

We chose the parameters for the interfering source according to a *low occurrence* with $p_{if} = 0.01$ and a *high occurrence* with $p_{if} = 0.025$. We can distinguish the duration between a *low duration* with $T_{if} = 10$, a *medium duration* with $T_{if} = 50$, and a *high duration* with $T_{if} = 100$. We do not present higher interference probabilities, because the measurements are unreliable for very high interference cases as not enough packets are correctly delivered. Table 4.3 shows the rest of the values.

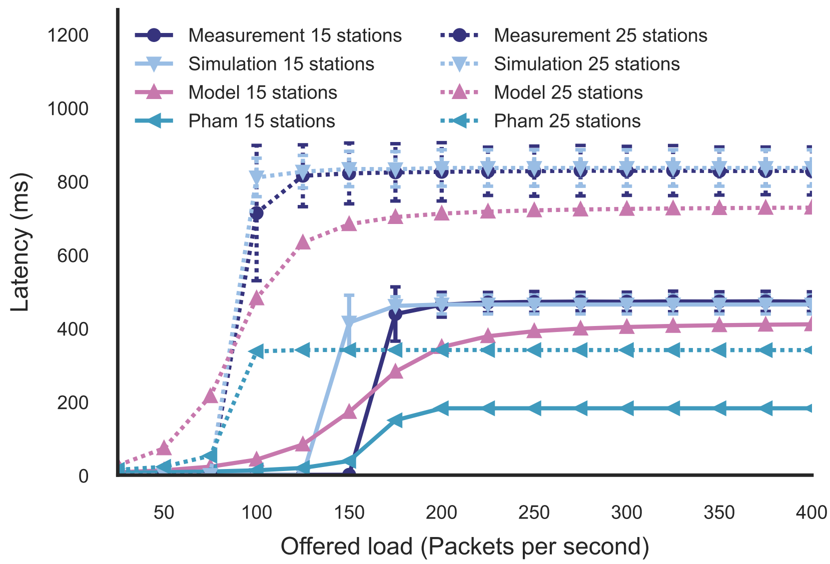
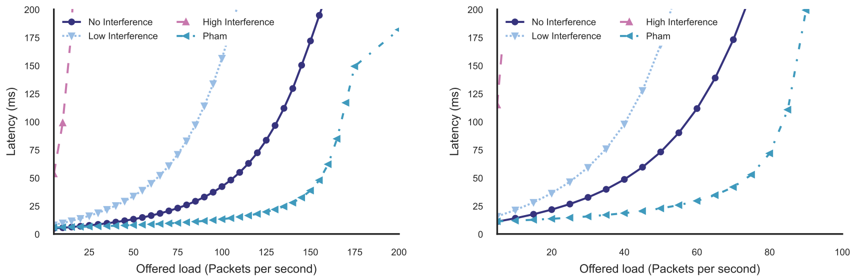


Figure 4.2: Latency comparison with no interference between simulation, measurements, our new model, and Pham's model for 15 and 25 stations [7].

4.5.2 Latency measurements

For the analysis of the latency, there are two significant points to consider. First, the point when the system reaches saturation. The system reaches this point when the latency suddenly goes from a few milliseconds to the maximum possible, which can be up to several seconds. Second, the maximum latency. The system achieves this maximum as soon as it reaches saturation. Both are essential indications for QoS and determine the quality of the link for a user.

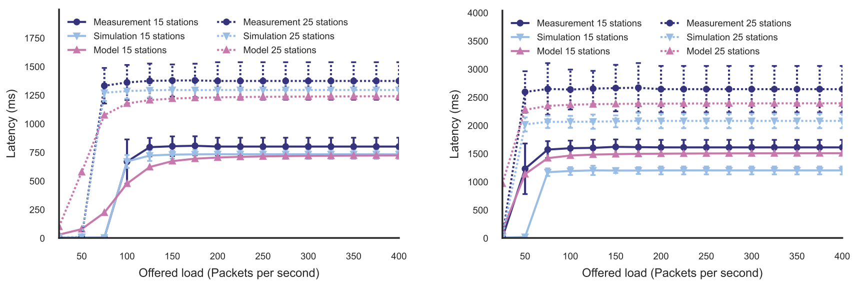
First, we compare the performance of our model with no interference against Pham's model, as well as simulation and measurements in Figure 4.2 [7]. While predicting the saturation point correctly, we can see that Pham's model significantly underestimates latency in both simulation and measurements, by 250 %. Our model predicts the saturation point correctly, but still slightly underestimates simulation and measurement by 11 %, but in general, performs much better. The difference between our model and Pham's model lies in a different formulation for the average slot time $\widetilde{\sigma}$. We corrected a mistake in Pham's formula. The slightly higher value for the simulation in the case of 25 stations compared to the measurements likely stems from two reasons. First, the fact that in a real setup, not every node hears all transmissions from all other nodes. Second, the exact behavior of IEEE 802.11 hardware and firmware are partially unknown as it is vendor dependent. This black box behavior is especially the case for the ED threshold, which influences the detection of a busy medium. The fact that this happens with 25, but not 15 stations,



(a) 15 stations.

(b) 25 stations.

Figure 4.3: Latency comparison of the models before saturation for variable interference and a fixed number of stations.



(a) Low interference occurrence and medium duration.

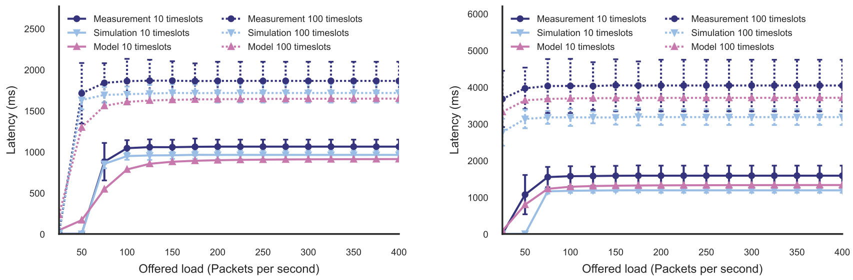
(b) High interference occurrence and medium duration.

Figure 4.4: Latency for fixed interference occurrence, fixed medium duration, and a changing number of stations.

makes it very likely that not all stations consider the channel busy at the same time.

Figure 4.3 gives an overview of the behavior captured by the models in the unsaturated phase in greater detail for 15 and 25 stations. We can see that the model of Pham has a more prolonged phase of lower latency, while our model increases latency early on [7]. Interference amplifies this effect, and with high interference, the latency is increased at around ten packets per second already.

Figure 4.4a and Figure 4.4b are the first figures that show interference with the first a low occurrence ($p_{if} = 0.01$), the second a high occurrence ($p_{if} = 0.025$), and both with a medium duration ($T_{if} = 50$). Our model predicts the saturation point correctly independent of the number of stations while it underestimates the maximum latency slightly. In the case of low occurrence, the difference in latency to the measurements is below 9 %, while it is below 3 % for the simulation. When a high occurrence of interference is present, the model keeps within the 9 % range but is further away from the simulation, which is acceptable as the simulation severely



(a) Low interference occurrence and 25 stations.

(b) High interference occurrence and 25 stations.

Figure 4.5: Latency for fixed interference occurrence, a fixed number of stations, and changing duration.

underestimates the interference.

In Figure 4.5a and Figure 4.5b, the number of stations is fixed to 25 while we have a low occurrence of interference (p_{if} = 0.01) in the first figure and high occurrence (p_{if} = 0.025) in the last. In all cases, our model predicts the saturation point accurately. In the case of low occurrence, our model is slightly underestimating both simulation and measurement by 3 % and 11-14 %, respectively, with a better result with a higher duration (T_{if} = 100) when compared to the measurements. In the case of high occurrence, our model is closer to the measurement compared to the simulation, independent of duration. We can achieve a difference compared to the measurement of 8.5-17 %, with higher accuracy when the duration is longer. The simulation deviates from the measurements by 27-33 %, which shows that the simulation can not fully capture the effect of interference on an IEEE 802.11 network.

Figure 4.6 shows a fixed duration, medium (T_{if} = 50), with a fixed number of stations (25), and changing occurrences. Our model predicts the saturation point correctly in all scenarios. The latency for low occurrence is jointly estimated by both model and simulation, with 9 % and 6 %, respectively. High occurrence increases this difference, and our model can capture the effects of a real deployment much better than the simulation with a difference of 9 % for the model and 27 % for simulation.

Our model can accurately predict the latency of a real setup while the simulation is too simplified to cope well with high interference. The results show that a setup where QoS is essential can use our model, compared to simulation.

4.5.3 Throughput measurements

Let us now consider the throughput. Again, two significant points are essential. First, the slope at the beginning where the requested bandwidth is still available until the system achieves the maximum throughput. This point is similar to the

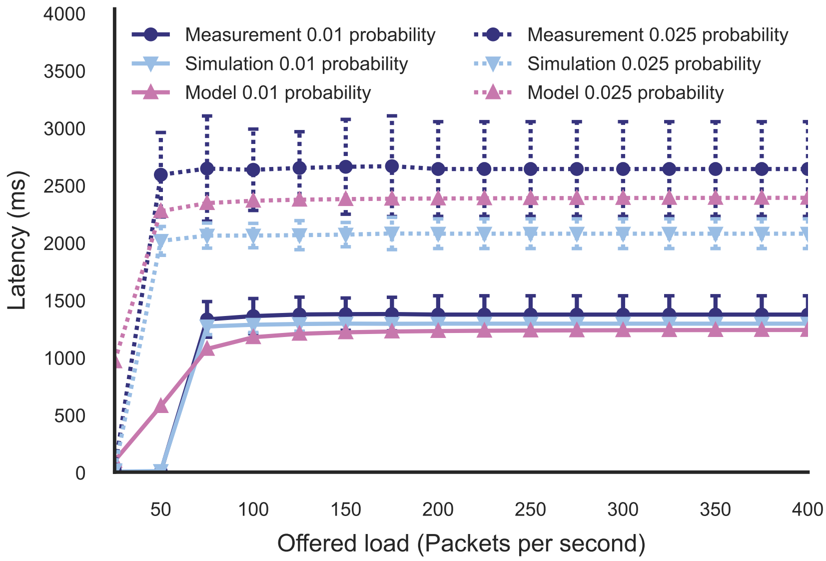


Figure 4.6: Latency for medium duration, 25 stations, and changing occurrence.

saturation point. Second, the maximum throughput itself that a station can achieve in each scenario.

Figure 4.7 compares our model against Pham's model and Challa et al.'s model, as well as against simulation and measurement for 15 and 25 stations [7, 8]. While our model estimates the slope correctly, both, Pham's model and Challa et al.'s model overestimate the slope and possible throughput for a station significantly. In the first case, by 10-11 % and the second case by 14-19 % [7, 8]. Simulation underestimates available throughput by 9-12 %, while our model slightly overestimates throughput by 2-4 %. Our model outperforms all other models, including simulation.

The difference between our model and Pham's model is the same as in Figure 4.2.

Figure 4.8a shows the first throughput results with interference with a low occurrence, and Figure 4.8b shows it with a high occurrence and medium duration. Simulation and model both follow the slope well while the model underestimates the throughput slightly by 9 % and the simulation by 6-9 % in the case of low occurrence. When the interference has a high occurrence, the model underestimates slightly by 6-9 %, and the simulation overestimates the performance by 27-34 %. Again, the model represents the effects of reality correctly when the interference has a high occurrence.

Figure 4.9a and Figure 4.9b show the effect of a changing duration in combination with different occurrences. If there is low interference occurrence, then

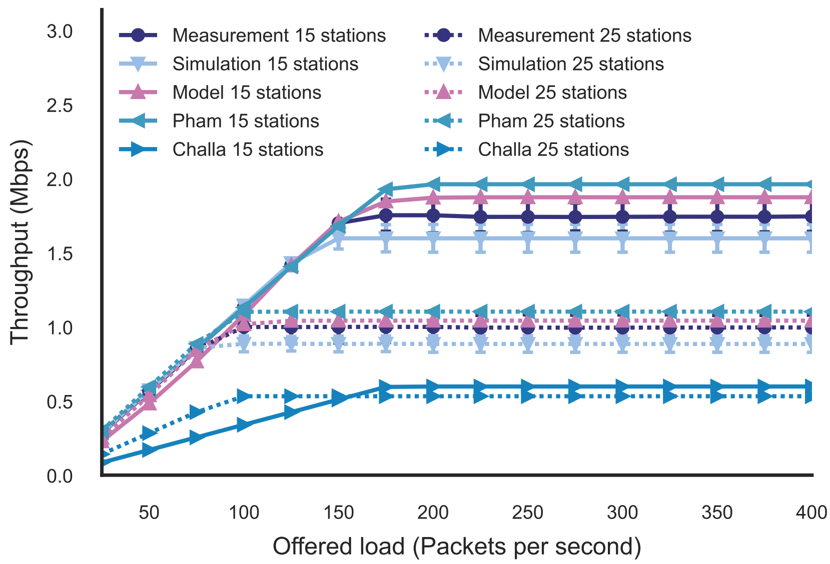


Figure 4.7: Throughput comparison with no interference between simulation, measurements, our new model, Pham's model, and Challa et al.'s for 15 and 25 stations [7, 8].

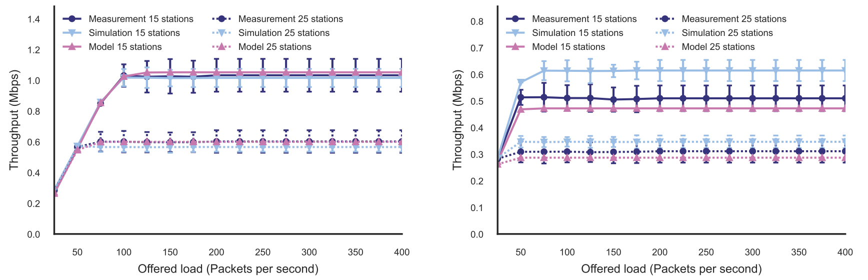
both simulation and model are reasonably accurate, with 11-14 % and 9-10 %, respectively. The duration itself has not much effect. In the case of high occurrence, the difference is more significant, with 27-33 % for simulation and 8-17 % for our model with better accuracy when the duration of the interference source is longer.

In Figure 4.10, we fixed the number of stations to 25 as well as the duration to medium ($T_{if} = 50$). The effect of the probability can be seen, especially in the simulation where the difference to the measurements ranges from 6 % to 27 %. In contrast, the model maintains a difference of 9 % and is much more reliable.

Our model performs well in predicting the throughput and, contrary to simulation, can give an accurate value for each scenario that we explored. Our model also performs better than state-of-the-art models like the one from Pham or Challa et al. that does not include external interference [7, 8].

4.5.4 Asymptotic Analysis

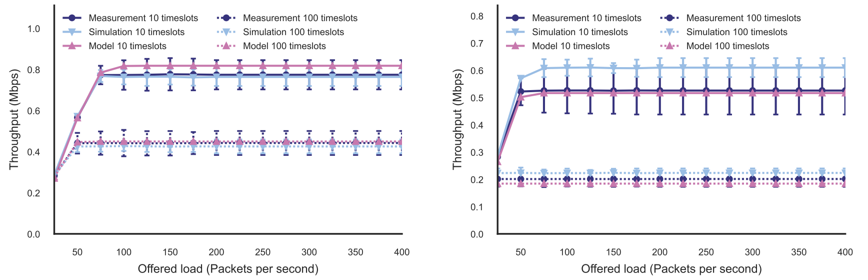
In this section, we evaluate the average packet delay in case of low load (i.e., λ small) and in case of high load (i.e., for high values of λ). In case of low load, we assume that the packet queue is empty upon arrival of a packet. Hence, the average delay, in that case, is given by the average service time a packet experiences, given



(a) Low interference occurrence and medium duration.

(b) High interference occurrence and medium duration.

Figure 4.8: Throughput for fixed interference occurrence, fixed medium duration, and a changing number of stations.



(a) Low interference occurrence and 25 stations.

(b) High interference occurrence and 25 stations.

Figure 4.9: Throughput for fixed interference occurrence, a fixed number of stations, and changing duration.

by

$$E[V] = \sum_{i=0}^{m+1} E[A]_i \cdot \frac{p^i \cdot (1-p)}{1-p^{m+2}} \tag{4.54}$$

In case of a high load, we assume that a tagged packet arrives at a queue with $Q_l - 1$ packets. Those $Q_l - 1$ packets are served before the tagged packet. The service of a packet may be a successful transmission or the dropping of a packet. Hence, the service time of each of these $Q_l - 1$ packets is given by

$$E[S] = \sum_{i=0}^{m+1} E[A]_i \cdot p^i \cdot (1-p) + E[A]_{m+2} \cdot (1-p^{m+2}) \tag{4.55}$$

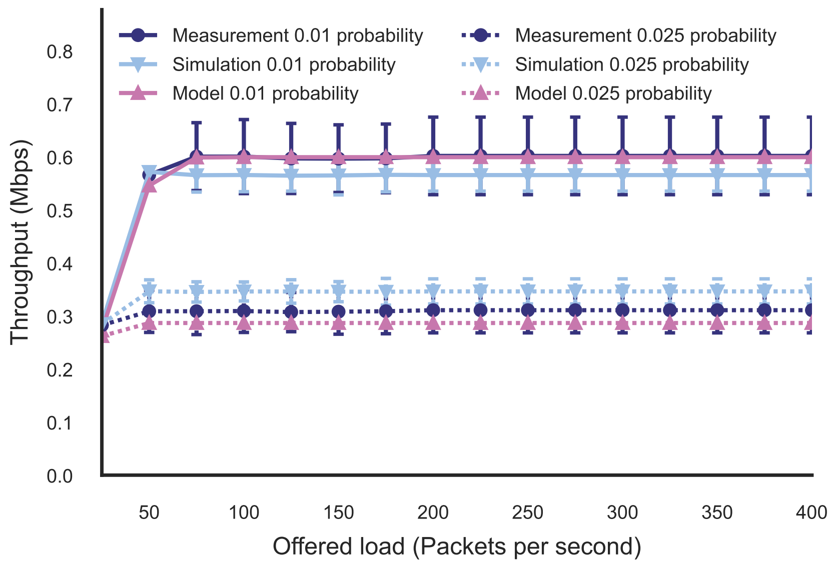


Figure 4.10: Throughput for medium duration, 25 stations, and changing occurrence.

The average delay of the tagged packet is then given by the sum of the service times $E[S]$ of the $Q_l - 1$ packets and the service time of the tagged packet

$$E[V] = (Q_l - 1) \cdot E[S] + \sum_{i=0}^{m+1} E[A]_i \cdot \frac{p^i \cdot (1-p)}{1 - p^{m+2}} \tag{4.56}$$

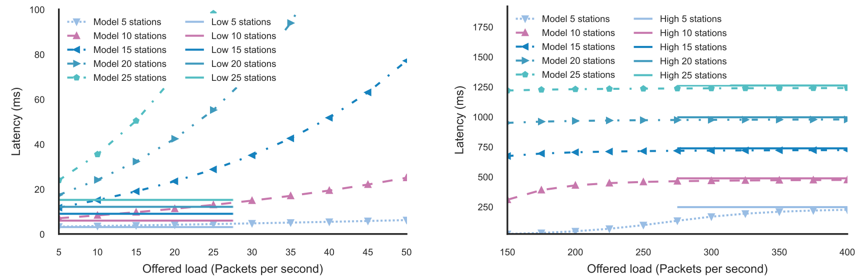
In what follows, we apply this asymptotic analysis to the following example. Let $T_{if} = 50$, $p_{if} = 0.01$ and let the number of stations vary $n = 5, 10, 15, 20, 25$. We compute for each n the values of p and τ under low load, i.e., $\lambda = 5$, and under high load, i.e., $\lambda = 400$. These values of p and τ allow computing the average delay making use of the above formulas. In Figure 4.11, for each value of n, the average delay for variable λ is shown using the detailed model, together with the low and high load asymptotic. We can see that the asymptotes fit in the low load scenario, as well as the high load scenario.

4.5.5 Parameter exploration

In this section, we will explore the influence of the parameters p_{if}, T_{if}, and ω, as well as the impact of the RTS/CTS mechanism on the latency and throughput.

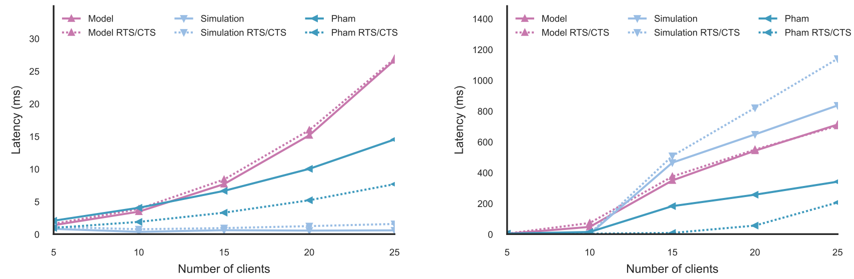
4.5.5.1 RTS/CTS mechanism

The RTS/CTS mechanism has the goal of reducing collisions, especially for hidden terminals. Figure 4.12a and Figure 4.12b show a low and high load scenario,



(a) Low asymptote. (b) High asymptote.

Figure 4.11: Latency for fixed interference and a different number of stations with asymptotes for high and low load.



(a) Low load. (b) High load.

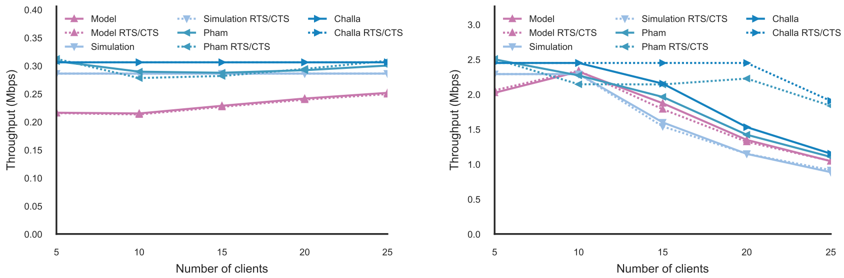
Figure 4.12: Latency comparison of RTS/CTS mechanism to basic mechanism with simulation, based on clients.

respectively, for latency based on an increasing client number. Model and simulation show a higher latency with RTS/CTS enabled. The throughput using the basic mechanism and using the RTS/CTS scheme are very close to each other, as shown in Figure 4.13a and Figure 4.13b. While the throughput remains close to the basic mechanism with an increasing number of stations, the latency further increases in that case. The improvement that the RTS/CTS mechanism shows in low bandwidth scenarios with DSSS and the reduction of T_c seems negated by the use of higher bandwidth and OFDM as the encoding scheme.

From a performance viewpoint, the RTS/CTS mechanism does not have a substantial impact on higher bandwidths. The difference to the basic mechanism is minimal, and therefore it can not always be recommended.

4.5.5.2 Interference occurrence

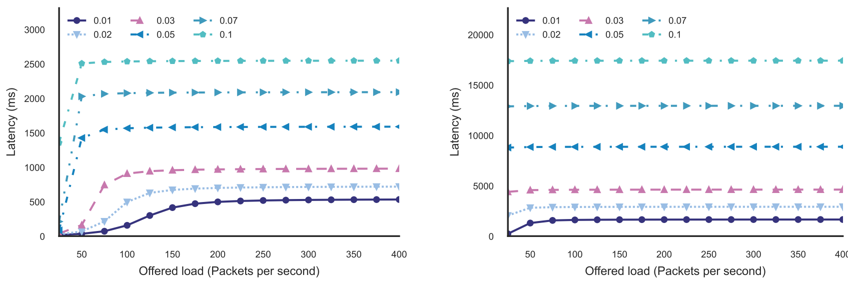
Figure 4.14 and Figure 4.15 show the latency and throughput for a variety of values for p_{if} with a fixed number of stations and duration. In both cases, a low number



(a) Low load.

(b) High load.

Figure 4.13: Throughput comparison of RTS/CTS mechanism to basic mechanism with simulation, based on clients.



(a) 15 stations and low duration.

(b) 25 stations and high duration.

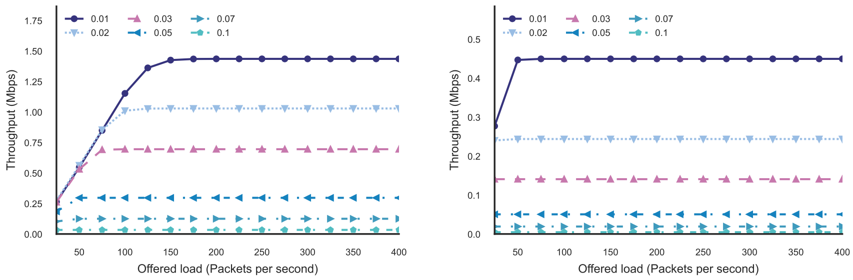
Figure 4.14: Latency comparison of the model with a varying occurrence and a fixed number of stations and duration.

of stations and low duration, as well as a high number of stations and high duration, an increase of p_{if} is combined with a significant increase in latency. In the first case, the increase is roughly five times the lowest value of p_{if} to the highest. In the second case, that increase is around ten times. Both the increase in duration and number of stations add to the problem that a high interference occurrence is causing. Similar to latency, the throughput also drops significantly, but much more than latency. In the case of a low number of stations and low duration, it loses 97.5 % of throughput from the lowest value of p_{if} to the highest. With a high number of stations and high duration, it loses even up to 99 % of its throughput.

We can conclude that the increase in interference occurrence has a much higher impact on throughput than latency.

4.5.5.3 Interference duration

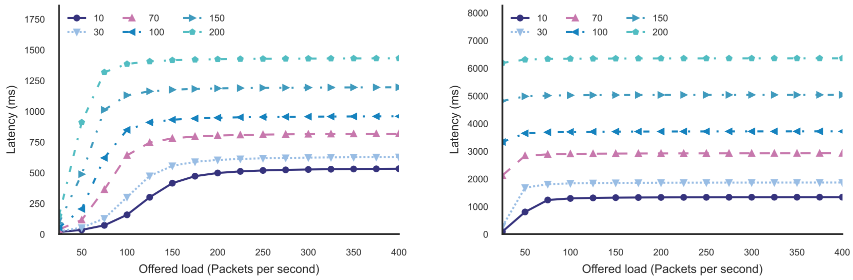
Similar to the interference occurrence, we also explore the duration in Figure 4.16 and Figure 4.17. With increased duration, latency increases by a bit less than three



(a) 15 stations and low duration.

(b) 25 stations and high duration.

Figure 4.15: Throughput comparison of the model with a varying occurrence and a fixed number of stations and duration.



(a) 15 stations and low occurrence.

(b) 25 stations and high occurrence.

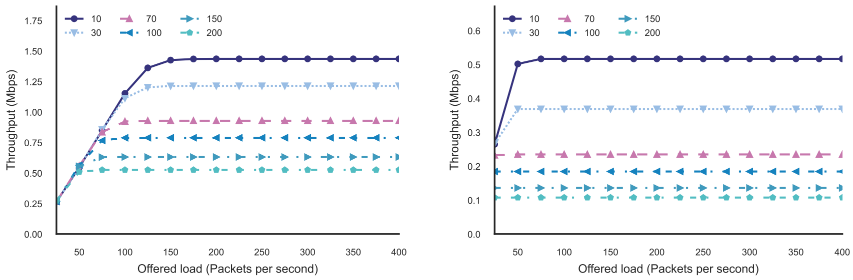
Figure 4.16: Latency comparison of the model with a varying duration and a fixed number of stations and occurrence.

times for a low number of stations and low occurrence and up to six times for a high number of stations and a high occurrence. It is interesting to note, though, that an up to 20 times increase of duration has much less impact than a ten times increase of occurrence. The decrease in throughput scales like the increase in latency, and we have up to three times and six times reduction in throughput from the lowest value for T_{if} up to the highest.

We can conclude that while duration has an impact, it has a significantly smaller impact than occurrence. This difference in impact has to do with the ED function of IEEE 802.11, where a higher occurrence has a higher probability of triggering an additional back-off phase.

4.5.5.4 FEC mechanism

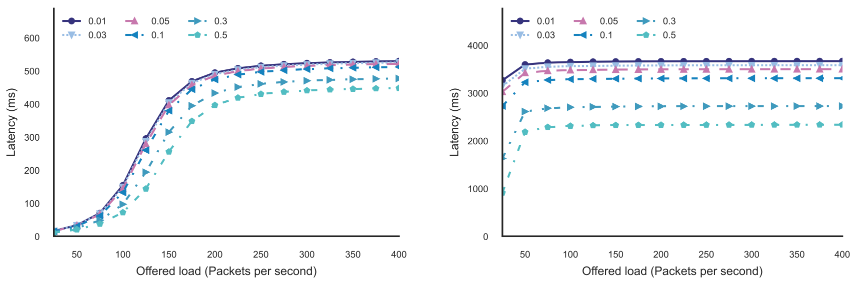
In Figure 4.18 and Figure 4.19, we can see the effect of ω, the probability that the FEC mechanism can recover the packet. However, it had a collision with the interfering source, on the latency and throughput. The latency reduction reaches



(a) 15 stations and low occurrence.

(b) 25 stations and high occurrence.

Figure 4.17: Throughput comparison of the model with a varying duration and a fixed number of stations and occurrence.



(a) 15 stations, low occurrence, and low duration.

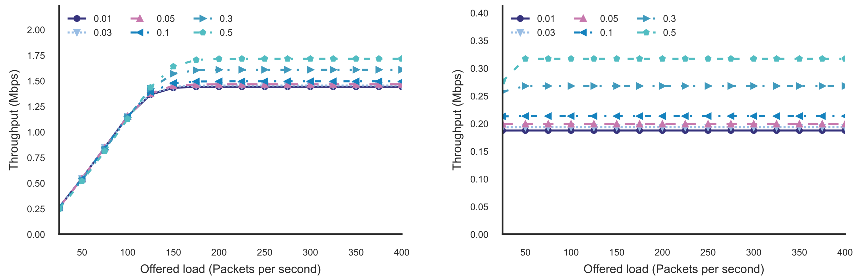
(b) 25 stations, high occurrence, and high duration.

Figure 4.18: Latency comparison of the model with a varying ω and fixed duration, number of stations, and occurrence.

from 0.4 % for $\omega = 0.01$ to 16 % for $\omega = 0.5$ in the case of a low number of stations, low occurrence, and low duration. With a high number of stations, high occurrence, and high duration, the reduction reaches from 1.3 % for $\omega = 0.01$ to 37 % for $\omega = 0.5$. In both cases, the reduction scales slightly lower than linear with the value of ω.

If we look at the throughput in Figure 4.19, we can see that an even more significant improvement is achieved. In the first case, the increase reaches from 0.7 % to 19.5 %, and in the second case, from 1.6 % to 72 %. While in the first case, it does scale lower than linear, in the second case, it scales significantly better than linear.

We can conclude that the FEC mechanism can have a significant impact on latency and throughput with high interference. For high interfering cases, an improvement of the FEC mechanism is worthwhile.



(a) 15 stations, low occurrence, and low duration.

(b) 25 stations, high occurrence, and high duration.

Figure 4.19: Latency comparison of the model with a varying ω and fixed duration, number of stations, and occurrence.

4.6 Conclusion

In this chapter, we provided an analytical model to describe the performance and behavior of an IEEE 802.11 system with a non-IEEE 802.11 interfering source. Based on a Markov Chain and a versatile description of the interfering source, it can describe the behavior of such systems accurately. In Section 4.5.5, we showed how the model could be used to explore further the behavior of IEEE 802.11 systems with a wide variety of different parameters, including the probability that the FEC mechanism can correct the packet. Compared to the model described in Section 3.4.1, the presented model is much more advanced at the cost of higher computation times. For real-time network planning, one can use a combination of pre-computed values of the advanced model with the on-demand calculation of the fast model.

5

ORCHESTRA: Managing heterogeneous wireless networks

The contributions presented in this chapter are based on the publications titled *"ORCHESTRA: Enabling Inter-Technology Network Management in Heterogeneous Wireless Networks"* and *"ORCHESTRA: Supercharging wireless backhaul networks through multi-technology management"* and the patent *"Network stack for a plurality of physical communication interfaces"*.

5.1 Introduction

We introduced the problem of heterogeneous wireless networks in Chapter 1 and highlighted the challenges that a possible solution needs to face. The main one among them is that technologies are managed independently from each other, and managing multiple technologies leads to inefficient use of resources and, therefore, higher costs or a decreased user experience. It is also not possible to leverage the individual strength of each technology. Applications just use whatever technology is available without verifying that it provides the properties the application needs.

Academia and industry alike defined multiple solutions to circumvent the problem of heterogeneous wireless networks. To circumvent the problem, multiple solutions have been defined. Most prominent among them are the IEEE 1905.1 standard, LTE-LWA, SDN-based solutions, and MPTCP. Chapter 2 provides a detailed overview of each solution. Each of them has its shortcomings though that limits in which situations it can be applied. While MPTCP is technology independent and provides packet-level control, it does not have any intelligence, only supports a singular transport protocol, and multiple flows can even lead to decreased performance [124]. The other solutions only provide flow-based control and are generally limited to specific technologies. LTE-LWA, offered by the 3GPP group, combines IEEE 802.11 with LTE but does not include any other technology [4, 118, 119]. Similar, IEEE 1905.1, which introduces an abstract later to manage multiple technologies, only works with IEEE 802.3 (Ethernet), IEEE 802.11, powerline, and MoCA. The same is the case for SDN-based solutions, which mostly focus on IEEE 802.11 networks, which, contrary to many technologies, have no management by default.

To solve the management challenge in heterogeneous wireless networks, we present ORCHESTRA. This management framework works with arbitrary technologies and is legacy compliant. It consists of two parts; the VMAC layer and the controller. The VMAC serves as an abstraction layer for technologies, while the controller serves as a central point to collect information and to implement intelligence.

The rest of this chapter is structured as follows. First, we present the architecture with the two components. Afterward, we explain the functionality and features of the framework. Then we show the implementation of a prototype using IEEE 802.11 and LTE, which we also evaluate.

5.2 Architecture

The ORCHESTRA framework aims to abstract any available and future technology so that management can be combined and centralized. For an application or end-user, it should appear as a single connection and therefore enabling the *Connectivity as a Service* paradigm. The two components of the architecture, the VMAC layer and the controller, aim to achieve this concept. Especially the VMAC is helping in abstracting arbitrary technologies. In the next two sections, we will first discuss the virtual layer in detail and afterward explain the controller.

5.2.1 The virtual layer

It is essential that connections and flows are not interrupted, but can switch from one technology to another in an instant to achieve *Connectivity as a Service*. This behavior is more accurately known as handovers or roaming. This mechanism is currently not used as every technology is managed by itself and implements its network stack according to the OSI model. This design means that every technology has an individual IP address, and switching between technologies would also mean switching between IP addresses. Such a switch, in general, leads to an interruption of an ongoing connection. For this purpose, we designed the VMAC layer as an abstract layer between the MAC layer and the network layer. An abstraction layer allows for transparency and legacy support as it does not change any existing technology or layer. It also enables additional functionality, such as the previously mentioned handover for enabled devices. The abstraction layer and its placement

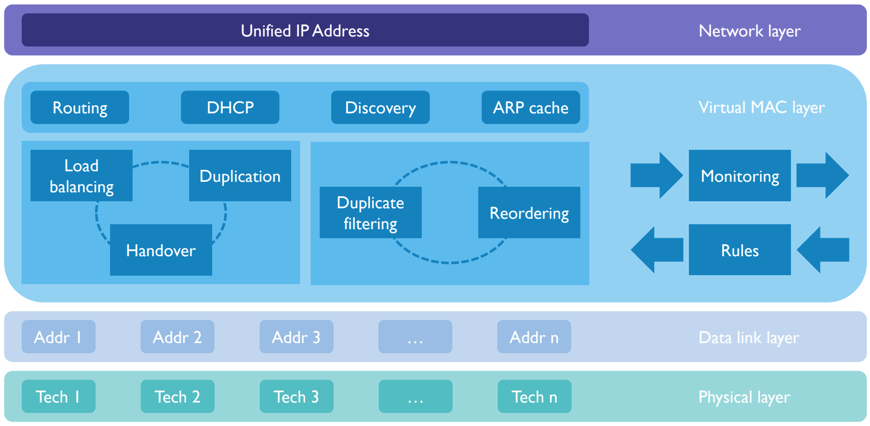


Figure 5.1: The virtual layer of the framework between the MAC layer and the network layer, abstracting each technology to higher layers.

between the layers, as well as its components and functionality, can be seen in Figure 5.1.

The abstraction layer identifies itself to the network layer as a single interface, thus, only requiring a single IP address. On the other side, it handles all available MAC interfaces. The virtual layer handles all the necessary lower level routing, such as based on MAC addresses through Address Resolution Protocol (ARP). Other standards, such as IEEE 1905.1, require a unique MAC address to identify devices, which is not necessary with our solution. The VMAC stays in compliance with the standards of each technology. To devices without a virtual layer, it appears as they are communicating with only the current technology. The technologies and their links can connect to different networks, for example, wireless edge networks and a core network. The virtual layer then needs to take care of the routing between networks and incorporates this functionality.

Those attributes of the virtual layer, the fact that it can handle all interfaces and that it requires only a single IP address enable several advanced functionalities:

1. Seamless handovers, either within a technology between wireless endpoints, or between technologies for mobility and QoS support.

2. Packet-based load balancing across technologies for increased throughput of a device combined with reordering to avoid adverse reactions of transport protocols.

3. Packet-based duplication across technologies to improve the reliability of a flow combined with deduplication to filter out packets that arrived more than once.

The use of packet matching rules realizes all of these advanced functionalities. Those rules are applied to incoming and outgoing packets alike to enable the desired feature. The controller uses rules and commands to apply features and used technologies. In the rare case, when no connection to the controller exists, the VMAC contains simplified logic that ensures that connectivity remains established. The decision which rule or command to use is made based on collected statistics and monitoring information that each device with a virtual layer forwards to the controller. Section 5.2.2.1 provides a more detailed explanation. The VMAC then applies the rules and commands to route the packets to the right interface and use the correct functionality, for example handing over to another interface. The actual transmission and medium access are handled by the interface and technology, which ensures that no changes to it are necessary. The granularity of the rules allows packet-based instead of flow-based control, which is more versatile and allows a centralized controller to be more flexible and dynamic.

The packet flow on a node with a VMAC is very similar to a legacy node. When an application tries to send a packet, it traverses the network stack until it reaches the network layer. In contrast to a legacy node, it does not enter the MAC of the interface directly, but it enters the virtual layer first. The rules and commands in place then decide based on the header of the packet to which interface or interfaces it forwards it and what actions it applies to the packet. Similarly, when an interface receives a packet, it forwards it to the virtual layer instead of the network layer, like on a legacy node. The VMAC then applies deduplication or reordering if necessary and afterward forwards the packet to the network layer. The exact steps and mechanisms applied are further discussed in Section 5.2.1.2.

The design of the VMAC allows its use on any kind of device, ranging from endpoints like consumer or M2M devices to infrastructure devices such as APs and base stations. Similarly, the design allows its use on any kind of network, may it be LAN, Radio Access Network (RAN), backhaul, or core networks. The virtual layer is also capable of connecting different networks, such as, for example, a RAN with a core network. For this purpose, it needs to fulfill bridging functionalities and route traffic from one network to the other. However, in this case, the VMAC needs to distinguish between two types of interfaces, internal and external ones. The internal interfaces are, in general, part of the wireless subnetwork that connects edge nodes to end devices and does not have direct access to an external or public network. In such a case, one or more external interfaces connect the internal interfaces to a core or external network and often allows access to the public Internet. An external interface is not always necessary. If the node is a termination point without further endpoints, no external interface is required, and the virtual layer can be implemented as a lighter version to offer a solution for resource-constrained devices. In the case an external interface is present, the VMAC needs to route traffic between subnetworks, especially it needs to translate IP addresses between the different networks.

5.2.1.1 Building blocks

Unified IP: Each device that communicates with the Internet requires an IP address to identify itself and that a transport protocol can deliver packets to the device. Typically, each interface has its IP address, but that breaks the connection if the device switches interfaces. The VMAC only requires a single IP address, which it requests through one of the available interfaces to deliver seamless handovers. For this purpose, the virtual layer handles the Dynamic Host Configuration Protocol (DHCP) functionality instead of the operating system to avoid address conflicts. To avoid conflict between the technologies, the controller needs to manage all different technologies of a device effectively, and the VMAC needs to inform the controller in detail about the available technologies of a device.

Handling multiple interfaces: In a local network, the ARP usually handles the identification of the next hop. If more than one interface is active, in the case of load balancing or duplication, the operating system and its ARP table are too limited to handle it correctly. In legacy devices, each interface has an IP address, and the operating system can request and respond to requests for each interface and save it in a table. With more than one active interface and the same IP address for each interface, the operating system would continuously overwrite the next-hop entry. To avoid this and make multiple interfaces usable at the same time, the VMAC takes care of the ARP handling. As IP addresses can be reachable through multiple interfaces, it keeps an ARP table for each interface separately and maintains in it each IP address and its next hop. To discover the next hop, the VMAC sends out ARP requests on each interface separately, which also ensures legacy compliance as it appears as only one interface is present. Similarly, the virtual layer replies to requests only on a single interface. For intermediate nodes or nodes without a virtual layer, nothing changes, and they can use any kind of equipment. In the case that a network consists only of devices with a VMAC, this functionality is not necessary anymore.

Monitoring: A paramount aspect of network management is available information. For this purpose, the virtual layer continuously sends configuration and monitoring information to the controller. The controller aggregates the information of all devices and can derive a detailed global view of the network from it. Available interfaces, their properties and capabilities, as well as their state, are part of the configuration information. The statistics about the interfaces and current traffic and other parameters are part of the monitoring information. The messages to the controller are simple UDP packets that require no additional support and work on any network.

Rules: While the VMAC contains simple intelligence to stay connected or establish a connection, the real intelligence is in the controller. In both cases, the virtual layer implements it by using rules that define the actions to be applied to packets. These rules handle specifics for monitoring reports and incoming and outgoing packets. The controller can specify with which frequency new information should be sent, depending on what is currently required to manage the network. Packet rules can be partially compared with OF rules and define advanced functionality such as load balancing, duplication, and handovers. Similar to OF rules, there are two parts, the matching and the action to take. The matching part can take nearly any field of the transport protocol. These fields include the source and destination addresses, port numbers, protocol type, and sequence numbers. The action part ranges from simple forwarding over a single interface to using advanced functionality such as load balancing or duplication, including reordering and deduplication, which uses multiple interfaces.

Similar to monitoring, the controller uses UDP packets to send rules to the virtual layer. The controller can use standard sockets for this, while the VMAC does packet header matching on incoming packets to check for packets from the controller. If the IP address matches, the virtual layer extracts the data from the packet. While the controller is the best choice to decide on rules for a device due to its global view, the VMAC can set them locally as well. The virtual layer contains limited intelligence to maintain connectivity, and in principle, a local application can set rules as well to avoid connection loss. These locally set rules are usually temporary until the controller is reachable again and overwrites them.

Discovery: A device needs to make itself known when it joins the network to receive messages from the controller. For this discovery process, the VMAC broadcasts discovery messages in the form of UDP packets. The best controller, in case of multiple controllers, answers to the device, and from that moment on, the virtual layer can communicate directly with the controller. As the controller IP is not known before the answer to the discovery message, the message itself contains a unique identifier that is put again into the first 64 bytes of the payload of the answer. Until the VMAC receives the identifier, it parses the payload of every UDP packet to learn the IP of the controller.

5.2.1.2 Features

There are three significant features available for the framework, handovers, load balancing, and duplication. In this section, we explain the mechanisms for each of them.

Handovers: A handover is defined as changing the current wireless endpoint that a device is connected to, to another one. An endpoint can be an AP, a base station, a gateway, or any other type. There are two types of handovers, horizontal and vertical. A horizontal handover keeps the technology but changes the endpoint, while a vertical handover includes a technology change as well. In most scenarios, the controller decides the time for a handover and sends the new configuration to the devices that are involved. In the case of a handover, the VMAC currently uses one interface, the active interface. If the handover is a horizontal one, the active interface stays the same, but the endpoint the VMAC is connected to changes. In the case of a vertical handover, the active interface changes, and traffic should flow through the new interface instead of the old one. For this purpose, the controller updates the actions of the rules to use the new interface to transmit traffic.

The VMAC needs to ensure that when a handover happens, no packets are lost so that the change in underlying technology appears seamless. As a first step, the VMAC starts to buffer all outgoing packets before initiating the switch to a new endpoint or the change to a new active interface. If the active interface changes to a new one, the VMAC first sends out a gratuitous ARP to trigger an update of the ARP tables of devices in the network. These devices then associate the MAC address of the new interface with the unified IP address. In the case that the interface is not connected, the VMAC first initiates the procedure to connect the interface, depending on the technology, and afterward switches the interface. In rare cases, the new interface might prevent continuing the connection. The VMAC will then fall back to the previous active interface as a fail-safe. Either way, the VMAC stops buffering and starts sending the buffered packets. The virtual layer is also capable of initiating a handover on its own if the connectivity on a technology drops. The procedure is the same.

The procedure for a horizontal handover is slightly different. It requires synchronization between all devices involved, which usually include a client device, wireless endpoints such as APs, and possibly switches in the wired part of the network. First, the controller informs all devices that a handover is imminent and which devices are involved. Then the devices need to agree on a time when the handover occurs. Usually, the client device sends a message with a synchronization timer to which all devices respond with a time window δ how long it will take to perform the actual handover. All participants agree on the largest δ by sending acknowledgments (ACKs) as well as on a time t when the handover starts. At time t, each device executes the necessary steps for a handover. It then tests the connection as soon as it completed all steps by sending an ACK to all involved devices. If any of the devices can not establish a connection, they switch back to the previous configuration. Similar to a vertical handover, the VMACs on all ends buffer the packets during the time δ and continue transmission when the handover is complete.

As not all devices contain a VMAC, there are other modes to achieve a handover. In the case that am SDN/NFV controller manages the wireless endpoints, it can instruct the endpoints to handover correctly. If no such controller exists and the APs operate independently, a handover is still possible. However, the VMAC can not guarantee an entirely seamless handover as it does not know how long it will take or what impact it has on performance. However, it can still use the buffer to reduce any performance impact if possible.

Load balancing: The goal of load balancing is improving throughput or better utilization of resources and uses two or more interfaces at once. The VMAC supports fine-grained packet-based load balancing compared to the more common flow-based load balancing. It uses a straightforward weighted Round Robin with adjustable weights, which sends a fixed number of packets to an interface and then switches to the next one. There is no overhead involved, as both interfaces are active and connected to their respective endpoints. The VMAC can simply switch between them. As is typical with using multiple links for load balancing, there is no guarantee of in-order arrival of the packets of the flow on the receiving end. The latency on different links might be different due to the used technology, interference, load, or different prioritization. It is, therefore, necessary to reorder packets in the virtual layer before forwarding them further. If no reordering occurs, transport protocols that require in-order arrival, such as TCP, show severely decreased performance. Reordering happens on a flow basis, identified by the transport protocol header, usually the source and destination address and port, combined with the sequence number. For each flow, the VMAC tracks the current required sequence number and buffers packets that arrived too early. When the packet with the correct sequence number arrives, the virtual layer passes it on with all the packets in the buffer reordered until the next sequence number is missing. As packet loss can occur, each packet also has a timeout, after which the virtual layer passes it on, ignoring the missing sequence number. The VMAC calculates the timeout dynamically based on the current traffic rate and keeps it minimal to reduce performance impacts due to the behavior of the transport protocol. After a timeout and the forwarding of the packet, regular operation continues with emptying the buffer until the next missing sequence number.

Duplication: Duplication increases reliability by using multiple interfaces instead of throughput, like load balancing. By sending a packet over multiple links, the probability of packet arrival increases, and less packet loss occurs, increasing the performance. To use duplication, the VMAC copies an outgoing packet and sends it across all interfaces that the rule specifies. Not every duplicated packet is lost and multiple copies of a packet can arrive on the receiver, which can cause reactions of the protocol or in applications that reduce performance. To void this, the VMAC of the receiver filters packets and forwards only one copy of each packet. The virtual layer keeps a hash map of all received packets. The VMAC uses network layer information, such as the source and destination IP, transport layer protocol, IP identifier, and IP fragmentation offset to identify packets. Similar to load balancing, each packet has a timeout that it computes dynamically based on the rate of the flow. The virtual layer removes a packet out of the hash map in two instances. Either, it reaches the timeout, or it reached the maximum number of duplicates, depending on the number of interfaces used.

5.2.2 The centralized controller

The controller is the center of the proposed framework and the place where intelligence and management resides. Together with the VMAC that offers fine-grained control and advanced functionality, it provides the basis for multi-technology management over multiple wireless networks. The controller uses the SDN principle where the control plane is abstracted from network devices such as the virtual layer and transferred to a central controller. It allows the controller to maintain an overview of connected devices, gather information with updates from the devices,

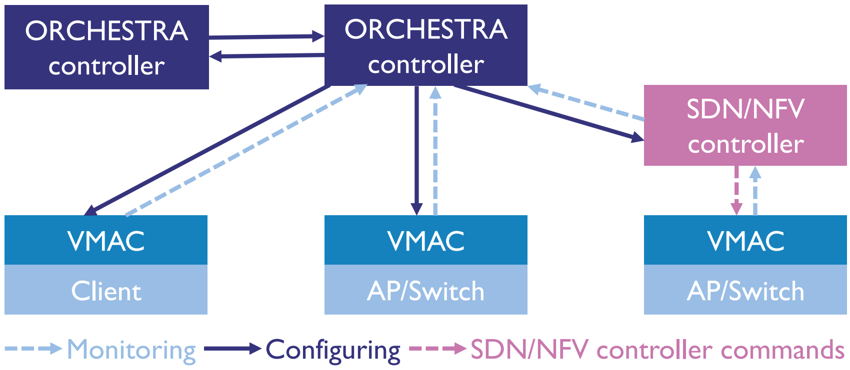


Figure 5.2: The ORCHESTRA controller and its communication interfaces to other components in a network.

and send out commands on how they should be configured to create an optimal configuration for the network. As transition towards new technology always overlaps with legacy devices still present, the controller offers communication to existing SDN controllers, for example, for IEEE 802.11 networks, as well as individual devices, such as APs or switches. As can be seen in Figure 5.2, the ORCHESTRA controller is on a new hierarchical layer that is above existing solutions. The introduction of the higher layer allows for the reuse of existing solutions and gives a central location to implement management intelligence. Existing solutions, such as MPTCP or LTE-LWA, do not offer such functionality. Scalability and reliability are essential aspects of network management as networks can become enormous. The ORCHESTRA controller supports distribution with communication between multiple instances to combat those aspects. The following sections highlight the details of the controller in terms of communication with different types of devices and management capabilities.

5.2.2.1 Communication and interfaces

First, we explore the communication aspect of the controller with different devices in the network.

VMAC: Section 5.2.1.1 explains the UDP based communication with the VMAC, which includes the discovery of each other, monitoring information from the virtual layer, and rules from the controller.

Other SDN controllers: We mentioned earlier the existence of SDN controllers in current networks and also provided examples of such in Section 2.3.3, which include ODIN and 5G-EmPOWER [3, 97]. Interfacing with these controllers is vital to support a gradient transition towards the more integrated solution that ORCHESTRA represents. The communication to an SDN controller is not as lightweight as the communication with the VMAC and requires a specific communication interface. Most controllers, such as Ryu OpenFlow and the 5G-EmPOWER controller, do not offer a northbound interface directly but offer support for applications running on top of the controller to enable higher-level functionality by using the offered features of the controller. We can use this application support to implement a northbound interface that is capable of communicating with the ORCHESTRA controller. The ORCHESTRA controller can issue commands to the SDN controller, which translates and forwards them to the devices connected with it by using the available functions. Similar, the SDN controller reports monitoring information of all devices to the ORCHESTRA controller. For wireless SDN controllers, the focus is on client handover (by using MAC addresses for identification) between endpoints. In contrast, for wired SDN controllers such as Ryu OpenFlow, the target is flow management and routing. Monitoring information can include flow information, such as IP addresses, ports, and achieved throughput, information about connected devices, their MAC addresses, and capabilities, and network conditions like the link capacity or signal strength. All of this information feeds into the global view that the ORCHESTRA controller has. It then uses it for optimizing the available networks. One can use a variety of communication frameworks and protocols to implement the communication between the ORCHESTRA controller and other SDN controllers. One famous and performant framework is ZeroMQ, which is available for many programming languages [178].

Infrastructure devices: Besides devices managed by an SDN controller, many devices have no management instance, and the controller needs to communicate directly with them. Depending on the type of device, and even the vendor, the ways to communicate with them can vary greatly. For wired devices, such as switches, the trend points towards OpenFlow as the most prominent communication protocol. This trend means that by adhering to that protocol, the ORCHESTRA controller can set flow-based rules directly on these devices. For wireless devices, such as APs, it is less clear what the standard protocol is. One popular way to configure these devices is the Network Configuration Protocol (NETCONF) with Yang as a modeling language. Yang allows specifying what a valid configuration needs to conform to be accepted by the device. Using Yang, every type of device requires its model. While it is also possible to extend the supported protocols of a device, this would require an update, which is not possible in many cases. If this is necessary, it might be a more feasible approach to install the proposed VMAC, install an alternative firmware with possibly the VMAC included, or replace the endpoint devices to move directly to an SDN solution or one with a virtual layer.

Scaling the ORCHESTRA controller: Networks, especially wireless ones, are getting continuously larger, especially with the introduction of the IoT, which will lead to deployment of significantly more devices. In such cases, a single controller can not ensure scalability and reliability as it poses a single point of failure. The ORCHESTRA controller offers distribution to solve those problems. Similar to communication with other SDN controllers, the ORCHESTRA controller maintains an eastbound interface that supports the discovery of other controllers through broadcasts and connection maintenance through heartbeats. To reduce overhead network traffic, the controllers only share information that is relevant to neighbor controllers. This information includes devices that are in the sphere of control of multiple controllers and might move from one sphere to another. A controller can request information about devices, or it can merely inform another controller that a device is leaving its sphere of control and needs to move to another controller. This situation happens if a device moves between two APs, each one controlled by a separate controller. Both controllers have information about the device and share it. If the device leaves the range of its current AP, the controller of it informs the other controller to take over the device. The new controller updates flow rules and reconfigures the AP and only after that acknowledges the handover to allow a seamless move. The initial controller simply removes the flow rules and AP configuration for the device and monitors it. The connections of the device remain without interruption. Similar to the communication with other SDN controllers, ZeroMQ, or other frameworks can be used for implementation [178].

5.2.2.2 Global view

Besides the communication interfaces, the controller has two more components, which are a data store and a northbound interface. The data store enables the previously mentioned global view by storing all monitoring information, received from each device in the network, and aggregating it into a state model. The information includes client-specific ones like throughput, latency, RSSI, infrastructure one like the number of connected clients, capabilities, such as bandwidth, and current and possible performance, and SDN controller one, which is usually the local view of the specific controller. The controller transforms and aggregates the information so that it can store it in one format in a database. Neighboring distributed ORCHESTRA controllers can request information and make their database available through a distributed database.

The northbound interface enables applications to run on top of the controller, similar to other SDN controllers. One can implement any kind of decision-making logic are complex algorithms on top of the controller by using the northbound interface and all the information the controller provides in its data store. A simple function call is enough to trigger actions such as handovers, load balancing, and duplication. The controller takes care of any underlying functionality and offers an easy to access abstract way to it. One example of such an algorithm is presented in our previous work [179]. However, one can implement all kinds of algorithms, such as Time-Division Multiple Access (TDMA)-based scheduling or an algorithm with a focus on energy efficiency. The algorithms simply provide a configuration of how the network should look like, and the controller applies it.

5.3 Applicability to wireless technologies

This section discusses how one can apply the ORCHESTRA framework and the VMAC to different technologies. The main focus lies on IEEE 802 and 3GPP technologies, the latter in the form of LTE.

5.3.1 IEEE 802

For the IEEE 802 standard, the applicability is straightforward. The standard only defines two layers, the physical layer, and the MAC layer. The first one defines the physical transmission on a signal level while the latter defines the access to the medium, mostly how interactions between multiple stations are handled. For example, the standard defines CSMA/CA for IEEE 802.11, better known as Wi-Fi, as its access mechanism. Similar is the case of other technologies, such as IEEE 802.2 (Ethernet) or wireless Personal Area Networks (PANs) technologies in IEEE 802.15. The definition of only those two layers allows using any network protocol, which is usually IP. The integration simply places the VMAC between the MAC layer and the network layer. Packets then travel through the virtual layer between those layers, and none of the underlying technologies needs to change.

5.3.2 3GPP

3GPP technologies also define a physical layer that handles the transmission and a MAC layer, which handles access in the form of Frequency Division Duplex (FDD) or Time Division Duplex (TDD) modes. Additionally, these technologies also define a management plane that handles authentication and connectivity of User Equipment (UE) to the eNB. The data plane and the management plane are separate from each other. The 3GPP standard specifies several entities to handle management. These are the UE, which is the client device, the eNB, which is the base station and wireless endpoint, and the Evolved Packet Core (EPC), which combines several management functions. The standard does not specify a network protocol. Instead, it specifies interfaces between entities that serve a particular function. For the same reason, it also specifies General Packet Radio Service (GPRS) Tunneling Protocol (GTP) tunnels for data transmission in connection with a gateway to allow connections to the outside. To be more specific, when a UE tries to connect, it communicates with the eNB, which in return requests authentication from the EPC for the UE. With a valid subscription, the eNB connects the UE to the gateway via a GTP tunnel and so enables Internet access, for example. Without a valid subscription, the UE does not get access.

The GTP tunnels are necessary for the operation of 3GPP networks due to the specification of interfaces without specifying a network protocol. It enables those technologies to use any kind of network protocol and allow for secure channels for each device. While it provided many advantages in the past, newer paradigms like edge computing, and in general, providing services closer to the edge of the network to reduce latency, can not be realized with it. Such applications would increase

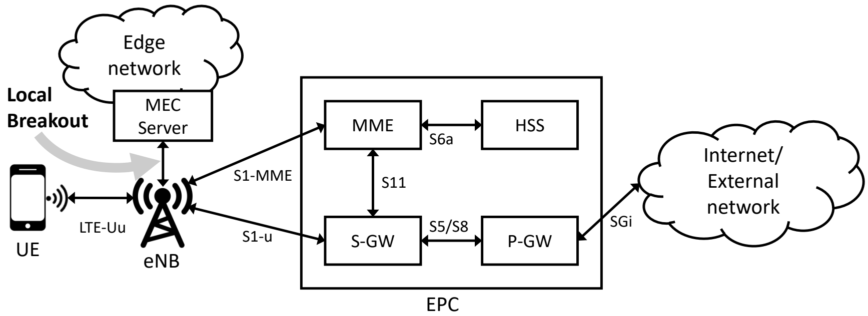


Figure 5.3: The basic MEC architecture.

the load in the core network as traffic first goes to the gateway and then back to the device close to the edge. It also brings limitations for the proposed VMAC in terms of packet-based operations. GTP tunnels are not accessible from the outside, and therefore one can not detect individual flows, which makes it impossible to aggregate flows from another technology. While theoretically, one can address this by using an additional header in the packet that includes flow information, it would add overhead and decrease network capabilities. As it is, the LTE core architecture is not compatible with the paradigm of the virtual layer, but solutions that solve the issue do exist.

5.3.2.1 MEC architecture and Local Breakout

Lee and Kim propose to break open the GTP tunnel and allow direct routing for packets [180]. This solution enables edge services with shorter routes, more control, and flexibility over network resources, and introduces an IP interface. Telecommunication operators are interested in more control and insight into data traffic, which allows for better optimization [181]. Edge computing allows for faster access to resources and lower latency for client services regarding 5G connectivity and Vehicular AdHoc Networks (VANETs) [182, 183, 184]. The Multi-access Edge Computing (MEC) architecture was proposed to fulfill these requirements (Figure 5.3). It allows us to decapsulate GTP packets in the eNB and have IP data packets readily available. Instead of routing the packets to the standard gateway, the introduction of a breakout rule changes the path, and packets are sent to the edge network, in most cases, the MEC server/gateway. 3GPP standardized this mechanism under the name Local Breakout (LBO) in Release 15 [185, 186]. It does not affect legacy operation where the eNB tunnels all traffic through GTP tunnels. The advantage of the MEC system is lower latency for edge services and lower load in the core network. While the development of the architecture was independent of existing LTE networks, several parties consider to include it in the further development of 5G technology [187]. The architecture allows for efficient edge computing, which is a crucial element for future IoT services.

The LBO of the MEC architecture allows a smooth integration of the VMAC layer in 3GPP technology. There are several ways to implement it. Placement of the virtual layer can be on the MEC server, which enables all ORCHESTRA features in the edge network, using LTE and other communication technologies. If no MEC server is present, one can use the LBO directly with ORCHESTRA-enabled devices. The LBO intercepts the packets of those devices and routes them to a VMAC in the infrastructure. For best performance, that virtual layer should be placed as close as possible to the edge of the network. This setup limits the impact of differences in latency and arrival time over the different links and helps maintain excellent performance. Depending on the network layout, a device connected to the eNB can house the VMAC, or a device in the core network, or an intermediate node between eNB and EPC, The main requirement is that a route exists for each wireless technology to reach that node so that the virtual layer can merge flows and deduplicate packets. Another advantage of the LBO is that it removes the overhead of GTP tunnels for data traffic entirely. Our prototype implementation in Section 5.5 uses the LBO to route traffic to a device connected directly to the eNB.

5.4 Use cases

After presenting the ORCHESTRA framework, we showcase its flexibility and applicability in different use cases and how. We highlight benefits to users and network operators alike.

5.4.1 Enhanced satellite networking solutions

Internet access through wired or wireless networks, like mobile networks, is still only available in limited areas, with over two-thirds of the human population having no access. For areas that have no direct access, also ships on the ocean, Internet access through satellite is often the answer [188, 189]. The first use of Geosynchronous (GEO) satellites that are stationed very high, but have a stable orbit, provided such support. They can provide a large area with Internet access, but only provide meager data rates and very high latency due to the signal distance. A more promising approach uses a hierarchical spot-beam architecture where a GEO satellite is in control of multiple Low Earth Orbit (LEO) satellites, which reduce the performance impact but support a smaller area [189]. LEO satellites are much more mobile, though, and frequent horizontal handovers are a result, which requires advanced SDN solutions for management [189]. The ORCHESTRA framework can reduce the management burden for the network operator and manage the handovers more transparently. Dual-receivers were one receiver stays connected to the old satellite while the other recalibrates and positions itself for the new satellite, have been proposed to reduce handover times. The VMAC can control both interfaces and allow for a smooth handover in this case.

Ships that get in and out of range of LTE network when they are close to the shore are a similar case. Installing the virtual layer in the edge node of the ship, the receiver that handles both satellite and mobile connections allows the parallel use of both technologies and take advantage of the better QoS of the mobile network. It also provides better performance in the local wireless network on the ship that provides Internet access for the crew and passengers.

5.4.2 Enabling autonomous driving

Intelligent assistance systems and autonomous driving are the future of the next generation of vehicles. Communication, may it be with other vehicles or infrastructure on the road, is a vital aspect of this evolution. Those vehicles will offer features such as platooning, real-time updates of road and traffic conditions, or optimal lane usage. All of them require communication with other vehicles or infrastructure. There are two major technologies for this, the older IEEE 802.11p standard (the basis for the IEEE 1609 and European ITS-G5 standard) and LTE-Vehicular (LTE-V) [24, 190]. The industry will likely deploy both technologies, and traffic can be intelligently load balanced between them and present devices. The benefits are low latency, as well as high-speed communication. A part of the transmitted data is critical and requires high reliability. The use of duplication can achieve the required reliability. The ORCHESTRA framework can also be used for Internet connectivity in the car, provided through a local endpoint, connected to several external networks.

5.4.3 Edge computing for large IoT deployments

Edge computing is a new paradigm that involves moving intelligence and computational resources closer to the edge of the network instead of in a central cloud location [191]. The benefits are lower response times, better battery life, more bandwidth efficiency, and better protection of data and privacy [191]. As such, it is considered as one of the essential enablers of large-scale IoT adoption as well as a critical component for 5G technology and currently under research [181, 187, 191]. Future deployments will include a large number of interconnected devices like sensors, intelligent displays, cameras, and end-user devices, which will have a wide array of different technologies. The ORCHESTRA framework can help in this environment, facilitating more efficient energy consumption by providing better communication and providing support for large volumes of data and users by offloading traffic through efficient inter-technology network management. A future step is provided in the discussion of Section 5.3.2.1 by integrating ORCHESTRA into the MEC architecture.

5.4.4 Extended coverage in rural areas

While in many countries the people live in large cities and therefore have an excellent wired broadband connection, a large portion of people live in rural areas as well. In many cases, these houses are more distributed, with larger distances between them. They usually have access via old DSL lines that were originally used for telephone communication and therefore are limited in speed. The demand of end-users is growing, though, but the deployment of high-speed solutions is too expensive in those areas. One solution is DSL-LTE bonding, where the connection is a hybrid between DSL and LTE, and the home gateway divides traffic among both links, which effectively increases the available bandwidth [192]. MPTCP is often the technology of choice to enable this aggregation, but it requires two endpoints, one where it splits the traffic and one where it merges it, increasing management overhead. Instead of managing each MPTCP connection for all end-users, network operators can use the ORCHESTRA framework to reduce the complexity through its transparency. It handles all interfaces without the need for additional management for merging and splitting flows and is also able to support more traffic types than TCP. Section—5.6 provides a comparison between ORCHESTRA and MPTCP.

5.4.5 Wireless community networks

IEEE 802.11 offers low-cost deployments of wireless endpoints for everybody due to its use of the unlicensed spectrum. This availability led to the emergence of so-called wireless community networks [193]. A combination of APs in infrastructure mode to provide connectivity to end-users and in mesh mode to provide backhaul connectivity enable a wide variety of services, some free and some costing money. For example, broadband connectivity for users, also over longer distances through point-to-point links, or VoIP between all users of the network. One or multiple gateways that can be connected to a mobile network, standard broadband connection, or fiber optics provide Internet excess. Large-scale events can use similar deployments where users receive wireless access through a network supported by a wireless backhaul consisting of a mesh network. ORCHESTRA can take care of the management of the wireless backhaul with transparent handovers to avoid connection drops, using load balancing on multiple paths and through different technologies to increase throughput, and duplication for traffic depending on high reliability like emergency communication. ORCHESTRA can also manage the wireless community network for end-users directly.

5.5 Prototype implementation

The Click modular router allows the implementation of arbitrary network functions from a high level with much build-in functionality to handle low-level operations and therefore allows fast prototyping [194]. The prototype implementation of the VMAC takes advantage of Click by using existing elements to do standard packet handling, like reading headers. At the same time, we introduce new elements to enable advanced functionality. Figure 5.4 shows the packet flow and general interaction of different elements. The *SuperFromDevice* and *SuperToDevice* elements exist multiple times and take care of packet forwarding from and to underlying interfaces. The packet flow for incoming and outgoing packets differ, and in the following two sections, we discuss each of them.

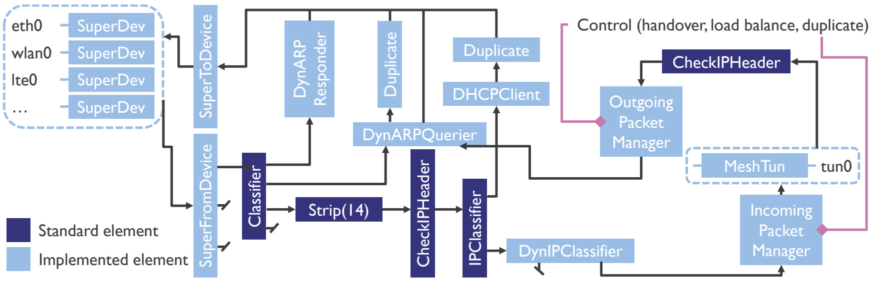


Figure 5.4: A detailed graph of the components in use and the packet flow through the components.

5.5.1 Incoming traffic

The VMAC first classifies every incoming packet in the *Classifier*, and depending on the type of packet forwards it differently. There are three types, an ARP request, an ARP reply, and a data or DHCP packet. The virtual layer forwards ARP requests immediately to the *DynARPResponder*, which sends out the reply as the VMAC integration the ARP functionality. The VMAC forwards an ARP reply to the *DynARPQuerier* as this means a packet is waiting for transmission. For both data and DHCP packets, the virtual layer strips the MAC header (*Strip(14)*) and checks and classifies the IP header (*CheckIPHeader*, *IPClassifier*). If the packet is a DHCP response to a request from the VMAC, the virtual layer forwards it to the *DHCPClient*. The VMAC forwards DHCP requests and responses not addressed to it, further to the closest DHCP server as it does not include this functionality.

If the virtual layer identifies the packet as data, it forwards it to the *IncomingPacketManager*. This component implements the advanced functionality on the receiving side, which includes deduplication and reordering of packets in the case that the sending side uses load balancing or duplication. The component also includes functionality to check for traffic from the controller that changes the configuration or any rule. If the packet is for the current node, the virtual layer forwards it to the *Tun* interface and makes it available to the next layer in the stack. Otherwise, it forwards it to the *OutgoingPacketManager*, which takes care of further forwarding the packet.

5.5.2 Outgoing traffic

The virtual layer handles outgoing traffic similar to incoming, may it be from a higher layer through the *Tun* interface or directly from the *IncomingPacketManager*. First, it checks the IP header (*CheckIPHeader*), after which it forwards it to the *OutgoingPacketManager*. This component handles the advanced functionality like load balancing, duplication, and handovers, which decides which interface will receive the packet. It also stores and applies the rules set by the controller. Afterward, the virtual layer forwards the packet to the *DynARPQuerier*, which first

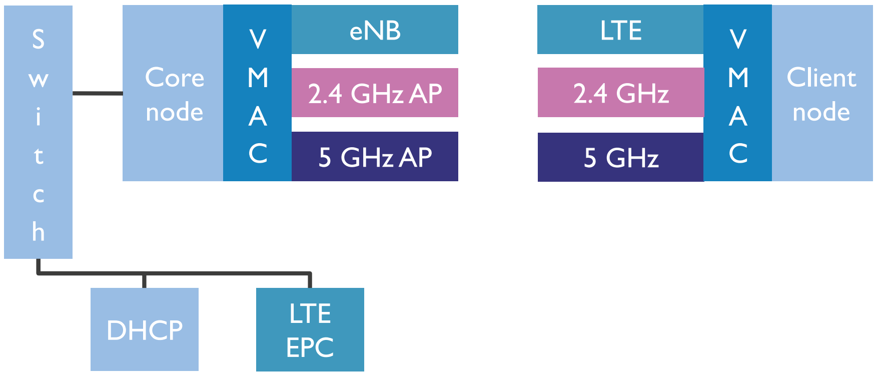


Figure 5.5: The prototype setup with all devices included.

checks if there is an entry and, if not, generates an ARP request. If there is an entry, it sends the packet to the correct interface. Otherwise, it buffers it until the ARP reply arrives and then forwards it.

5.6 Evaluation

In this section, we evaluate the capabilities of the ORCHESTRA framework compared to MPTCP for the advanced functionalities described in Section 5.2.1.2. We describe the evaluation setup first and afterward compare handovers, load balancing, and duplicating across two interfaces. The technologies in use are IEEE 802.11 and LTE.

5.6.1 Experimental setup

The setup of the prototype in Figure 5.5 consists out of two nodes with three links each and additional nodes that handle the management and authentication. The first node is the client node, which can be an end or edge device that is an IEEE 802.11 client and an LTE UE. The second node is the core node, which offers an IEEE 802.11 AP and an LTE eNB connected to it over Ethernet. The EPC handles LTE authentication and serves as the endpoint for the traffic tests while the DHCP server distributes DHCP addresses.

The AP uses an APU2c4 board with LEDE as the operating system and an IEEE 802.11n card with a 20 MHz channel [195]. The wireless interface is bridged directly to the wired interface, and the device only handles associations and traffic forwarding. The eNB uses a computer with an Intel Core i7 8700K processor, 16 GB memory, and a USRP B210 SDR with a 15 MHz channel. It uses srsLTE as a base for the eNB with added LBO functionality to break open the GTP tunnels [196]. Similarly, the EPC uses the implementation of srsLTE to manage the eNB and UE. The EPC, as well as the client and core device, uses an Intel NUC with a Core i5

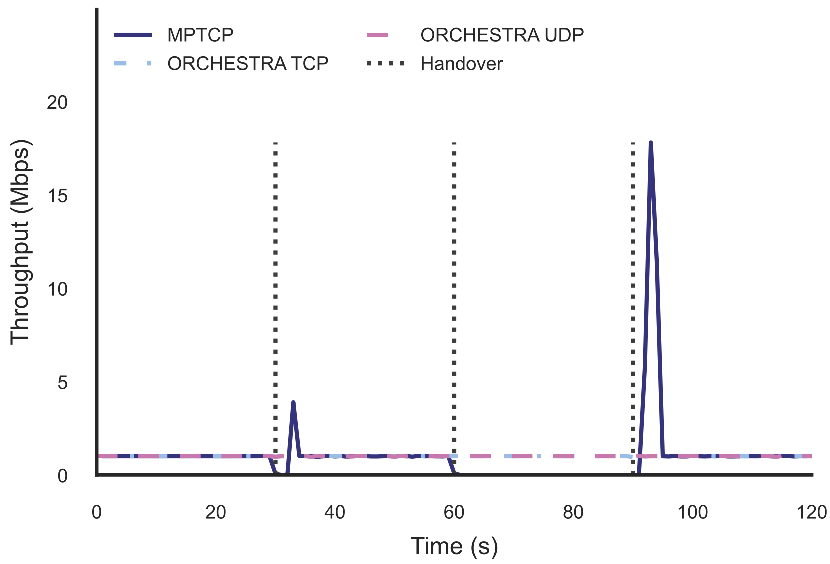


Figure 5.6: Handover performance comparing MPTCP and ORCHESTRA in terms of throughput.

4250U processor and 16 GB of memory. The UE is a Huawei E3372 LTE USB stick, and the DHCP server is a generic home router.

All following scenarios use LTE and IEEE 802.11 on the 5 GHz band for the two interfaces. We use *iperf* with TCP and UDP streams and compare it against MPTCP in version 0.94 [197]. The testing environment is in an office space with a distance of 2 m between the two nodes, and we repeated and averaged each scenario ten times.

5.6.2 Seamless and transparent multi-technology handovers

This scenario represents handovers, through, for example, mobility or environmental influences, between IEEE 802.11 and LTE. A handover happens every 30 s, which is highlighted by the vertical lines in the figures, and we use a 1 Mbps stream for 120 s. Figures 5.6 and 5.7 present the throughput and latency results for MPTCP and ORCHESTRA with TCP and UDP. MPTCP can use any available interface and, therefore, can do limited handovers by using another link. In Figure 5.6, we can see that the handover takes time as MPTCP first needs to lose connection and re-establish a connection over the new link. Creating a new sub-flow comes at a cost, though; packets are waiting for transmission, and therefore the latency spikes when a handover is occurring. ORCHESTRA, on the other hand, can handover seamlessly and has neither a throughput drop nor a latency spike. The downtime for each solution is 21 % and 0 % for MPTCP and ORCHESTRA, respectively. The

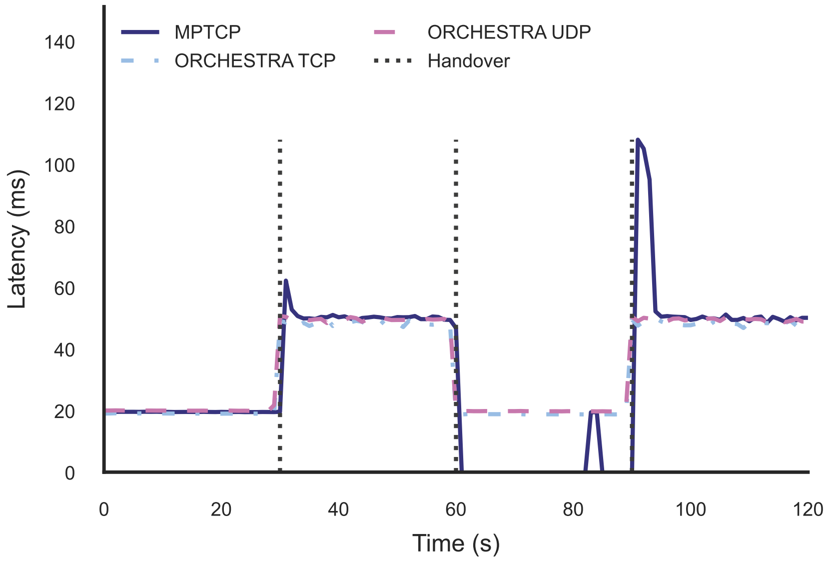


Figure 5.7: Handover performance comparing MPTCP and ORCHESTRA in terms of latency.

handovers at the 30 s and 90 s mark, from Wi-Fi to LTE, incur a connection loss of 3 s and 2 s, which is in line reported in the literature [198, 199]. The second handover, from LTE to Wi-Fi, at the 60 s mark, however, has a downtime of 30 s, until the next handover. It is unclear why this high downtime occurs, and this may be due to a misconfiguration. While ORCHESTRA can provide seamless connectivity for TCP and UDP, MPTCP only provides support for TCP.

As ORCHESTRA is capable of operating with legacy devices, we have a second handover scenario where we tested three configurations of an AP and a station that show the difference in handover times. This configuration is most likely to occur in a LAN setting with newer and older devices. The three configurations consist of two ORCHESTRA-enabled devices (AP and station), one ORCHESTRA-enabled and one legacy device (ORCHESTRA AP, legacy client), and two legacy devices. We use Ubuntu as an operating system that reacts very slow to connection drops when multiple technologies are available. Wireless connections can be down for 15 s or more (experimentally determined), which results in a complete connection drop with *iperf*. To be more in line with end-user oriented operating systems like Windows and macOS, which usually react faster, we simulated the handover of legacy devices. To avoid complete connection loss, we used a script to monitor the active link continuously and switched the route to the correct interface if there was no traffic for four seconds. This approach, without the monitoring script, has similarities to band steering, where the AP tries to force a station to a specific

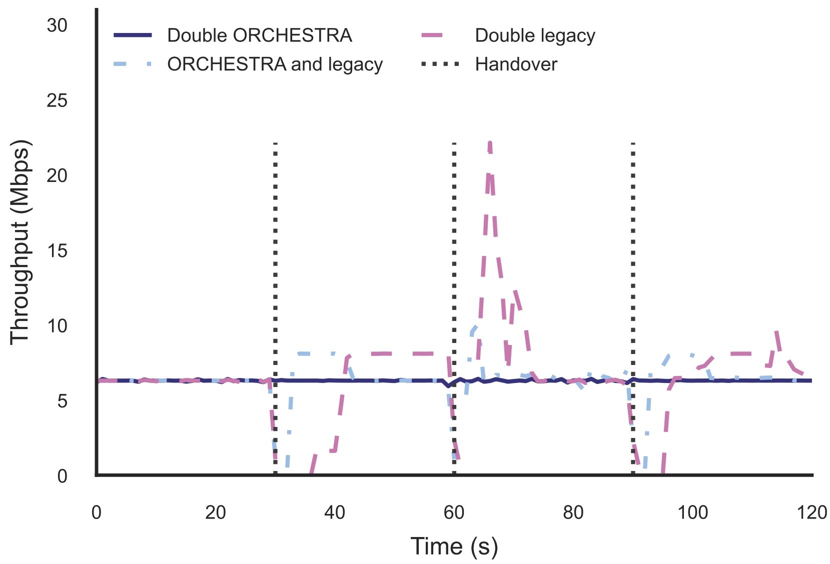


Figure 5.8: Handover performance comparing ORCHESTRA and legacy devices with TCP traffic.

frequency band. We provide throughput results for TCP and UDP with a traffic flow of 6 Mbps.

Figures 5.8 and 5.9 show the results for TCP and UDP with differences between protocols and device combinations. When two legacy devices are in use, the traffic drops completely after a handover as the underlying connection is lost, and the device needs to re-establish it. If *iperf* uses TCP, it overcompensates and increases the traffic after the downtime to achieve an average throughput of 6 Mbps. Figure 5.8 shows a maximum throughput of 23 Mbps on IEEE 802.11 and 8 Mbps on LTE, which is the limit for this LTE link. The connection loss and re-establishment need to be handled by the application itself, which is usually higher than lower layers, and not all applications can handle it. UDP is less affected as it does not require ACKs, but it still experiences a throughput drop, which results in packet loss in this case.

Changing to one device with a VMAC and one legacy device, we can see an improvement in the handover time, especially in the case of TCP in Figure 5.8. The device with VMAC can switch instantly to another technology to send packets and decrease the downtime. While a connection drop is not entirely avoidable due to the legacy device having no information about it, the ORCHESTRA-enabled device can re-establish the connection much faster as it is aware of it, and therefore, can steer the legacy device. When two ORCHESTRA-enabled devices are in use, the handover is seamless, and there is no throughput drop, with both TCP and UDP

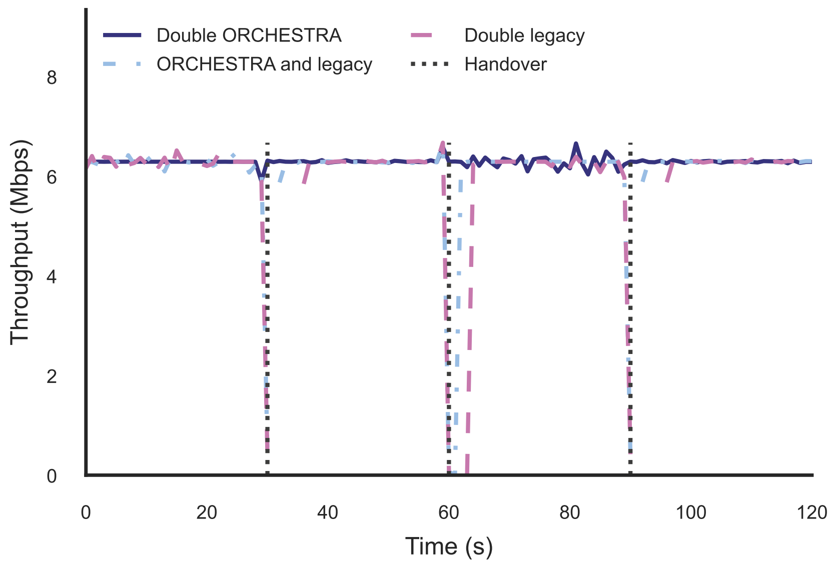


Figure 5.9: Handover performance comparing ORCHESTRA and legacy devices with UDP traffic.

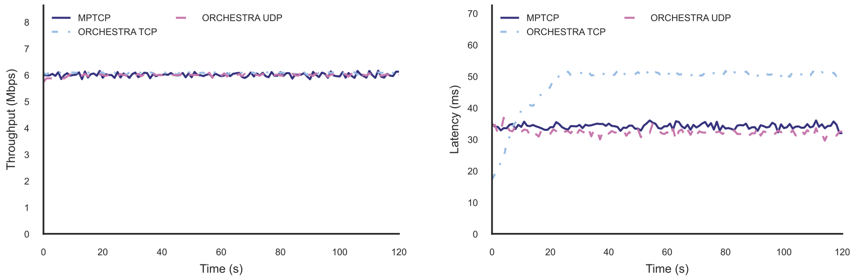
(Figures 5.8 5.9). No overcompensation of *iperf* is necessary, and the application is not responsible to re-establish a new connection anymore. A central controller or the network operator is responsible for the decision.

5.6.3 Fine-grained packet-level load balancing

Next, we showcase packet-level load balancing by the use of two simultaneous active interfaces. The first scenario uses TCP and UDP and fixes the weights on each interface to be evenly distributed (50/50 %). We use MPTCP with the default round-robin scheduler, sending a fixed number of packets over one interface and then rotating to the next, as a comparison. We use a traffic flow of 6 Mbps.

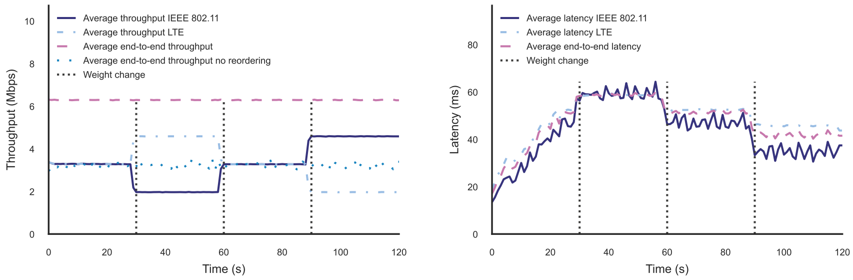
Figures 5.10a and 5.10b show the results for throughput and latency. Both solutions achieve the full rate of 6 Mbps throughput, while ORCHESTRA is capable of doing it for TCP and UDP alike. The latency in Figure 5.10b shows an increased latency (40.6 %) for TCP in combination with ORCHESTRA. In this case, there is a build-up for the latency for the first 25 s, after which it stabilizes. ORCHESTRA with UDP traffic and MPTCP exhibit a lower latency in the experiments.

There are two parts to the explanation of the behavior that shows increased latency with ORCHESTRA in combination with TCP. First, the latency properties of links become an essential part of successfully load balancing across those links because packets might arrive out-of-order. In our case, the latency properties of IEEE 802.11 and LTE are very different, as can be seen in Figure 5.7. How each



(a) Throughput. (b) Latency.

Figure 5.10: Load balancing performance comparing MPTCP and ORCHESTRA.



(a) Throughput. (b) Latency.

Figure 5.11: Load balancing 6 Mbps TCP flow over IEEE 802.11 and LTE with a weight change from 50/50 to 30/70 to 50/50 to 70/30.

solution reacts to it determines the overall latency that it can achieve. MPTCP uses its scheduler to determine how many consecutive packets it sends out on an interface before switching. The result is unequal distribution, around 70 % on IEEE 802.11, and 30 % on LTE. Compared to that, ORCHESTRA indeed sends out an even distribution, not favoring any technology. The behavior of MPTCP using IEEE 802.11 with a higher weight reduces the average latency as this link has a lower base latency. MPTCP also sends out more packets sequentially before switching the interface, which reduces the need for reordering as those packets always arrive in order. The fluctuations in latency with MPTCP hint at the differences in latency when the scheduler switches.

Second, as ORCHESTRA splits the flow equally, packets will arrive out-of-order with TCP, and the VMAC needs to reorder them to ensure high performance. Section 5.2.1.2 dicusses this mechanism. The virtual layer uses the TCP sequence number to order packets. Out-of-order packets are stored in a hash map until the missing packet arrives, or a timeout is triggered. Reordering frequently happens with equal distribution and different latency properties on links, which causes an

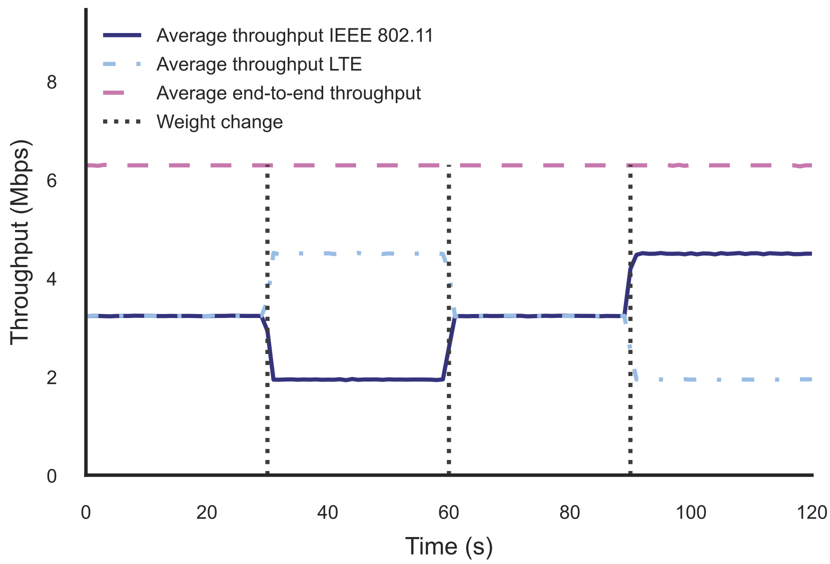


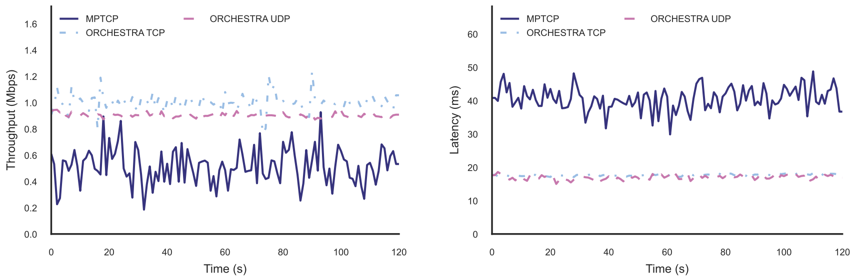
Figure 5.12: Load balancing 6 Mbps UDP flow over IEEE 802.11 and LTE with a weight change from 50/50 to 30/70 to 50/50 to 70/30.

increase in latency in the beginning. Additionally, the TCP rate control mechanism causes slight fluctuations throughout the experiment as it reacts to the varying inter-packet arrival times. UDP does not have a rate control algorithm and no sequence numbers, which means it does not show this behavior.

The next experiment in Figures 5.11a, 5.11b, and 5.12 highlight the packet-level load balancing capabilities and sheds light on the impact of reordering by switching the weights during the experiment. We use a 6 Mbps stream, both TCP and UDP, split across the IEEE 802.11 and LTE interface. We start the experiment with equal distribution, then switch to a 30/70 % distribution after 30 s for IEEE 802.11 and LTE, respectively. The next switch at 60 s levels out the distribution again, and at mark 90 s, we switch to a 70/30 % distribution for IEEE 802.11 and LTE, respectively.

The throughput in Figures 5.11a and 5.12 follows the set weights correctly and shows that the packet-level load balancing works as intended. Both protocols, TCP and UDP, achieve the full specified throughput of 6 Mbps. Figure 5.11a also includes the throughput of TCP without reordering, which only achieves 3.2 Mbps and clearly shows that reordering is necessary; otherwise, TCP lowers the control rate. Figure 5.11b shows the latency, which shows a similar increase in the first 25 s compared to the previous experiment. After the increase, the latency depends on which technology is in use.

Summarizing, the packet-level load balancing with reordering offers increased



(a) Throughput.

(b) Latency.

Figure 5.13: Duplication performance comparing MPTCP and ORCHESTRA.

throughput and flexibility with a penalty of an increased latency compared to MPTCP, where the scheduler sets the weights. Later on, we will demonstrate a mechanism to close the gap in latency between both solutions under the same conditions. The evaluation also shows that the abstraction of the VMAC serves its intended purpose and that it can to load balance UDP and TCP on a packet-level without affecting performance. Future work can extend the load balancing by introducing a time-based scheduler to minimize interference from external sources.

5.6.4 Duplication of critical data in unreliable environments

The duplication scenario is similar to the load balancing scenario where two interfaces (IEEE 802.11 and LTE) are active at the same time. While the load balancing mechanism switches between the interfaces, the duplication mechanism copies each packet and sends it out over all interfaces. To verify the duplication capabilities of the VMAC, we create an unstable environment by dropping packets with a 25 % chance per packet on each link. We use a flow of 1 Mbps with TCP as well as UDP. For both protocols, the receiving VMAC needs to detect and drop duplicated packets. We compare the performance against MPTCP by using the redundant scheduler instead of the default scheduler [200]. This scheduler uses all available sub-flows to send the data over and establishes back-up sub-flows for retransmissions.

Figures 5.13a and 5.13b show the throughput and latency of this experiment. We can immediately see that MPTCP can not achieve the desired throughput of 1 Mbps and only achieves around half of it with 0.5 Mbps. This performance decrease goes hand in hand with a significant increase in latency due to retransmissions. ORCHESTRA, on the other hand, achieves the desired throughput with both UDP and TCP. The latency for both protocols also stays below 20 ms. The difference between ORCHESTRA compared to MPTCP is due to the way the duplication works. ORCHESTRA duplicates the packets without the transport protocol being aware of it, while MPTCP uses TCP sub-flows that all suffer from packet loss. However, while ORCHESTRA significantly improves reliability, there are still fluctuations that indicate that both links lost the packet, which can happen with a chance of 6.25 %.

5.7 Conclusion

This chapter presented the ORCHESTRA framework for inter-technology network management, which is composed of two components; the VMAC and the centralized controller. On the one hand, the VMAC enables the abstraction of underlying technology by providing a single interface to a higher layer and transparently managing each interface. On the other hand, the controller provides a global view of the network and central coordination and intelligence. The VMAC, in combination with the controller, enables advanced functionality like duplication, packet-level load balancing, and seamless inter-technology handovers. The presented prototype is capable of using IEEE 802.11 LTE. It demonstrates that all functionality works as intended for the TCP and UDP transport protocols while outperforming the de facto standard MPTCP.

Disclaimer

The contributions to the ORCHESTRA framework, presented in this chapter, were done equally by Tom De Schepper, Ensar Zeljković, and myself.

6

Machine Learning based load balancing over links with different latency properties

The contributions presented in this chapter are based on the publication titled *"A machine learning approach for optimizing latency and inter-packet arrival rate in TCP multi-path load balancing"*.

6.1 Introduction

The amount of available technologies on devices allows for advanced functionality, such as load balancing, and the presented ORCHESTRA framework. The previous chapter presented a framework for technology abstraction that includes transparent packet-level load balancing over multiple links. These links can have different QoS properties. For example, consumer technologies such as IEEE 802.11, better known as Wi-Fi, and LTE are both intended for high throughput but differ in actual performance. They also show significant differences in latency. Other technologies show even more differences in throughput, latency, and other properties. The central controller can manage throughput differences as it knows the available throughput of a technology and an application's requirements.

On the other hand, latency is more challenging to manage, and it is an essential aspect of load balancing. Applications widely use TCP due to its reliability. Sequence numbers in the packet header, acknowledgments, and re-transmissions if a packet is lost or did not arrive in time achieve this reliability. If the protocol

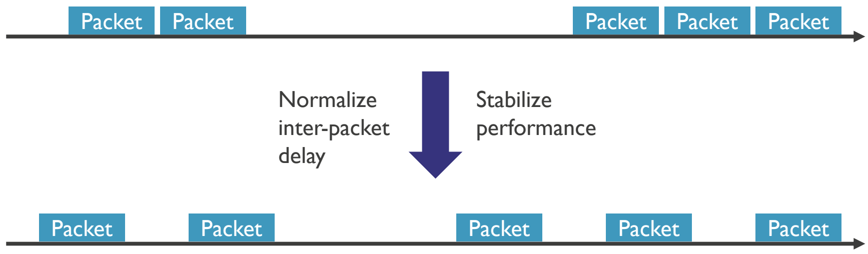


Figure 6.1: Reordered packets arrive with large delays between packets (top). Adding time between packets reduces the delay between packets and achieves better performance (bottom).

considers a packet lost, it reduces the sending rate. This mechanism can cause problems in load balancing as packets can potentially arrive out-of-order, and there can also be a considerable delay between two packets. In both cases, the protocol considers packets lost, and the result is performance degradation. To combat this, we need to reorder packets, which usually results in packets on the faster path waiting on packets on the slower path. Ideally, we need to reduce the delay between two packets to provide a better user experience in real-time applications such as audio or video conferences. One can argue that it would still be beneficial as many applications assume in-order arrival even for unreliable protocols.

Currently, existing load balancing approaches (e.g., MPTCP, ORCHESTRA) fail to manage the latency difference. The result can be performance degradation due to packets timing out given the large variability in the inter-arrival time. Existing load balancing approaches address this issue by reducing the flexibility in the load balancing process. MPTCP, for example, avoids packet losses and latency differences by not sending in a real round-robin way but sending a long stream of packets over a link before switching. Additionally, the protocol chooses the weights for the links, resulting in unfavorable weights that do not consider other applications. As a result, the use of MPTCP reduces the available flexibility, which can hurt overall performance in an extensive managed network.

For this purpose, we propose a mechanism that limits latency differences and inter-packet arrival times but still provides the highest level of flexibility in load balancing (i.e., allowing per packet load balancing). We do this by predicting the future packet arrival rate and limiting the arrival time variability by delaying specific packets. We call this packet normalization. Figure 6.1 illustrates the overall packet normalizing approach. The top shows the effect of only reordering, while the bottom shows the effect of normalizing the inter-packet delays. The result is a better distribution, which results in fewer packets considered lost. The two combined links show only a single latency property to the transport protocol layer. We use machine learning to predict the future packet arrival rate on the receiving node that reorders packets to achieve this result. We then adjust an artificial delay that releases packets from a queue with a slight inter-packet delay. This way, we avoid bursts of packets suddenly arriving at the receiving application and slightly

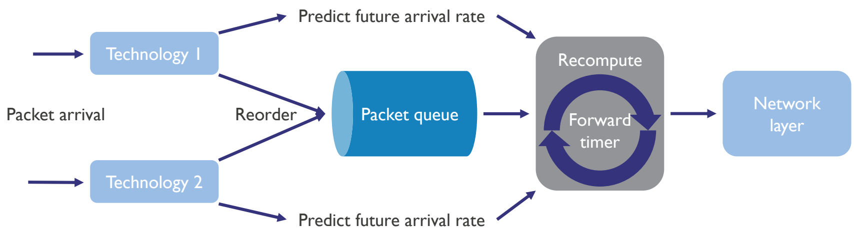


Figure 6.2: Load balancing architecture using a queue, packet arrival prediction, and a forwarding timer to normalize packet arrival to the network layer.

reduce latency as we achieve a more normal distribution of packet arrivals and, therefore, a more steady TCP behavior.

The rest of this chapter is structured as follows. First, we present the architecture in Section 6.2. We explain the reordering scheme, the packet arrival prediction, what type of data we used for training, how we gathered it, the features we extracted, and how we compute the artificial delay. Afterward, we evaluate the mechanism's implementation, which we integrate into the previously presented VMAC, in the same prototype that we used for the ORCHESTRA evaluation in Section 6.3.

6.2 Architecture

In this section, we will describe the general architecture of our approach. First, we describe the high-level architecture, then the reordering process, followed by predicting the future packet arrival rate, and afterward, the normalization with the calculation of the artificial delay. This chapter's architecture is embedded in the ORCHESTRA framework and its VMAC layer, presented in Chapter 5.

Figure 6.2 shows an overview of the high-level architecture. Packets on a networking node arrive on multiple links that another networking node previously sent according to a load balance scheme, which the central controller defines according to the available resources' best use. According to a reordering scheme, the node reorders the packets, which usually involves waiting on packets and then puts them in a queue. The queue releases the packets and forwards them to the network layer according to a forward timer. The node calculates the timer every 50 ms based on the information available from this interval. This behavior leads to a more regular arrival rate on the transport layer or the next networking node.

The goal of the approach is to reduce the number of packets that currently wait in the queue but ensure that a packet is at the head of the queue to be forwarded to the next layer as soon as the forward timer is up. Additionally, each packet's added latency needs to be kept at a minimum to prevent that it outweighs the benefits of a normalized packet arrival rate. For this purpose, the forward timer calculation needs to consider several parameters. First, the packets that will arrive in the future on each technology. They are not accurately known, so we predict them using machine learning. Second, the packets in the queue or, more generally, the queue size. These are the packets that are in-order and ready to be sent out. This number should neither be zero nor should it be too high. Third, the packets that arrived at the node but are not in the queue yet. These packets arrived out-of-order and need to wait for the correct packet to arrive first.

6.2.1 Reordering scheme

We mentioned earlier that different links have different performance properties and that latency is the most important for load balancing TCP flows. Sequence numbers and acknowledgments that ensure reliability need to arrive in-order and on time so that no performance degradation happens. Therefore, to achieve high performance, a node needs to reorder packets if one link has a higher latency than the other. In general, this means that packets on the faster path need to wait for packets on the slower path.

As mentioned in the previous chapter, a node needs to consider and store two components to reorder TCP streams. The first component is the current sequence number of the packet that TCP expects and is next in line for pushing in the queue. While the second is the packets with a higher sequence number than the current one waiting to be submitted. When packets arrive, the node checks the sequence number against the current sequence number, and if it is correct, it will immediately push the packet to the queue. Otherwise, it stores the packet. Each packet has a timeout, after which the node pushes it to the queue regardless if a previous packet is missing or not. This timeout is dynamic and calculated based on the average time packets spend waiting to be reordered and pushed to the queue. We add 50 % to the average time as tolerance to ensure that packets taking slightly longer than average do not immediately time out. We determined the percentage experimentally, and a value of 50 % includes all regular packets but excludes outliers, which would increase the latency for all other packets. The node recalculates the timeout every 250 ms in our prototype. This time frame ensures that enough packets are present to calculate a reliable timeout. If the recalculation time is much lower, it is possible that not enough packets are present, and it would skew the resulting timeout, possibly degrading performance. If the packet reaches the timeout, the node considers the previous packet lost, and the node pushes the following packets to the queue. It then updates the current sequence number with the last package that was pushed. After the node pushed the packet with the current sequence number, it pushes all other packets until the next sequence number is missing. The node executes this process for each TCP stream separately.

This scheme allows for reordering to be applied in any scenario and, therefore, freely chosen weights for each link, giving a centralized controller full control. However, it does rely on a mechanism to minimize additional packet latency in the packet queue; otherwise, packets might experience high latency. We will discuss this mechanism in the next two sections.

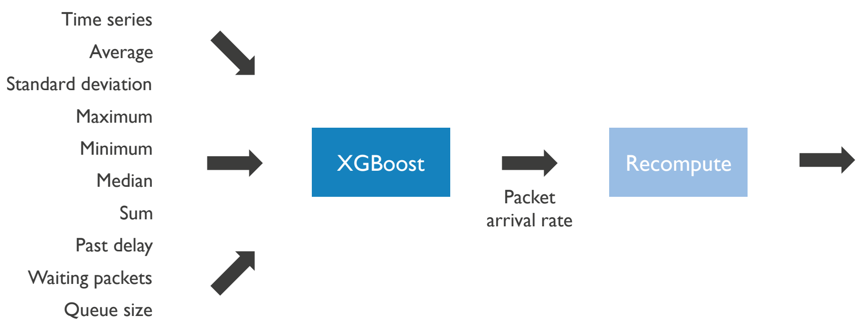


Figure 6.3: Input and output of the machine learning based future packet arrival prediction.

6.2.2 Future packet arrival prediction

One component in calculating the forward timer is the future packet arrival prediction, depicted in Figure 6.3. This component needs to compute the prediction as fast as possible, not to incur any computational overhead that might impact latency and to be able to react to changing network conditions. We, therefore, focus on execution speed and favor it compared to accuracy. The task at hand is a regression that uses a time series plus additional features as input to get a continuous number of packets as output.

XGBoost is a well-known tree boosting algorithm that is highly optimized for performance and low computational times and performs well in real-life scenarios [163]. Its performance allows for sub-millisecond predictions, which are necessary for our use case. Additionally, the robustness of the models helps in achieving our goals. It supports both classification and regression tasks based on arbitrary features.

There are two choices for the model, either one model for all technologies or one model for each technology. Especially with technologies that exhibit very different behaviors, fitting one model is much more complicated than using different models. Regarding extensibility, one model has the disadvantage that each time one adds a new technology, one needs to retrain the model for all technologies. In comparison, one model for each technology allows for training the new technology's model only and reuse the existing models for the other technologies. Using multiple models also allows for models to be smaller and less complex, as they only need to model one technology's behavior. Therefore each model predicts faster than using one model several times. The model's size is also an essential aspect as communication devices are often embedded devices with minimal storage capacity. For these reasons, we chose one model for each technology.

6.2.2.1 Data generation and preparation

For data generation, we used the same prototype that we also use in the validation later on in Section 6.3.1. This setup ensures that the data used for training and testing our model behaves the same as the data used in validating the scheme. As TCP itself is adaptive and reacts to changes, data generation needs to include the protocol's behavior to a changing artificial delay. To correctly map the TCP' behavior and make it available in training, we generated data by randomizing the artificial delay between packets that are forwarded within a range of 0 ms to 25 ms while sending constant bitrate *iperf* traffic with different transmission rates, ranging from 6 Mbps to 12 Mbps. TCP reacts to the changes and adjusts the transmission rate of packets, which is then mapped in our data. Depending on the delay's value, for example, 25 ms, the reactions were significant, and in many cases, the throughput could not be maintained. However, even smaller changes of 5 ms already prompt a reaction of TCP. This choice of data generation limits the data to a fixed interval, though, the one with which it was generated. Therefore, we need to choose the interval carefully beforehand.

Our initial investigation discovered that while longer intervals can achieve higher accuracy, it has one major drawback. If the prediction and the artificial delay calculation are off too far, packets fill up the queue, and the added latency to a packet is significant. Also, TCP will react immediately to the change, and throughput will throttle. As we can not avoid this possibility entirely, a lower interval that can react to possible wrong values fast is much more advantageous than achieving higher accuracy. For this reason, we chose an interval of 50 ms, which is fast enough to correct possible wrong predictions and long enough to achieve a high enough accuracy.

The training, validation, and testing data consist of 800 runs of 60 seconds, with four different bandwidths ranging from 6 Mbps to 12 Mbps with increments of 2 Mbps. We split the data set using 80 % for training and 10 % each for validation and testing.

6.2.2.2 Features

From the generated data, we can extract several features that we can use for the model. Figure 6.3 presents an overview of them. First, we have the raw packet arrivals, which means every packet that arrived over the last interval. To properly use them, we aggregate them together based on their arrival time into buckets of 1 ms to get a time series on a fixed basis. In the case of the recomputation interval being 50 ms, we have 50 features that represent each a millisecond into the past from the current time. Additionally, to this time series, we can derive multiple features from the series itself. We can calculate and use the average, standard deviation, maximum, minimum, median, and sum in training. These features are aggregation features that give us more information about the time series of packet arrivals over the last interval. These features aim to give the model less granular information about the time series to lead to better predictions. The time series itself

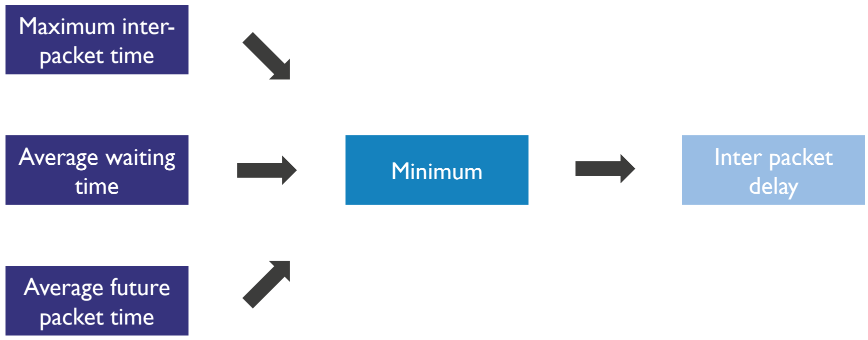


Figure 6.4: Different components for the inter-packet delay computation.

can contain many zeros, making it more difficult for the model to learn. We chose these features as TCP's future performance is strongly linked with past performance, which is the main idea behind TCP's adaption algorithm. By generating the data in the prototype, we can get additional features that include past decisions and TCP's effect on it, incorporating its behavior. These features are the past chosen artificial delay, the packets currently waiting to be reordered, and the queue size. All of these features are not directly available to TCP, but they all influence its performance and mainly reflect the consequences of our past decisions. The past chosen artificial delay combined with the current situation is intended to allow the model to learn about possible mistakes in the past and how to fix those mistakes. The packets currently waiting to be reordered reflect the situation about the technologies and their interdependence. There will always be packets that wait to be reordered as there are differences in latency between the technologies. If the packet with the correct sequence number arrives, these packets will need to be forwarded. This scenario means that suddenly many packets are in the queue and could potentially lead to higher latency. This feature is intended to avoid that and encourage the model to consider these waiting packets. The queue size consists of the packets that are ready to be forwarded and can get larger if the previous artificial delay was too big. This situation needs to be avoided at all costs as it significantly increases latency. In case one of the model's decisions was wrong, these features reflect this, and the model will try to incorporate and fix it. Later on, we evaluate the importance of each feature.

6.2.3 Artificial delay computation

The artificial delay computation is a crucial part of normalizing the packet arrival rate and avoiding bursts. Our scheme's packet queue mainly functions as a buffer for varying network conditions, similar to a buffer in video streams, but on a much smaller scale. For example, several packets may arrive shortly after each other, but the following packet is delayed because another device occupies the medium. Especially in networks where there is no centralized management through

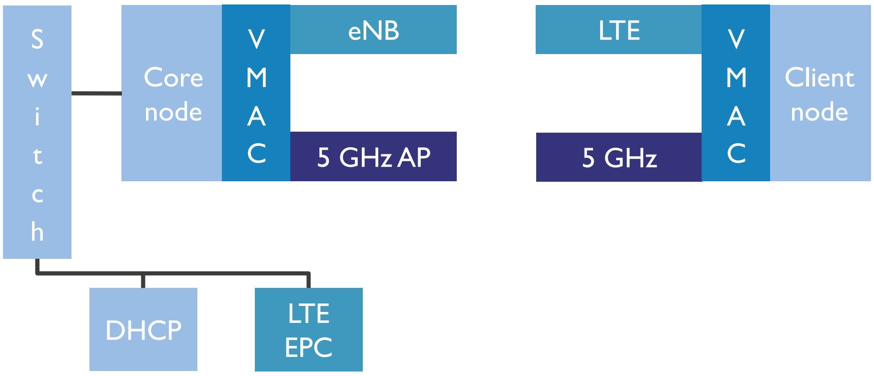


Figure 6.5: The prototype used in the experiments for data generation and evaluation. The VMAC includes the presented architecture and prediction.

a scheduler, such as in IEEE 802.11 networks, latency can vary significantly. Networks with a centralized scheduler, like LTE, have a more predictable latency as the central scheduler manages each device's sending times. The variation between packets can still be considerable, especially if many clients try to send data.

Different components influence the forward timer's calculation (as seen in Figure 6.2) that regularly releases a packet from the queue. We will consider the predicted future packets to get an estimate about future TCP behavior. However, we also need to include the current situation, which can be negatively affected by the previous prediction and computation. As the goal is to normalize packets and reduce the latency, we need to keep added latency at an absolute minimum. Therefore, we need to consider the packets currently in the queue and the packets waiting to be reordered for the next interval.

Figure 6.4 shows the three necessary components. First, we introduce a parameter d_{max}, representing the maximum added latency that a packet should have. We can choose this parameter manually, and besides influencing the end-to-end latency, it also influences the packet inter-arrival rate. A higher value leads to higher latency but a more even packet inter-arrival rate, while a low value leads to lower latency and less even packet arrivals. According to QoS needs, a controller or network operator can use this parameter to optimize the network one way or the other.

In the case that the queue is not empty, we calculate the average time q_{avg} that a packet in the queue should have so that we can achieve zero waiting packets for the next interval by using d_{max} and the size of the queue q_s

$$q_{avg} = \frac{d_{max}}{q_s} \tag{6.1}$$

otherwise, we set it to d_{max}.

Similar, if packets are waiting to be reordered, we also calculate their average time w_{avg} by using d_{max} and the number of packets waiting to be reordered and

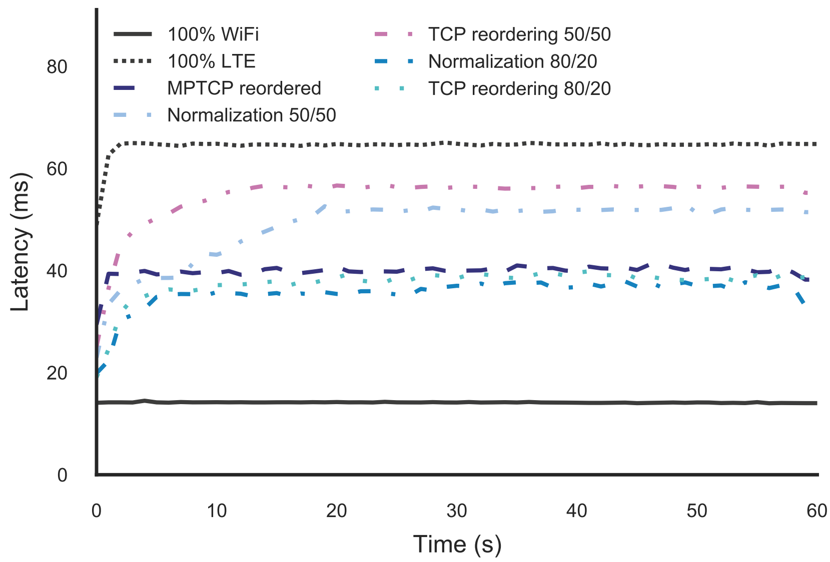


Figure 6.6: The achieved latency of different solutions. Our solution supports flexible weights (50 %/50 % and 80 %/20 % IEEE 802.11 and LTE) and outperforms MPTCP if both solutions' weights are the same.

soon forwarded to the queue w_s

$$w_{avg} = \frac{d_{max}}{w_s} \tag{6.2}$$

which is set to d_{max} if there are no packets.

Using the interval between recomputations of the artificial delay i in microseconds and the number of packets that will arrive in the next interval, predicted by our model, f_s, we can calculate the average delay for future packets so that we can have a minimum number of packets in the queue at the time of the next recomputation of the artificial delay

$$f_{avg} = \frac{i - (q_{avg} \cdot q_s) - (w_{avg} \cdot w_s)}{f_s} \tag{6.3}$$

where the impact of the queue or the waiting packets cancels out if they are zero.

To ensure that we meet all conditions, we then take the minimum artificial delay

$$\min(q_{avg}, w_{avg}, f_{avg}) \tag{6.4}$$

If there were no packets in the past interval and there are no packets predicted in the next interval, we set the artificial delay to zero as we can not calculate an accurate value. This value also ensures that we do not have an artificially inflated

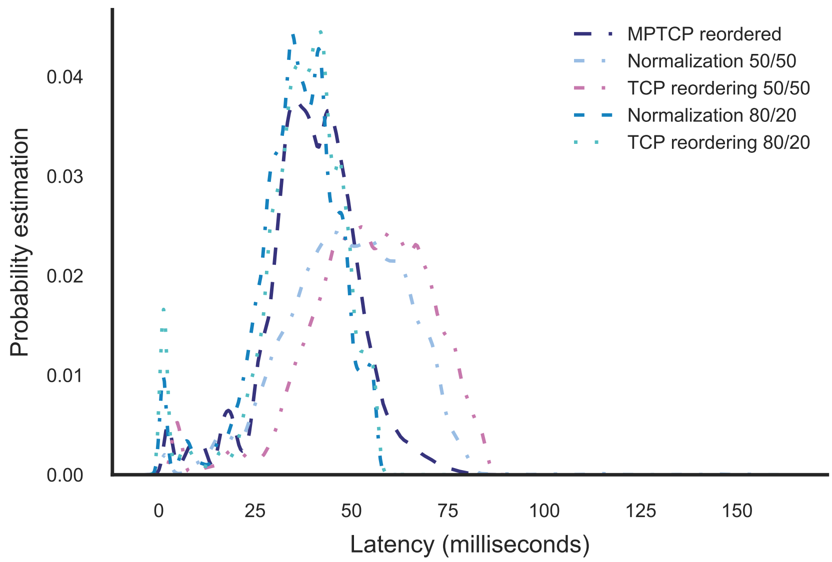


Figure 6.7: Latency distribution presented as probability estimation (KDE), confirming our solution's improvement compared to standard TCP.

delay at the beginning of a transmission. At the start of a transmission, this only impacts the first 50 ms; afterward, we can use the previously described calculation.

6.3 Evaluation

This section presents the prototype used in generating data and evaluating the models and the results themselves.

6.3.1 Experimental setup

Figure 6.5 shows the prototype used in experimentation. We used IEEE 802.11 and LTE to demonstrate different latency properties in different architectures in general. The client, which consists of a NUC running Ubuntu 18.04, uses two USB sticks, one for IEEE 802.11 in the 5 GHz spectrum and one Huawei LTE stick, for connection. The virtual MAC layer, implemented using Click, abstracts them into one connection and applies the load balancing scheme [194].

The core or receiver node consists of a NUC, running Ubuntu 18.04, with a connected AP, running OpenWRT, and another connected machine with an Intel 8700k to act as the eNB for LTE with a B210 USRP attached, using srsLTE. The virtual MAC layer on the receiver node is applying the reordering, queuing, and

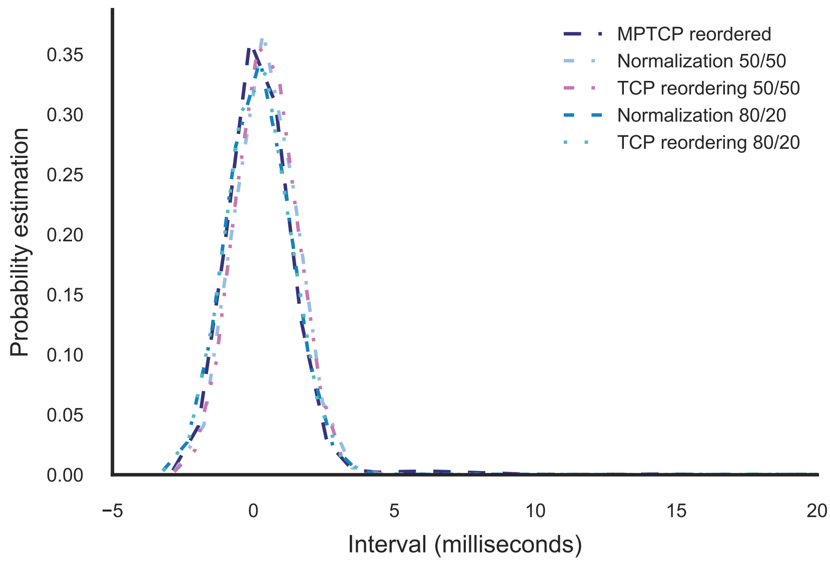


Figure 6.8: Inter packet arrival distribution as probability estimation (KDE) on receiver showing that standard TCP has significant burst arrival, followed by MPTCP, while our solution can improve and normalize it. The figure excludes the outliers above 50 ms.

normalization schemes. Additionally, a switch, a DHCP server, and the EPC are needed to ensure connectivity.

In our experiments, we compare our solutions against MPTCP, TCP with only reordering in the virtual layer and as reference IEEE 802.11 and LTE each without load balancing. We compared a 6 Mpbs stream from the client to the EPC on all solutions and repeated each scenario ten times. We split the throughput equally among the links, except for MPTCP, where the scheduler decides the distribution. We do not compare against TCP without reordering as it can not even reach 6 Mpbs due to too many packets arriving out of order. We used synchronized clocks on each device. We then measured the latency by each packet's timestamps, which we captured with *tcpdump* on the MAC layer. For MPTCP, we took an additional step to get the latency. As packets arrive out of order in this case and only the transport layer reorders them, we calculated the latency by reordering first and then taking the best timestamp for each packet. For example, if two packets arrive out of order, both will have the same timestamp of arrival. This calculation leads to a best-case scenario for the latency of MPTCP and can be worse in reality.

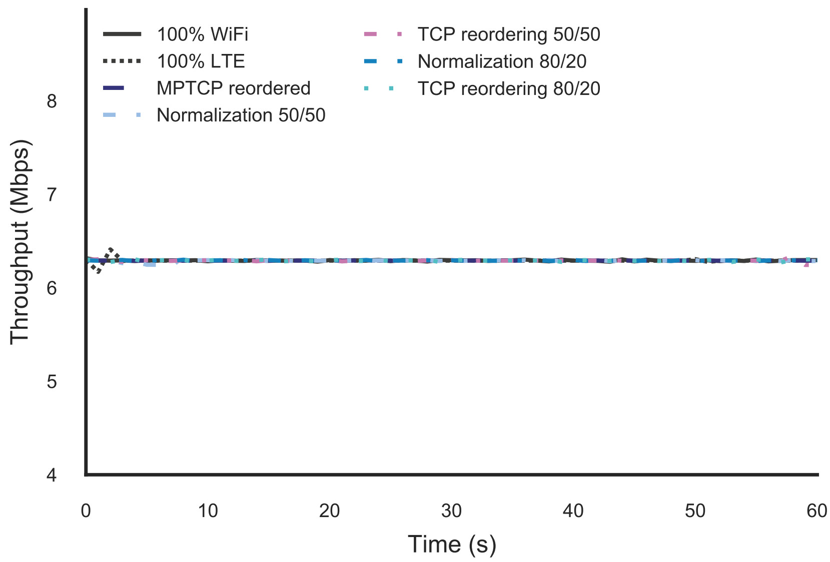


Figure 6.9: Throughput of different solutions, confirming that all solutions achieve the required throughput.

6.3.2 Results

Figure 6.6 displays the latency for all five solutions. We can see that purely IEEE 802.11 has the lowest latency by far with around 14 ms, and LTE has the highest latency, around 65 ms. The difference between those two technologies is relatively high, with LTE having around four times the latency as IEEE 802.11. If we look at the latency with only TCP but reordering on the virtual layer, we can see that it reaches 56 ms after a startup phase, mainly due to LTE and its transmission scheme. Our improved solution can keep latency low in the build-up phase of LTE, useful for short connections, and achieves an average latency of 51 ms. This value is 10 % lower than simple reordering. Next, we compare MPTCP to our solution with similar weights as MPTCP does not allow to set weights and sets them on its own. For MPTCP, the weights are between 70 % and 80 %, leaning towards 80 %, in favor of IEEE 802.11. Our solution uses similar weights with 80 % IEEE 802.11 and 20 % LTE. TCP with reordering reaches 39 ms while MPTCP is slightly slower and reaches 40 ms. Our solution, on the other hand, reaches 37 ms and is faster than both of them. The reduction is lower than with equal weight distribution, but that is expected as each link has its own lower limit on possible achievable latency. Compared to MPTCP, our solution does not have any direct insight into TCP's current parameters, allowing MPTCP to adjust the flows on each link. Still, our solution outperforms it and gives full flexibility to centralized management intelligence.

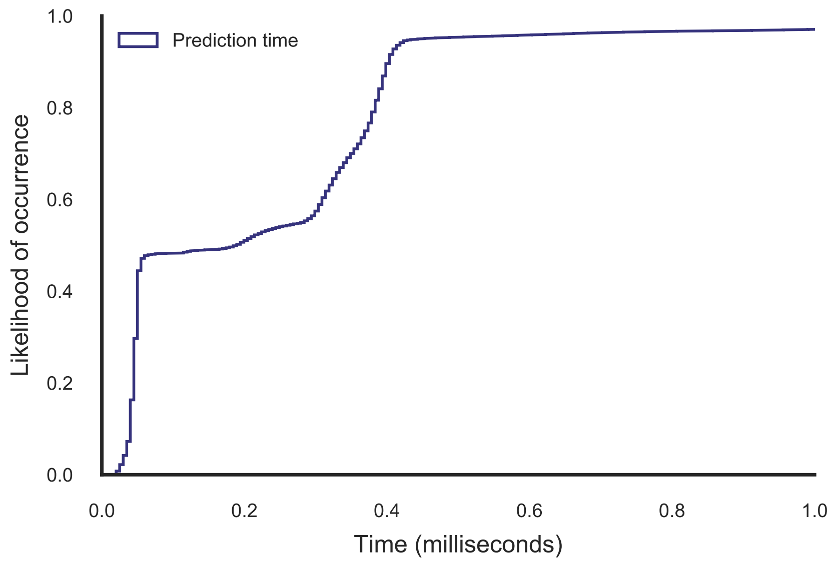


Figure 6.10: Cumulative distribution of prediction time in milliseconds, showing that 97 % of the predictions are below 1 ms. Processor interruptions and different prioritizations cause outliers above.

Figure 6.7 displays the latency probability distribution, using the Kernel Density Estimation (KDE) method for reordered TCP, MPTCP, and our solution. We can see that MPTCP has a high density between 25 ms and 50 ms due to its scheduler and weights for each link. TCP with reordering and our solution, both with equal weights, exhibit a similar density distribution. However, our solution shows a shift to lower latencies. Our solution can also normalize density spikes with lower latencies that can happen due to the reordering scheme and is present with reordered TCP and MPTCP alike. This behavior is still the case for a weight distribution equal to the one MPTCP uses, but less pronounced as physical limits of the links limit the optimization.

Figure 6.8, on the other hand, shows the density estimation of the arrival rate at the receiver using a KDE. If the arrival rate were completely normalized, a 6 Mbps stream would lead to an inter-packet arrival time of 1.8 ms. The reordered TCP's distribution can achieve the highest likelihood but gets closer to zero as many of the packets have no inter-arrival time between them. MPTCP can achieve a slightly better distribution, while our solution with equal weights offers even better inter-packet arrival than MPTCP, getting very close to the 1.8 ms mentioned above. Our solution can maintain that advantage with unequal weights, and the result being between MPTCP and equal weights.

Figure 6.9 gives an overview of the achieved throughput of each solution. As expected, all solutions can achieve 6 Mbps and do not have any dips or spikes due to

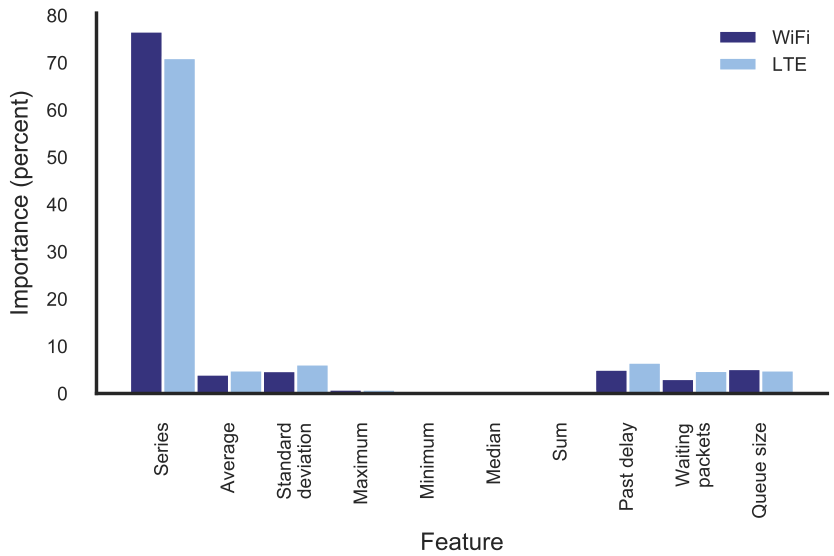


Figure 6.11: Feature importance by weight with time series features aggregated into one value.

TCP reacting on packet loss or overcompensating for lost throughput. This behavior is a validation that all solutions perform as intended.

As we stated earlier, our goal is to achieve a prediction time that is below 1 ms to have as little impact as possible on transmission. Figure 6.10 displays the cumulative distribution of the prediction time of our solution. As we can see, the vast majority (97 %) is below 1 ms, while around 50 % of the prediction times are even below 0.1 ms. This split is mainly due to the two models, where one can perform significantly faster than the other. There are very rare outliers that go above 1 ms, but the low amount does not impact the overall performance. Small load on the processor from a different application and different prioritization of the operating system's processes can cause these spikes. As we used a desktop operating system, background processes are running, but this can be further reduced for a production environment if necessary by reducing the number of background processes and focus only on essential ones. In an embedded environment, we can remove these outliers entirely as minimal background processes are running.

Figure 6.11 displays feature importance. For better readability, we aggregated all time series features into one value. The time series itself takes 71 % of the weight for LTE and 76 % for IEEE 802.11. The calculated features take around 13 % for LTE and 10 % for IEEE 802.11. The features representing the system's current state and past delay make up around 16 % for LTE and 14 % for IEEE 802.11. With its packet arrivals, the time series has by far the most substantial impact on the model. In contrast, aggregated features and features representing the current state have balanced but much lower importance. This behavior is the case for both technologies and their representative models. However, it also shows that catching the dynamic of TCP and its behavior requires enough information to make accurate predictions.

6.4 Conclusion

This chapter proposed a load balancing and reordering scheme for TCP to reduce latency and normalize the inter-packet arrival rate on links with different latency properties. We base our approach on packet arrival normalization set in the ORCHESTRA framework's grander scheme with a virtual MAC layer towards the network layer, predicting the future packet arrival rate through machine learning. We use a queue to store reordered packets that need to be transmitted and forward packets according to an artificial delay timer. With additional information about the system's current state, like the queue size and the packets waiting to be reordered, we can calculate an appropriate artificial delay. We can achieve lower latency than TCP with reordering and can even outperform MPTCP when we use similar weights. Our solution helps in centralized network planning by giving the flexibility to use multiple links while maintaining low latency.

7 Conclusions and Perspectives

This dissertation presented several contributions in IEEE 802.11 performance measurements and models and the area of multi-technology network orchestration and its application. The contributions include a wide range of performance measurements and models for the IEEE 802.11 system performance with external interferer. The proposed orchestration solutions enable advanced multi-technology features, such as handovers, load balancing, and duplication, and their optimization for varying link properties. This chapter summarizes how we addressed the challenges defined in Chapter 1 and verifies if we have addressed the hypothesis and research question adequately. Additionally, we provide an overview of new challenges and research topics for future researchers.

7.1 Review of problem statements

We addressed the challenges in Chapter 1 in the following way:

1. The behavior of IEEE 802.11 in large and dense environments with a non-IEEE 802.11 interfering source is unknown. A large number of stations can affect IEEE 802.11 performance negatively, even more so with an additional interfering source. Chapter 3 presents two performance measurements of IEEE 802.11 systems, one on a large event and one in a lab environment, and a basic model for latency in a system with an interfering source. The measurement at the massive event with a mesh network shows very high latency on each hop that can reach over 2 s, resulting in up to 10 s latency over several hops. This performance indicates the existence of an external source. We then evaluated the performance in a controlled environment

and confirmed an external source's significant performance impact. Based on these findings, we proposed a method of determining the latency of a system with an external source. We based the model on the performance of a system without such a source. The model achieves high accuracy, both in determining the saturation point as well as the maximum latency.

2. There is no model of IEEE 802.11 in environments with a non-IEEE 802.11 interfering source that can deliver a full explanation and description of the behavior of IEEE 802.11. As Chapter 3 showed, external sources can impact IEEE 802.11 performance and degrade the user experience. This chapter presented a basic model, which we extended in Chapter 4 with a fully analytical model that also allows predicting the behavior of an IEEE 802.11 system in various situations. Based on a Markov chain, the model includes a generic model for an interfering source to model different types of interferers. The analytical model fully covers all cases of how an interferer can affect an IEEE 802.11 system and, therefore, can serve as a basis to explain IEEE 802.11 behavior in different situations. We evaluated the model against real-life measurements and simulation, where it performed well compared to measurements. The simulations showed behavior differently from both measurement and model in some cases and seem to be not as reliable. We also demonstrated how one could use the model to explore different situations.

3. There is a lack of coordination between different technologies, which leads to inefficient use of the wireless spectrum and interference between technologies. The current design of communication technologies and their implementation of the OSI network stack prevents adequate coordination. This lack of coordination results in poor performance through connection interrupts during a handover or insufficient available bandwidth on the used technology. Chapter 5 presents the ORCHESTRA framework to alleviate the problem by introducing inter-technology management. The VMAC layer on devices abstracts available technologies and links and offers a single interface to higher layers. The central controller enables network management over multiple technologies with a global view and capabilities to manage devices with a virtual layer and legacy devices alike. These two components allow the framework to offer seamless inter-technology handovers, actual packet-level load balancing, and duplication. The ORCHESTRA framework is completely agnostic towards underlying technology through its abstraction scheme and higher layers like transport protocols and applications. This independence allows deploying the framework in various scenarios, ranging from LANs over edge or backhauling networks to satellite networks. We provided an in-depth evaluation with a real-life prototype, using LTE and IEEE 802.11 as technologies, and showed the features against the state-of-the-art solution MPTCP. Our solution performed similarly or better while allowing more fine-grained control in load balancing and duplication and broader transport protocol support. Network-wide intelligence and generic technology support also make it superior to other solutions like LTE-LWA or IEEE 1905.1.

4. To improve throughput, load balancing over multiple links is necessary, but sharing multiple links with different latency properties leads to reduced throughput in the most widely used transport protocol, TCP. Current protocols like MPTCP do not directly mitigate this issue but circumvent it by sending longer streams of packets over one link before switching. Chapter 6 presents a load balancing scheme that allows for full packet-level load balancing while minimizing differing links' latency effects. It utilizes machine learning to predict the future packet arrival rate and takes additional system parameters into account that affect TCP behavior. With its re-computation interval of 50 ms, it can adapt fast to changing environments. It achieves lower latency by normalizing the packet arrival rate and, therefore, a better packet arrival distribution leading to a smoother user experience while giving a central controller full flexibility over a flow's weight distribution.

7.2 Review of the hypothesis and research questions

We defined the following hypothesis in Chapter 1: Quantifying and modeling interference of wireless networks is necessary to mitigate it through cross-technology management with new mechanisms to fully support future networks and the increasing QoS requirements of users. This hypothesis encompasses measuring and modeling the behavior of wireless networks, specifically IEEE 802.11 networks, in an interferer's presence. Additionally, to elevate the issues with interference present, it describes cross-technology management's necessity to support QoS requirements for user applications. This dissertation presented several contributions to enable the envisioned solution. The first contribution in Chapter 3 includes performance measurements, highlighting the impact of external sources on IEEE 802.11 specifically, and provides an efficient model for deriving the expected latency. The second one is a fully analytical model in Chapter 4 that provides a full explanation of IEEE 802.11 performance behavior with an external source in various situations, allowing exploration of parameters and scenarios. The third is the ORCHESTRA framework in Chapter 5, which solves multi-technology management's challenge transparently and without changes to involved technologies. The fourth is a load balancing mechanism for links with different latency properties in Chapter 6, which reduces the latency, normalizes the packet arrival rate, and leads to more flexibility and reliability. The measurements and models show how dire the current situation is with interference between technologies and other types of interference sources. It also shows that in some situations, the only viable solution is not to use the technology or only in combination with another one. Of course, we can also use the models for network planning and proactive measures. The ORCHESTRA framework provides this functionality and is the basis for cross-technology management. It can use handovers to avoid interference, duplication to increase reliability, and load balancing for improved throughput. The machine learning assisted load balancing ensures that it is not affected by different link properties. Together, they offer a comprehensive solution to support current and future heterogeneous wireless networks while maintaining optimal user performance. Therefore, we can conclude that interference measurements and models in combination with inter-technology management and new mechanisms are indeed needed to support future heterogeneous wireless networks.

We answer the research questions of Section 1.4 as follows:

1. How does IEEE 802.11 behave in heterogeneous environments with a non-IEEE 802.11 interfering source? Moreover, can latency be estimated from the base performance when no such source is present? Chapter 3 provided the results of two measurement scenarios, one mesh network on a large event and one infrastructure setup in a lab environment. With latency as the primary QoS factor, we demonstrated that IEEE 802.11 systems are heavily affected by external interference. While an increasing number of participating devices can already heavily increase the latency due to the employed LBT protocol, an external source has a similar impact and further increases latency. Based on the observation that an external source partially affects IEEE 802.11 systems like a colossal packet, we provided a model that uses the base performance to calculate performance in a system with a source. The proposed model achieves high accuracy in case of interference and fast computation times as well. Using the system's base performance without interference has the advantage that it already includes medium access, and we do not need to model it explicitly.

2. Can IEEE 802.11 behavior and latency with a non-IEEE 802.11 interfering source be fully modeled, and does it allow for explanations? Chapter 4 derived a fully analytical model for IEEE 802.11 systems with an interfering source based on a Markov Chain. The model provides performance computations for throughput and latency in the saturated, meaning when the system capacity is already full, and the unsaturated case. The model takes advantage of a Markov chain of an IEEE 802.11 system with added transition probabilities for an interfering source to provide the full behavior of an IEEE 802.11 system. In the validation of the model, we also show that such a model performs better than simulation and is, therefore, more suitable for performance exploration in various scenarios.

3. How can one manage heterogeneous wireless networks and introduce solutions that allow for more functionality while still maintaining legacy compliance? Chapter 5 presented the ORCHESTRA framework, capable of abstracting underlying technology and offering a transparent solution to lower and higher layers. Most technologies, for example, IEEE 802.11 technologies, need to change at all and work seamlessly. Other technologies, such as 3GPP technologies, do not allow fine-grained packet-level control in their current state; they would only allow more coarse-grained control. Current and future releases of the 3GPP standard include new mechanisms, such as the LBO, combined with the MEC architecture that introduces opportunities to include the VMAC layer and take advantage of its full functionalities.

The advanced functionality, such as load balancing, handovers, and duplication, is entirely transparent to any higher layer through mechanisms like reordering and deduplication. Shifting the DHCP and ARP functionality from the network stack towards the virtual layer allows complete transparency towards the network and enables full legacy compliance. On the other hand, the ORCHESTRA controller manages all available devices and provides a central platform to perform network intelligence. It aggregates monitoring information and enables a global view. Additionally, it can work with devices without a VMAC layer to some extent to manage them.

4. Can a mechanism be defined that allows for load balancing over several links with different latency properties while maintaining or increasing throughput with TCP? In Chapter 6, we provide a mechanism that normalizes the packet arrival rate on the receiver. The receiver or an intermediate node does not forward packets in a burst after waiting for a missing packet to be reordered. Instead, the node forwards packets with a small artificial delay, which normalizes the packet arrival rate and the inter-packet time. The chosen machine learning approach for predicting the future packet arrival rate ensures that the computation time stays below 1 ms so that computation every 50 ms is feasible. The proposed mechanism efficiently manages links with different latency and allows more flexibility for a management solution.

7.3 Future perspectives

This dissertation proposed solutions for performance modeling in dense and heterogeneous IEEE 802.11 systems affected by external influence and solutions to orchestrate heterogeneous wireless networks with multiple technologies present. Those topics are ever-evolving, by new technologies, the evolution of standards, and new challenges through user demand. Following, we list areas of possible future research, corresponding to each contribution. Note that we combine the topics related to Chapter 3 and Chapter 4 as the latter builds upon the first.

7.3.1 Performance studies and models in IEEE 802.11 systems with interfering source

- More extensive and controlled measurement studies with different sources of interference. Measurement studies give significant insight into the behavior and challenges of IEEE 802.11 systems. As we have seen, simulation can not fully duplicate the behavior of IEEE 802.11 with different types of interfering sources. Therefore, additional measurement studies can help in understanding and characterizing different types of interference. Those studies might also help discover the behavior of other technologies, which can lead to better models for those technologies.

- Extending the model to newer and future standards. Our current models take the IEEE 802.11a standard as a basis. Since its presentation, the IEEE association presented a newer and more advanced standard and will do so in the future. These newer standards include different new features that improve performance, like block acknowledgments and others but are not covered by our model. One can extend our model to include those and future features. Additionally, one can adapt our model to fit technologies that work in a different spectrum like the IEEE 802.11ad standard with 60 GHz. This spectrum has different properties and possibly behaves differently on an interfering source.
- Combine the advantages of fast computation and full behavior modeling. In this dissertation, we provided two models, one that allows for fast computation and one that fully models the system's behavior. The first one is currently limited to latency only and requires a base performance, while the latter includes throughput as well but requires significantly more time to compute. Combining the advantages of both models, fast computation, and better representation can be a research task. Especially for real-time network management, such a model is necessary as the computation time can not be too long. Otherwise, the information is already outdated.
- Provide models for new technologies. This dissertation's focus was the IEEE 802.11 technology, primarily because it uses an LBT protocol that can be impacted much more severely than a centrally scheduled protocol, such as LTE. Future research can include models with an external interferer for other technologies, and computing the expected performance of a link in a particular scenario and environment is incredibly vital for optimizing the network's performance. While many models for different technologies describing the performance already exist, most do not include a non-communication protocol interferer. If models consider external interference, then it is mostly in the form of another communication technology sharing the same spectrum.

7.3.2 Managing heterogeneous wireless networks

- Develop management and load balancing algorithms that constitute the central intelligence. The ORCHESTRA framework provides the tools to handle arbitrary technologies on a device by introducing the VMAC layer. It offers a central location for management with a global view of the network. To effectively manage an extensive wireless network, one needs an algorithm that takes all the available information into account, determines an optimal or near-optimal solution, and then uses the available tools the framework offers to achieve the solution. For example, an algorithm could continuously move clients from wireless endpoints, such as APs or eNBs, to avoid overloading such an endpoint. A more proactive approach could predict each device's future QoS requirements and assign technologies and endpoints to each device to fulfill the requirements.

- Enable the use of uncontrolled networks. As we presented it, the ORCHESTRA framework assumes that all technologies and networks are under its control. We see more often that users take advantage of a mix of networks operated by different network operators. For example, one operator operates the mobile network, and the user or another operator, the IEEE 802.11 network. Features, such as handovers, load balancing, or duplication, only work limited or not at all in such a scenario. Some commercial SDN solutions use a cloud-based solution to establish a tunnel from each technology to the cloud instance and forward traffic from there. ORCHESTRA could realize something similar without the need for tunnels through the use of so-called hole punching to circumvent firewalls and Network Address Translation (NAT) to establish a connection with devices that use technologies not controlled by ORCHESTRA [201]. This option would also work with legacy mobile networks that do not employ an LBT and MEC architecture.

- Provide packet scheduling across technologies. The ORCHESTRA framework, with its VMAC, allows for packet-level features such as load balancing and duplication. Currently, the virtual layer splits the packets or copies them and then directly forwards them to the underlying technology, letting it take care of the transmission. Instead, the VMAC could use a scheduling scheme on top of each technology or even a combination of technologies to optimize the transmission window while still letting the underlying technology and their medium access handle the transmission. Through the overarching scheduling scheme, the VMAC can reduce the interference between technologies and devices and improve the overall performance.

- Improve performance through kernel-level implementation. The current prototype of the VMAC in Section 5.5 uses the Click modular router for its implementation. While this implementation method allows for fast prototyping and is heavily optimized, it runs in user space, which can not compete with a kernel space implementation. A kernel space implementation is advisable for deployment on embedded devices and a more detailed study of the performance impact of the proposed features on memory and computing resources.

7.3.3 Machine Learning based load balancing over links with different latency

- Extending supported technologies. Currently, the prototype presented in Section 6.3.1 only supports LTE and IEEE 802.11 as technologies. As those two technologies have a large difference in latency, they were the obvious choice for demonstrating the capabilities. One straightforward future research path is the inclusion of additional technologies. There is a wide variety of technologies that the ORCHESTRA framework supports and, therefore, the models and packet arrival prediction should be available for these technologies. This dissertation's focus was on high bandwidth and low latency technologies, but future research should also cover low bandwidth and high latency solutions.

- Increasing accuracy of packet arrival prediction. The load balancing mechanism's current focus is re-computation speed above accuracy and correcting wrong predictions as fast as possible instead of purely focusing on accuracy. We only achieved an interval of 50 ms and a prediction time of 1 ms due to this focus. Improving the accuracy of the packet arrival prediction might be possible without sacrificing the computation speed and is one possible way to improve the performance of the system.

- Using deep learning for more extensive behavior capturing. The choice of machine learning framework in this dissertation was Xgboost as it offers fast computation time, high reliability, and was easily compatible with our VMAC implementation. With current and future development in deep learning, small and fast performing neural networks for predicting the packet arrival rate becomes a possibility. Depending on the task, neural networks might achieve higher accuracy as they can better represent TCP's behavior in this case.

References

[1] D. Corujo, S. Sargento, L.J.G. Villalba, F. Buiati, and R.L. Aguiar. IEEE 802.21 Information Services deployment for heterogeneous mobile environments. *IET Communications*, 5(18):2721–2729, 2011.

[2] IEEE Communications Society. IEEE Standard for a Convergent Digital Home Network for Heterogeneous Technologies. Technical report, IEEE Standards Association, 2013.

[3] Roberto Riggio, Mahesh K. Marina, Julius Schulz-Zander, Slawomir Kuklinski, and Tinku Rasheed. Programming Abstractions for Software-Defined Wireless Networks. *IEEE Transactions on Network and Service Management*, 12(2):146–162, 2015.

[4] Daniela Laselva, David Lopez-Perez, Mika Rinne, and Tero Henttonen. 3GPP LTE-WLAN Aggregation Technologies: Functionalities and Performance Comparison. *IEEE Communications Magazine*, 56(3):195–203, 2018.

[5] Christoph Paasch, Simone Ferlin, Ozgu Alay, and Olivier Bonaventure. Experimental evaluation of multipath TCP schedulers. In *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop - CSWS '14*, pages 27–32. ACM Press, 2014.

[6] Stephan Reiff-Marganiec and Kenneth J. Turner. Use of Logic to Describe Enhanced Communications Services. In *Proceedings of the 22Nd IFIP WG 6.1 International Conference Houston on Formal Techniques for Networked and Distributed Systems*, number November 2002 in Lecture Notes in Computer Science, pages 130–145. Springer-Verlag, 2002.

[7] Peter P. Pham. Comprehensive analysis of the IEEE 802.11. *Mobile Networks and Applications*, 10(5):691–703, 2005.

[8] Rama Krishna Challa, Saswat Chakrabarti, and Debasish Datta. An Improved Analytical Model for IEEE 802.11 Distributed Coordination Function under Finite Load. *International Journal of Communications, Network and System Sciences*, 02(03):237–247, 2009.

[9] Cisco. The Internet of Things - How the Next Evolution of the Internet is Changing Everything, 2011. URL https://www.cisco.com/c/dam/en_us/

about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf. Accessed on 2020-02-10.

[10] Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Larry G. Roberts, and Stephen Wolff. A brief history of the internet. *ACM SIGCOMM Computer Communication Review*, 39(5):22, 2009.

[11] Statista. Forecast number of mobile devices worldwide from 2019 to 2023, 2019. URL https://www.statista.com/statistics/245501/multiple-mobile-device-ownership-worldwide/. Accessed on 2020-02-10.

[12] Statista. Installed base of personal computers (PCs) worldwide from 2013 to 2019, 2016. URL https://www.statista.com/statistics/610271/worldwide-personal-computers-installed-base/. Accessed on 2020-02-10.

[13] Cisco. Cisco Visual Networking Index: Forecast and Trends, 20172022, 2017. URL http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html. Accessed on 2020-02-10.

[14] IoT Analytics. State of the IoT 2018: Number of IoT devices now at 7B Market accelerating, 2018. URL https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/. Accessed on 2020-02-10.

[15] Statista. Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025, 2016. URL https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. Accessed on 2020-02-10.

[16] M. Shahwaiz Afaqui, Eduard Garcia-Villegas, and Elena Lopez-Aguilera. IEEE 802.11ax: Challenges and Requirements for Future High Efficiency WiFi. *IEEE Wireless Communications*, 24(3):130–137, 2017.

[17] Boris Bellalta. IEEE 802.11ax: High-efficiency WLANS. *IEEE Wireless Communications*, 23(1):38–46, 2016.

[18] Ian F. Akyildiz, David M. Gutierrez-Estevez, and Elias Chavarria Reyes. The evolution to 4G cellular systems: LTE-Advanced. *Physical Communication*, 3(4):217–244, 2010.

[19] Shao Yu Lien, Shin Lin Shieh, Yenming Huang, Borching Su, Yung Lin Hsu, and Hung Yu Wei. 5G New Radio: Waveform, Frame Structure, Multiple Access, and Initial Access. *IEEE Communications Magazine*, 55(6):64–71, 2017.

[20] Erik Dahlman and Stefan Parkvall. NR - The New 5G Radio-Access Technology. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, volume 2018-June, pages 1–6, 2018.

[21] Aloÿs Augustin, Jiazi Yi, Thomas Clausen, and William Townsley. A Study of LoRa: Long Range & Low Power Networks for the Internet of Things. *Sensors*, 16(9):1466, 2016.

[22] Kuor-Hsin Chang and Bob Mason. The IEEE 802.15.4g standard for smart metering utility networks. In *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, pages 476–480, 2012.

[23] Victor Baños-Gonzalez, M. Afaqui, Elena Lopez-Aguilera, and Eduard Garcia-Villegas. IEEE 802.11ah: A Technology to Face the IoT Challenge. *Sensors*, 16(11):1960, 2016.

[24] Shadi Al-Sarawi, Mohammed Anbar, Kamal Alieyan, and Mahmood Alzubaidi. Internet of Things (IoT) communication protocols: Review. In *2017 8th International Conference on Information Technology (ICIT)*, pages 685–690, 2017.

[25] Brecht Reynders, Wannes Meert, and Sofie Pollin. Range and coexistence analysis of long range unlicensed communication. In *2016 23rd International Conference on Telecommunications (ICT)*, pages 1–6, 2016.

[26] Usman Raza, Parag Kulkarni, and Mahesh Sooriyabandara. Low Power Wide Area Networks: An Overview. *IEEE Communications Surveys & Tutorials*, 19(2):855–873, 2017.

[27] Thomas Nitsche, Carlos Cordeiro, Adriana Flores, Edward Knightly, Eldad Perahia, and Joerg Widmer. IEEE 802.11ad: directional 60 GHz communication for multi-Gigabit-per-second Wi-Fi [Invited Paper]. *IEEE Communications Magazine*, 52(12):132–141, 2014.

[28] Yasaman Ghasempour, Claudio R. C. M. da Silva, Carlos Cordeiro, and Edward W. Knightly. IEEE 802.11ay: Next-Generation 60 GHz Communication for 100 Gb/s Wi-Fi. *IEEE Communications Magazine*, 55(12): 186–192, 2017.

[29] Harald Haas, Liang Yin, Yunlu Wang, and Cheng Chen. What is LiFi? *Journal of Lightwave Technology*, 34(6):1533–1544, 2016.

[30] Shravan Rayanchu, Ashish Patro, and Suman Banerjee. Airshark: Detecting Non-WiFi RF Devices using Commodity WiFi Hardware. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference - IMC '11*, page 137, 2011.

[31] Shyamnath Gollakota, Fadel Adib, Dina Katabi, and Srinivasan Seshan. Clearing the RF Smog: Making 802.11 Robust to Cross-Technology Interference. In *Proceedings of the ACM SIGCOMM 2011 conference on SIGCOMM - SIGCOMM '11*, page 170, 2011.

[32] Jiun Ren Lin, Timothy Talty, and Ozan K. Tonguz. An empirical performance study of Intra-vehicular Wireless Sensor Networks under WiFi and Bluetooth interference. In *GLOBECOM - IEEE Global Telecommunications Conference*, pages 581–586, 2013.

[33] Fuad M. Abinader, Erika P.L. Almeida, Fabiano S. Chaves, Andre M. Cavalcante, Robson D. Vieira, Rafael C.D. Paiva, Angilberto M. Sobrinho, Sayantan Choudhury, Esa Tuomaala, Klaus Doppler, and Vicente A. Sousa. Enabling the coexistence of LTE and Wi-Fi in unlicensed bands. *IEEE Communications Magazine*, 52(11):54–61, 2014.

[34] Andre M. Cavalcante, Erika Almeida, Robson D. Vieira, Sayantan Choudhury, Esa Tuomaala, Klaus Doppler, Fabiano Chaves, Rafael C. D. Paiva, and Fuad Abinader. Performance Evaluation of LTE and Wi-Fi Coexistence in Unlicensed Bands. In *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, pages 1–6, 2013.

[35] Hubert Zimmermann. OSI Reference Model–The ISO Model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications*, 28(4):425–432, 1980.

[36] Anitha Varghese and Deepaknath Tandur. Wireless requirements and challenges in Industry 4.0. In *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, pages 634–638, 2014.

[37] Amitabha Ghosh, Rittwik Jana, V. Ramaswami, Jim Rowland, and N. K. Shankaranarayanan. Modeling and characterization of large-scale Wi-Fi traffic in public hot-spots. In *Proceedings - IEEE INFOCOM*, pages 2921–2929, 2011.

[38] Thomas K. Paul and Tokunbo Ogunfunmi. Wireless LAN comes of age: Understanding the IEEE 802.11n amendment. *IEEE Circuits and Systems Magazine*, 8(1):28–54, 2008.

[39] Jiansong Zhang, Kun Tan, Jun Zhao, Haitao Wu, and Yongguang Zhang. A practical SNR-guided rate adaptation. In *Proceedings - IEEE INFOCOM*, pages 146–150, 2008.

[40] Kishore Ramachandran, Haris Kremo, Marco Gruteser, Predrag Spasojević, and Ivan Šeškar. Scalability analysis of rate adaptation techniques in congested IEEE 802.11 networks: An ORBIT testbed comparative study. In *2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WOWMOM*, pages 1–12, 2007.

[41] Ramakrishna Gummadi, David Wetherall, Ben Greenstein, and Srinivasan Seshan. Understanding and mitigating the impact of RF interference on 802.11 networks. *ACM SIGCOMM Computer Communication Review*, 37 (4):385, 2007.

[42] Mark A. McHenry, Peter A. Tenhula, Dan McCloskey, Dennis A. Roberson, and Cynthia S. Hood. Chicago spectrum occupancy measurements & analysis and a long-term studies proposal. In *Proceedings of the first international workshop on Technology and policy for accessing spectrum - TAPAS '06*, pages 1–es, 2006.

[43] Sanjit Biswas, John Bicket, Edmund Wong, Raluca Musaloiu-E, Apurv Bhartia, and Dan Aguayo. Large-scale Measurements of Wireless Network Behavior. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication - SIGCOMM '15*, pages 153–165, 2015.

[44] Rutvij H. Jhaveri and Narendra M. Patel. Mobile Ad-hoc Networking with AODV : A Review. *International Journal of Next-Generation Computing*, 6 (3):165–191, 2015.

[45] Jakob Eriksson, Sharad Agarwal, Paramvir Bahl, and Jitendra Padhye. Feasibility study of mesh networks for all-wireless offices. In *Proceedings of the 4th international conference on Mobile systems, applications and services - MobiSys 2006*, page 69, 2006.

[46] Llorenç Cerdà-Alabern, Axel Neumann, and Pau Escrich. Experimental evaluation of a wireless community mesh network. In *Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems - MSWiM '13*, pages 23–30, 2013.

[47] Serdar Vural, Dali Wei, and Klaus Moessner. Survey of experimental evaluation studies for wireless mesh network deployments in urban areas towards ubiquitous internet. *IEEE Communications Surveys and Tutorials*, 15(1): 223–239, 2013.

[48] Antonios Michaloliakos, Ryan Rogalin, Yonglong Zhang, Konstantinos Psounis, and Giuseppe Caire. Performance modeling of next-generation WiFi networks. *Computer Networks*, 105:150–165, 2016.

[49] Fuad M. Abinader, Erika P.L. Almeida, Sayantan Choudhury, Vicente A. Sousa, Andre M. Cavalcante, Fabiano S. Chaves, Esa Tuomaala, Robson D. Vieira, and Klaus Doppler. Performance evaluation of IEEE 802.11n WLAN in dense deployment scenarios. In *IEEE Vehicular Technology Conference*, pages 1–5, 2014.

[50] Xueheng Hu, Lixing Song, Dirk Van Bruggen, and Aaron Striegel. Is There WiFi Yet? How Aggressive WiFi Probe Requests Deteriorate Energy and Throughput. In *Proceedings of the 2015 ACM Conference on Internet Measurement Conference - IMC '15*, pages 317–323, 2015.

[51] Utpal Paul, Anand Kashyap, Ritesh Maheshwari, and Samir R. Das. Passive Measurement of Interference in WiFi Networks with Application in Misbehavior Detection. *IEEE Transactions on Mobile Computing*, 12(3):434–446, 2013.

[52] Paul Fuxjager, Danilo Valerio, and Fabio Ricciato. The myth of non-overlapping channels: interference measurements in IEEE 802.11. In *2007 Fourth Annual Conference on Wireless on Demand Network Systems and Services*, pages 1–8, 2007.

[53] Eduard Garcia Villegas, Elena López-Aguilera, Rafael Vidal, and Josep Paradells. Effect of adjacent-channel interference in IEEE 802.11 WLANs. In *Proceedings of the 2nd International Conference on Cognitive Radio Oriented Wireless Networks and Communications, CrownCom*, pages 118–125, 2007.

[54] Marco Cardenas-Juarez, Miguel A. Diaz-Ibarra, Ulises Pineda-Rico, Armando Arce, and Enrique Stevens-Navarro. On spectrum occupancy measurements at 2.4 GHz ISM band for cognitive radio applications. In *2016 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, pages 25–31, 2016.

[55] Peizhong Yi, Abiodun Iwayemi, and Chi Zhou. Developing ZigBee Deployment Guideline Under WiFi Interference for Smart Grid Applications. *IEEE Transactions on Smart Grid*, 2(1):110–120, 2011.

[56] Giuseppe Bianchi. IEEE 802.11-saturation throughput analysis. *IEEE Communications Letters*, 2(12):318–320, 1998.

[57] Giuseppe Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, 2000.

[58] Haitao Wu, Yong Peng, Keping Long, Shiduan Cheng, and Jian Ma. Performance of reliable transport protocol over IEEE 802.11 wireless LAN: analysis and enhancement. In *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 599–607, 2002.

[59] P. Chatzimisios, A.C. Boucouvalas, and V. Vitsas. IEEE 802.11 Packet Delay - A Finite Retry Limit Analysis. In *GLOBECOM '03. IEEE Global Telecommunications Conference (IEEE Cat. No.03CH37489)*, volume 2, pages 950–954, 2003.

[60] P. Chatzimisios, A.C. Boucouvalas, and V Vitsas. Influence of channel BER on IEEE 802.11 DCF. *Electronics Letters*, 39(23):1687, 2003.

[61] I.N. Vukovic and N. Smavatkul. Delay analysis of different backoff algorithms in IEEE 802.11. In *IEEE 60th Vehicular Technology Conference, 2004. VTC2004-Fall. 2004*, volume 6, pages 4553–4557, 2004.

[62] P. Chatzimisios, A. C. Boucouvalas, and V. Vitsas. Performance analysis of the IEEE 802.11 MAC protocol for wireless LANs. *International Journal of Communication Systems*, 18(6):545–569, 2005.

[63] Taka Sakurai and Hai L. Vu. MAC access delay of IEEE 802.11 DCF. *IEEE Transactions on Wireless Communications*, 6(5):1702–1710, 2007.

[64] Yun Li, Chonggang Wang, Keping Long, and Weiliang Zhao. Modeling channel access delay and jitter of IEEE 802.11 DCF. *Wireless Personal Communications*, 47(3):417–440, 2008.

[65] P. Raptis, V. Vitsas, and K. Paparrizos. Packet delay metrics for IEEE 802.11 distributed coordination function. *Mobile Networks and Applications*, 14(6): 772–781, 2009.

[66] N.A. Lei, Ting Zhang, Liang Zhou, Xiaoqin Song, and Shengsuo Cai. Saturation throughput analysis of IEEE 802.11 DCF with heterogeneous node transmit powers and capture effect. *International Journal of Ad Hoc and Ubiquitous Computing*, 26(1):1, 2017.

[67] Omesh Tickoo and O. Sikdar. Modeling queueing and channel access delay in unsaturated IEEE 802.11 random access MAC based wireless networks. *IEEE/ACM Transactions on Networking*, 16(4):878–891, 2008.

[68] F. Daneshgaran, M. Laddomada, F. Mesiti, and M. Mondin. Unsaturated throughput analysis of IEEE 802.11 in presence of non ideal transmission channel and capture effects. *IEEE Transactions on Wireless Communications*, 7(4):1276–1286, 2008.

[69] Ash Mohammad Abbas and Khaled Abdullah Mohammed Al Soufy. A queue state driven analysis of IEEE 802.11 DCF for ad hoc networks under non-saturation conditions. *International Journal of Computational Science and Engineering*, 12(2/3):237, 2016.

[70] Emad Felemban and Eylem Ekici. Single Hop IEEE 802.11 DCF Analysis Revisited: Accurate Modeling of Channel Access Delay and Throughput for Saturated and Unsaturated Traffic Cases. *IEEE Transactions on Wireless Communications*, 10(10):3256–3266, 2011.

[71] Guosong Tian and Yu-Chu Tian. Markov Modelling of the IEEE 802.11 DCF for Real-Time Applications with Periodic Traffic. In *2010 IEEE 12th International Conference on High Performance Computing and Communications (HPCC)*, pages 419–426, 2010.

[72] Changchun Xu, Kezhong Liu, Gan Liu, and Jianhua He. Accurate Queuing Analysis of IEEE 802.11 MAC Layer. In *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, pages 1–5, 2008.

[73] Yaw-Wen Kuo, Wei-Fu Lu, and Tung-Lin Tsai. A framework to approximate the delay distribution for IEEE 802.11 DCF protocol. In *2009 IEEE 9th Malaysia International Conference on Communications (MICC)*, pages 874–879, 2009.

[74] Liying Xie, Hongjiang Wang, Gang Wei, and Zaijin Xie. Performance Analysis of IEEE 802.11 DCF in Multi-hop Ad Hoc Networks. In *2009 International Conference on Networks Security, Wireless Communications and Trusted Computing*, pages 227–230, 2009.

[75] Morteza Mehrnoush, Vanlin Sathya, Sumit Roy, and Monisha Ghosh. Analytical Modeling of Wi-Fi and LTE-LAA Coexistence: Throughput and Impact of Energy Detection Threshold. *IEEE/ACM Transactions on Networking*, 26 (4):1990–2003, 2018.

[76] Przemysław Machań and Jozef Wozniak. On the fast BSS transition algorithms in the IEEE 802.11r local area wireless networks. *Telecommunication Systems*, 52(4):2713–2720, 2013.

[77] Peng Li, Yan Pan, and Xiao Xin Yi. A seamless handover mechanism for IEEE 802.16e systems. *International Conference on Communication Technology Proceedings, ICCT*, pages 1–4, 2006.

[78] Hossam Fattah and Hussein Alnuweiri. A new handover mechanism for IEEE 802.16e wireless networks. *IWCMC 2008 - International Wireless Communications and Mobile Computing Conference*, pages 661–665, 2008.

[79] Jyotsna Agrawal, P. Mor, J.M. Keller, Rakesh Patel, and P. Dubey. Introduction to the basic LTE handover procedures. *2015 International Conference on Communication Networks (ICCN)*, pages 197–201, 2016.

[80] Rami Ahmad, Elankovan A. Sundararajan, Nor E. Othman, and Mahamod Ismail. Handover in LTE-advanced wireless networks: state of art and survey of decision algorithm. *Telecommunication Systems*, 66(3):533–558, 2017.

[81] George Lampropoulos, A.K. Salkintzis, and Nikos Passas. Media-independent handover for seamless service provision in heterogeneous networks. *IEEE Communications Magazine*, 46(1):64–71, 2008.

[82] Kenichi Taniuchi, Yoshihiro Ohba, Victor Fajardo, Subir Das, Miriam Tauil, Yuu-Heng Cheng, Ashutosh Dutta, Donald Baker, Maya Yajnik, and David Famolari. IEEE 802.21: Media independent handover: Features, applicability, and realization. *IEEE Communications Magazine*, 47(1):112–120, 2009.

[83] A. De La Oliva, A. Banchs, I. Soto, T. Melia, and A. Vidal. An overview of IEEE 802.21: media-independent handover services. *IEEE Wireless Communications*, 15(4):96–103, 2008.

[84] Meng Shu Chiang, Chung Ming Huang, Duy Tuan Dao, and Binh Chau Pham. The Backward Fast Media Independent Handover for Proxy Mobile IPv6 Control Scheme (BFMIH-PMIPV6) over Heterogeneous Wireless Mobile Networks*. *Journal of Information Science and Engineering*, 34(3):765–780, 2018.

[85] IEEE Communications Society. IEEE 802.21-2017 - IEEE Standard for Local and metropolitan area networks–Part 21: Media Independent Services Framework. Technical report, IEEE Standards Association, 2017.

[86] Vishal Sharma, Jiyoon Kim, Soonhyun Kwon, Ilsun You, and Fang-Yie Leu. An Overview of 802.21a-2012 and Its Incorporation into IoT-Fog Networks Using Osmotic Framework. In *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, pages 64–72. Springer International Publishing, 2018.

[87] Behrouz Shahgholi Ghahfarokhi and Naser Movahhedinia. A survey on applications of IEEE 802.21 Media Independent Handover framework in next generation wireless networks. *Computer Communications*, 36(10-11): 1101–1119, 2013.

[88] Chih-Peng Lin, Hsing-Lung Chen, and Jenq-Shiou Leu. A predictive handover scheme to improve service quality in the IEEE 802.21 network. *Computers & Electrical Engineering*, 38(3):681–693, 2012.

[89] Bathich Ammar A., Dani Baba Mohd, and Ibrahim Muhammad. IEEE 802.21 based vertical handover in WiFi and WiMAX networks. In *2012 IEEE Symposium on Computers & Informatics (ISCI)*, pages 140–144, 2012.

[90] Donato Macone, Guido Oddi, Andi Palo, and Vincenzo Suraci. A dynamic load balancing algorithm for Quality of Service and mobility management in next generation home networks. *Telecommunication Systems*, 53(3):265–283, 2013.

[91] Mangal Sain, Young Jin Kang, and Hoon Jae Lee. Survey on security in Internet of Things: State of the art and challenges. In *2017 19th International Conference on Advanced Communication Technology (ICACT)*, pages 699–704, 2017.

[92] Andy Crabtree, Richard Mortier, Tom Rodden, and Peter Tolmie. Unremarkable Networking: The Home Network as a Part of Everyday Life. In *Proceedings of the Designing Interactive Systems Conference on - DIS '12*, page 554, 2012.

[93] Bouchet, Javaudin, Kortebi, El Adbellaouy, Brzozowski, Katsianis, Mayer, Guan, Lebouc, Fontaine, Cochet, Jaffre, Mengi, Celeda, Gundogdu Aytekin, and Kurt. ACEMIND: The Smart Integrated Home Network. In *2014 International Conference on Intelligent Environments*, pages 1–8, 2014.

[94] Pierluigi Gallo, Katarzyna Kosek-Szott, Szymon Szott, and Ilenia Tinnirello. SDN@home: A method for controlling future wireless home networks. *IEEE Communications Magazine*, 54(5):123–131, 2016.

[95] Ke Xu, Xiaoliang Wang, Wei Wei, Houbing Song, and Bo Mao. Toward software defined smart home. *IEEE Communications Magazine*, 54(5): 116–122, 2016.

[96] Giuseppe Bianchi, Pierluigi Gallo, Domenico Garlisi, Fabrizio Giuliano, Francesco Gringoli, and Ilenia Tinnirello. MAClets: Active MAC Protocols over Hard-Coded Devices. In *Proceedings of the 8th international conference on Emerging networking experiments and technologies - CoNEXT '12*, page 229, 2012.

[97] Lalith Suresh, Julius Schulz-Zander, Ruben Merz, Anja Feldmann, and Teresa Vazao. Towards programmable enterprise WLANS with Odin. In *Proceedings of the first workshop on Hot topics in software defined networks - HotSDN '12*, page 115, 2012.

[98] Luis Sequeira, Juan Luis de la Cruz, Jose Ruiz-Mas, Jose Saldana, Julian Fernandez-Navajas, and Jose Almodovar. Building an SDN Enterprise WLAN Based on Virtual APs. *IEEE Communications Letters*, 21(2):374–377, 2017.

[99] Behnam Dezfouli, Vahid Esmaeelzadeh, Jaykumar Sheth, and Marjan Radi. A Review of Software-Defined WLANs: Architectures and Central Control Mechanisms. *IEEE Communications Surveys & Tutorials*, 21(1):431–463, 2019.

[100] Estefania Coronado, Shah Nawaz Khan, and Roberto Riggio. 5G-EmPOWER : A Software-Defined Networking Platform for 5G Radio Access Networks. *IEEE Transactions on Network and Service Management*, 16(2): 715–728, 2019.

[101] Wi-5. What to do With the Wi-Fi Wild West? A proposal for Wi-Fi prosumer networking, 2016. URL https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5bab1f966&appId=PPGMS.

[102] Jose Saldana, David De Hoz, Julián Fernández-Navajas, José Ruiz-Mas, Fernando Pascual, Diego R. Lopez, David Florez, Juan A. Castell, and Manuel Nuñez. Small-Packet Flows in Software Defined Networks: Traffic Profile Optimization. *Journal of Networks*, 10(4):176–187, 2015.

[103] Jose Saldana, Jose Ruiz-Mas, and Jose Almodovar. Frame Aggregation in Central Controlled 802.11 WLANs: The Latency Versus Throughput Tradeoff. *IEEE Communications Letters*, 21(11):2500–2503, 2017.

[104] Jose Saldana, Ruben Munilla, Salim Eryigit, Omer Topal, Jose Ruiz-Mas, Julian Fernandez-Navajas, and Luis Sequeira. Unsticking the Wi-Fi Client: Smarter Decisions Using a Software Defined Wireless Solution. *IEEE Access*, 6:30917–30931, 2018.

[105] David Astely, Erik Dahlman, Gabor Fodor, Stefan Parkvall, and Joachim Sachs. LTE release 12 and beyond. *IEEE Communications Magazine*, 51(7): 154–160, 2013.

[106] Rapeepat Ratasuk, Nitin Mangalvedhe, and Amitava Ghosh. LTE in unlicensed spectrum using licensed-assisted access. In *2014 IEEE Globecom Workshops (GC Wkshps)*, pages 746–751, 2014.

[107] Nadisanka Rupasinghe and Ismail Guvenc. Licensed-assisted access for WiFi-LTE coexistence in the unlicensed spectrum. In *2014 IEEE Globecom Workshops (GC Wkshps)*, pages 894–899, 2014.

[108] Amitav Mukherjee, Jung-Fu Cheng, Sorour Falahati, Laetitia Falconetti, Anders Furuskar, Bruhtesfa Godana, Du Ho Kang, Havish Koorapaty, Daniel Larsson, and Yu Yang. System architecture and coexistence evaluation of licensed-assisted access LTE with IEEE 802.11. In *2015 IEEE International Conference on Communication Workshop (ICCW)*, pages 2350–2355, 2015.

[109] Ran Zhang, Miao Wang, Lin X. Cai, Zhongming Zheng, Xuemin Shen, and Liang-Liang Xie. LTE-unlicensed: the future of spectrum aggregation for cellular networks. *IEEE Wireless Communications*, 22(3):150–159, 2015.

[110] Xuyu Wang, Shiwen Mao, and Michelle X. Gong. A Survey of Lte Wi-Fi Coexistence in Unlicensed Bands. *GetMobile: Mobile Computing and Communications*, 20(3):17–23, 2017.

[111] Amitav Mukherjee, Jung-Fu Fu Cheng, Sorour Falahati, Havish Koorapaty, Du Ho Kang, Reem Karaki, Laetitia Falconetti, and Daniel Larsson. Licensed-Assisted Access LTE: coexistence with IEEE 802.11 and the evolution toward 5G. *IEEE Communications Magazine*, 54(6):50–57, 2016.

[112] Shweta Sagari, Samuel Baysting, Dola Saha, Ivan Seskar, Wade Trappe, and DIpankar Raychaudhuri. Coordinated dynamic spectrum management of LTE-U and Wi-Fi networks. In *2015 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pages 209–220, 2015.

[113] Yue Wu, Weisi Guo, Hu Yuan, Long Li, Siyi Wang, Xiaoli Chu, and Jie Zhang. Device-to-device meets LTE-unlicensed. *IEEE Communications Magazine*, 54(5):154–159, 2016.

[114] Christian Hoymann, David Astely, Magnus Stattin, Gustav Wikstrom, Jung-Fu Cheng, Andreas Hoglund, Mattias Frenne, Ricardo Blasco, Joerg Huschke, and Fredrik Gunnarsson. LTE release 14 outlook. *IEEE Communications Magazine*, 54(6):44–49, 2016.

[115] Global mobile Suppliers Association (GSA). LTE in Unlicensed Spectrum: Trials, Deployments and Devices, 2018. URL https://gsacom.com/download.php?id=5699. Accessed on 2020-02-10.

[116] David Chambers. MulteFire lights up the path for universal wireless service, 2016.

[117] Claudio Rosa, Markku Kuusela, Frank Frederiksen, and Klaus I. Pedersen. Standalone LTE in Unlicensed Spectrum: Radio Challenges, Solutions, and Performance of MulteFire. *IEEE Communications Magazine*, 56(10):170–177, 2018.

[118] Pavan Nuggehalli. LTE-WLAN aggregation [Industry Perspectives]. *IEEE Wireless Communications*, 23(4):4–6, 2016.

[119] Helka-Liina Maattanen, Gino Masini, Mattias Bergstrom, Antti Ratilainen, and Torsten Dudda. LTE-WLAN aggregation (LWA) in 3GPP Release 13 & Release 14. In *2017 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 220–226, 2017.

[120] Ning Zhang, Shan Zhang, Shaohua Wu, Ju Ren, Jon W. Mark, and Xuemin Shen. Beyond Coexistence: Traffic Steering in LTE Networks with Unlicensed Bands. *IEEE Wireless Communications*, 23(6):40–46, 2016.

[121] Prashant Sharma, Ajay Brahmakshatriya, Thomas Valerrian Pasca S., Bheemarjuna Reddy Tamma, and Antony Franklin. LWIR: LTE-WLAN Integration at RLC Layer with Virtual WLAN Scheduler for Efficient Aggregation. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2016.

[122] Yi-Bing Lin, Ying-Ju Shih, and Pei-Wen Chao. Design and Implementation of LTE RRM With Switched LWA Policies. *IEEE Transactions on Vehicular Technology*, 67(2):1053–1062, 2018.

[123] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. TCP Extensions for Multipath Operation with Multiple Addresses. Technical report, Internet Engineering Task Force, 2013. URL https://www.rfc-editor.org/info/rfc6824.

[124] Ramin Khalili, Nicolas Gast, Miroslav Popovic, and Jean-Yves Le Boudec. MPTCP Is Not Pareto-Optimal: Performance Issues and a Possible Solution. *IEEE/ACM Transactions on Networking*, 21(5):1651–1665, 2013.

[125] Sabur Hassan Baidya and Ravi Prakash. Improving the performance of multipath TCP over heterogeneous paths using slow path adaptation. In *2014 IEEE International Conference on Communications (ICC)*, pages 3222–3227, 2014.

[126] Kae Won Choi, Young Su Cho, Aneta, Ji Wun Lee, Sung Min Cho, and Jaehyuk Choi. Optimal load balancing scheduler for MPTCP-based bandwidth aggregation in heterogeneous wireless environments. *Computer Communications*, 112:116–130, 2017.

[127] Christoph Paasch, Ramin Khalili, and Olivier Bonaventure. On the benefits of applying experimental design to improve multipath TCP. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies - CoNEXT '13*, pages 393–398, 2013.

[128] Sinh Chung Nguyen and Thi Mai Trang Nguyen. Evaluation of multipath TCP load sharing with coupled congestion control option in heterogeneous networks. In *Global Information Infrastructure Symposium - GIIS 2011*, volume 6, pages 1–5, 2011.

[129] Filippo Rebecchi, Marcelo Dias de Amorim, Vania Conan, Andrea Passarella, Raffaele Bruno, and Marco Conti. Data Offloading Techniques in Cellular Networks: A Survey. *IEEE Communications Surveys & Tutorials*, 17(2): 580–603, 2015.

[130] Quentin De Coninck, Matthieu Baerts, Benjamin Hesmans, and Olivier Bonaventure. A First Analysis of Multipath TCP on Smartphones. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9631, pages 57–69. Springer International Publishing, 2016.

[131] Iljitsch Van Beijnum. *BGP: Building reliable networks with the Border Gateway Protocol*. O'Reilly Media, Inc., 2002.

[132] E. W. Stevenson. A Border Gateway Protocol 4 (BGP-4). Technical Report 2, Internet Engineering Task Force, 2006. URL https://www.rfc-editor.org/info/rfc4271.

[133] H. Schulzrinne and E. Wedlund. Application-layer mobility using SIP. In *IEEE Globecom '00 Workshop. 2000 IEEE Service Portability and Virtual Customer Environments (IEEE Cat. No.00EX498)*, pages 29–36, 2000.

[134] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. Technical report, Internet Engineering Task Force, 2002. URL https://www.rfc-editor.org/info/rfc3261.

[135] R. Shacham, H. Schulzrinne, S. Thakolsri, and W. Kellerer. Session Initiation Protocol (SIP) Session Mobility. Technical report, Internet Engineering Task Force, 2009. URL https://www.rfc-editor.org/info/rfc5631.

[136] Maarten Weyn, Glenn Ergeerts, Rafael Berkvens, Bartosz Wojciechowski, and Yordan Tabakov. DASH7 alliance protocol 1.0: Low-power, mid-range sensor and actuator communication. In *2015 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 54–59, 2015.

[137] TS 102 690. Machine-to-Machine communications (M2M); Functional architecture, 2013. URL https://www.etsi.org/deliver/etsi_ts/102600_102699/102690/02.01.01_60/ts_102690v020101p.pdf.

[138] Zhengguo Sheng, Chinmaya Mahapatra, Chunsheng Zhu, and Victor C. M. Leung. Recent Advances in Industrial Wireless Sensor Networks Toward Efficient Management in IoT. *IEEE Access*, 3(Oma Dm):622–637, 2015.

[139] Zhigang Wen, Xiaoqing Liu, Yicheng Xu, and Junwei Zou. A RESTful framework for Internet of things based on software defined network in modern manufacturing. *The International Journal of Advanced Manufacturing Technology*, 84(1-4):361–369, 2016.

[140] Jeroen Famaey, Rafael Berkvens, Glenn Ergeerts, Eli De Poorter, Floris Van Den Abeele, Tomas Bolckmans, Jeroen Hoebeke, and Maarten Weyn. Flexible Multimodal Sub-Gigahertz Communication for Heterogeneous Internet of Things Applications. *IEEE Communications Magazine*, 56(7): 146–153, 2018.

[141] Jeroen Hoebeke, Jetmir Haxhibeqiri, Bart Moons, Matthias Van Eeghem, Jen Rossey, Abdulkadir Karagaac, and Jeroen Famaey. A Cloud-based Virtual Network Operator for Managing Multimodal LPWA Networks and Devices. In *2018 3rd Cloudification of the Internet of Things (CIoT)*, pages 1–8, 2018.

[142] Dong Lu, Yi Qiao, P.A. Dinda, and F.E. Bustamante. Characterizing and Predicting TCP Throughput on the Wide Area Network. In *25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, pages 414–424, 2005.

[143] Qi He, Constantinos Dovrolis, and Mostafa Ammar. On the predictability of large transfer TCP throughput. *Computer Networks*, 51(14):3959–3977, 2007.

[144] Mun Choon Chan and Ramachandran Ramjee. TCP/IP Performance over 3G Wireless Links with Rate and Delay Variation. *Wireless Networks*, 11 (1-2):81–97, 2005.

[145] Eymen Kurdoglu, Yong Liu, Yao Wang, Yongfang Shi, ChenChen Gu, and Jing Lyu. Real-time bandwidth prediction and rate adaptation for video calls

over cellular networks. In *Proceedings of the 7th International Conference on Multimedia Systems - MMSys '16*, pages 1–11, 2016.

[146] Yi Sun, Xiaoqi Yin, Junchen Jiang, Vyas Sekar, Fuyuan Lin, Nanshu Wang, Tao Liu, and Bruno Sinopoli. CS2P: Improving Video Bitrate Selection and Adaptation with Data-Driven Throughput Prediction. In *Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference - SIGCOMM '16*, pages 272–285, 2016.

[147] Konstantin Miller, Abdel-Karim Al-Tamimi, and Adam Wolisz. Low-Delay Adaptive Video Streaming Based on Short-Term TCP Throughput Prediction. *arXiv preprint arXiv:1503.02955*, pages 1–34, 2015.

[148] Jianfeng Li, Xiaobo Ma, Junjie Zhang, Jing Tao, Pinghui Wang, and Xiaohong Guan. Mining repeating pattern in packet arrivals: Metrics, models, and applications. *Information Sciences*, 408:1–22, 2017.

[149] Gaetano Carlucci, Luca De Cicco, Stefan Holmer, and Saverio Mascolo. Making Google Congestion Control robust over Wi-Fi networks using packet grouping. In *Proceedings of the 2016 workshop on Applied Networking Research Workshop - ANRW 16*, pages 74–80, 2016.

[150] Yan Liu and Jack Y. B. Lee. An Empirical Study of Throughput Prediction in Mobile Data Networks. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2014.

[151] Yung-Chih Chen, Yeon-sup Lim, Richard J. Gibbens, Erich M. Nahum, Ramin Khalili, and Don Towsley. A measurement-based study of MultiPath TCP performance over wireless networks. In *Proceedings of the 2013 conference on Internet measurement conference - IMC '13*, pages 455–468, 2013.

[152] Kaiping Xue, Jiangping Han, Dan Ni, Wenjia Wei, Ying Cai, Qing Xu, and Peilin Hong. DPSAF: Forward Prediction Based Dynamic Packet Scheduling and Adjusting With Feedback for Multipath TCP in Lossy Heterogeneous Networks. *IEEE Transactions on Vehicular Technology*, 67(2):1521–1534, 2018.

[153] Mohammad Alizadeh, Albert Greenberg, David A. Maltz, Jitendra Padhye, Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, and Murari Sridharan. Data center TCP (DCTCP). In *Proceedings of the ACM SIGCOMM 2010 conference on SIGCOMM - SIGCOMM '10*, page 63, 2010.

[154] Costin Raiciu, Sebastien Barre, Christopher Pluntke, Adam Greenhalgh, Damon Wischik, and Mark Handley. Improving datacenter performance and robustness with multipath TCP. *ACM SIGCOMM Computer Communication Review*, 41(4):266, 2011.

[155] Michael Bredel, Zdravko Bozakov, Artur Barczyk, and Harvey Newman. Flow-based load balancing in multipathed layer-2 networks using OpenFlow and multipath-TCP. In *Proceedings of the third workshop on Hot topics in software defined networking - HotSDN '14*, pages 213–214, 2014.

[156] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[157] Bendong Zhao, Huanzhang Lu, Shangfeng Chen, Junliang Liu, and Dongya Wu. Convolutional neural networks for time series classification. *Journal of Systems Engineering and Electronics*, 28(1):162–169, 2017.

[158] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2017-May, pages 1578–1585, 2017.

[159] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[160] Cristian Hernandez Benet, Andreas Kassler, and Enrica Zola. Predicting expected TCP throughput using genetic algorithm. *Computer Networks*, 108: 307–322, 2016.

[161] Mariyam Mirza, Joel Sommers, Paul Barford, and Xiaojin Zhu. A Machine Learning Approach to TCP Throughput Prediction. *IEEE/ACM Transactions on Networking*, 18(4):1026–1039, 2010.

[162] Jerome H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[163] Tianqi Chen and Carlos Guestrin. XGBoost. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 785–794. ACM Press, 2016.

[164] IEEE Computer Society. 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 802.11-2016 (Revision of IEEE Std 802.11-2012)*, 2016:1–3534, 2016.

[165] Ubiquiti Nanostation M Datasheet. URL https://dl.ubnt.com/datasheets/nanostationm/nsm_ds_web.pdf. Accessed on 2020-02-10.

[166] OpenWrt, . URL https://openwrt.org/. Accessed on 2020-02-10.

[167] B.A.T.M.A.N. advanced, . URL http://www.open-mesh.org/projects/open-mesh/wiki. Accessed on 2020-02-10.

[168] Hideki Kanemoto, Shinichi Miyamoto, and Norihiko Morinaga. Statistical model of microwave oven interference and optimum reception. In *ICC '98. 1998 IEEE International Conference on Communications. Conference Record. Affiliated with SUPERCOMM'98 (Cat. No.98CH36220)*, volume 3, pages 1660–1664, 1998.

[169] Dieter Fiems, Tom Maertens, and Herwig Bruneel. Queueing systems with different types of server interruptions. *European Journal of Operational Research*, 188(3):838–845, 2008.

[170] Anwar Hithnawi, Hossein Shafagh, and Simon Duquennoy. Understanding the impact of cross technology interference on IEEE 802.15.4. In *Proceedings of the 9th ACM international workshop on Wireless network testbeds, experimental evaluation and characterization - WiNTECH '14*, pages 49–56, 2014.

[171] Andrea Conti, Davide Dardari, Gianni Pasolini, and Oreste Andrisano. Bluetooth and IEEE 802.11b coexistence: Analytical performance evaluation in fading channels. *IEEE Journal on Selected Areas in Communications*, 21(2): 259–269, 2003.

[172] Jung-Hyuck Jo and H. Jayant. Performance evaluation of multiple IEEE 802.11b WLAN stations in the presence of Bluetooth radio interference. In *IEEE International Conference on Communications, 2003. ICC '03.*, volume 2, pages 1163–1168, 2003.

[173] K. Shuaib, M. Boulmalf, F. Sallabi, and A. Lakas. Co-existence of Zigbee and WLAN, A Performance Study. In *2006 Wireless Telecommunications Symposium*, pages 1–6, 2006.

[174] Hongwei Huo, Youzhi Xu, Celal Can Bilen, and Hongke Zhang. Coexistence Issues of 2.4GHz Sensor Networks with Other RF Devices at Home. In *2009 Third International Conference on Sensor Technologies and Applications*, pages 200–205, 2009.

[175] Kemal Bicakci and Bulent Tavli. Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks. *Computer Standards & Interfaces*, 31(5):931–941, 2009.

[176] Vittoria de Nitto Personè and Vincenzo Grassi. Solution of finite QBD processes. *Journal of Applied Probability*, 33(04):1003–1010, 1996.

[177] S. K. Gupta and J. K. Goyal. Queues with Poisson Input and Hyper-Exponential Output with Finite Waiting Space. *Operations Research*, 12(1): 75–81, 1964.

[178] Pieter Hintjens. *ZeroMQ: messaging for many applications*. O'Reilly Media, 2013.

[179] Tom De Schepper, Patrick Bosch, Ensar Zeljkovic, Farouk Mahfoudhi, Jetmir Haxhibeqiri, Jeroen Hoebeke, Jeroen Famaey, and Steven Latre. ORCHESTRA: Enabling Inter-Technology Network Management in Heterogeneous Wireless Networks. *IEEE Transactions on Network and Service Management*, 15(4):1733–1746, 2018.

[180] Seung-Que Lee and Jin-up Kim. Local breakout of mobile access network traffic by mobile edge computing. In *2016 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 741–743, 2016.

[181] Fabio Giust, Gianluca Verin, Kiril Antevski, Joey Chou, Yonggang Fang, Walter Featherstone, Francisco Fontes, Danny Frydman, Alice Li, Antonio Manzalini, Debashish Purkayastha, Dario Sabella, Christof Wehner, Kuo-Wei Wen, and Zheng Zhou. MEC Deployments in 4G and Evolution Towards 5G, 2018.

[182] Tarik Taleb, Konstantinos Samdanis, Badr Mada, Hannu Flinck, Sunny Dutta, and Dario Sabella. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration. *IEEE Communications Surveys & Tutorials*, 19(3):1657–1681, 2017.

[183] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B. Letaief. A Survey on Mobile Edge Computing: The Communication Perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358, 2017.

[184] Fabio Giust, Vincenzo Sciancalepore, Dario Sabella, Miltiades C. Filippou, Simone Mangiante, Walter Featherstone, and Daniele Munaretto. Multi-Access Edge Computing: The Driver Behind the Wheel of 5G-Connected Cars. *IEEE Communications Standards Magazine*, 2(3):66–73, 2018.

[185] 3GPP TS 23.501. Technical Specification Group Services and System Aspects: System Architecture for the 5G System, Stage 2 (Release 15),V1.3.0, 2017.

[186] Yifan Yu. SDN-based Local breakout for mobile edge computing in radio access network. *IEEE Wireless Communications and Networking Conference, WCNC*, 2018-April:1–6, 2018.

[187] Sami Kekki, Walter Featherstone, Yonggang Fang, Pekka Kuure, Alice Li, Anurag Ranjan, Debashish Purkayastha, Feng Jiangping, Danny Frydman, Gianluca Verin, and Others. MEC in 5G networks, 2018. URL https://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp28_mec_in_5G_FINAL.pdf. Accessed on 2020-02-10.

[188] Farooq Khan. Mobile Internet from the Heavens. *arXiv preprint arXiv:1508.02383*, page 8, 2015.

[189] Shuang Xu, Xing-wei Wang, and Min Huang. Software-Defined Next-Generation Satellite Networks: Architecture, Challenges, and Solutions. *IEEE Access*, 6(c):4027–4041, 2018.

[190] Shanzhi Chen, Jinling Hu, Yan Shi, and Li Zhao. LTE-V: A TD-LTE-Based V2X Solution for Future Vehicular Network. *IEEE Internet of Things Journal*, 3(6):997–1005, 2016.

[191] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal*, 3(5): 637–646, 2016.

[192] Tessares. Hybrid Access Networks with MPTCP. URL https://www.tessares.net/. Accessed on 2020-02-10.

[193] Pantelis Frangoudis, George Polyzos, and Vasileios Kemerlis. Wireless community networks: an alternative approach for nomadic broadband network access. *IEEE Communications Magazine*, 49(5):206–213, 2011.

[194] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, 2000.

[195] LEDE docs. URL https://lede.readthedocs.io/en/latest/. Accessed on 2020-02-10.

[196] Ismael Gomez-Miguelez, Andres Garcia-Saavedra, Paul D. Sutton, Pablo Serrano, Cristina Cano, and Douglas J. Leith. srsLTE: An open-source platform for LTE evolution and experimentation. In *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, volume 03-07-Octo, pages 25–32, 2016.

[197] MultiPath TCP - Linux Kernel implementation. URL https://multipath-tcp.org/pmwiki.php. Accessed on 2020-02-10.

[198] Jari Kellokoski. Real-life multipath TCP based make-before-break vertical handover. In *2013 IEEE Symposium on Computers and Communications (ISCC)*, pages 000252–000256, 2013.

[199] Christoph Paasch, Gregory Detal, Fabien Duchene, Costin Raiciu, and Olivier Bonaventure. Exploring mobile/WiFi handover with multipath TCP. In *Proceedings of the 2012 ACM SIGCOMM workshop on Cellular networks: operations, challenges, and future design - CellNet '12*, page 31, 2012.

[200] Igor Lopez, Marina Aguado, Christian Pinedo, and Eduardo Jacob. SCADA Systems in the Railway Domain: Enhancing Reliability through Redundant MultipathTCP. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, volume 2015-Octob, pages 2305–2310, 2015.

[201] Bryan Ford, Pyda Srisuresh, and Dan Kegel. Peer-to-peer communication across network address translators. *USENIX 2005 Annual Technical Conference*, pages 179–192, 2005.