

UNIVERSITY CARLOS III OF MADRID

Ph.D. THESIS

ANALYSIS, CHARACTERIZATION AND OPTIMIZATION OF THE
ENERGY EFFICIENCY ON SOFTWAREZED MOBILE PLATFORMS

Author: Carlos A. Donato Morales
University Carlos III of Madrid
Director: Dr. Pablo Serrano Yáñez-Mingot
University Carlos III of Madrid

In partial fulfilment of the requirements for the degree of Doctor in Telematics
Engineering

DEPARTMENT OF TELEMATIC ENGINEERING

Leganés (Madrid), July 2019

Analysis, Characterization and Optimization of the Energy Efficiency on Softwarized Mobile Platforms

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Prepared by
Carlos A. Donato Morales

Under the advice and supervision of
Dr. Pablo Serrano Yáñez-Mingot
Department of Telematic Engineering, University Carlos III of Madrid

Copyright ©2019 Carlos A. Donato Morales

PUBLISHED BY UNIVERSITY CARLOS III OF MADRID

Licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License (the “License”). You may not use this file except in compliance with the License. You may obtain a copy of the License at <http://creativecommons.org/licenses/by-nc/3.0>.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Acknowledgements

Following the Spanish saying “*lo primero es lo primero*” (first things first), I would like to thank Pablo, my advisor of thesis (and friend) for making possible this work and for all of his priceless advises and constant support; without his help I would have never achieved any of this. Thank you Pablo, you can not know how lucky I am to have been your student.

The second round of thanking (and because I would never forgive myself if I did not mention the invaluable help and support) is for Antonio and Carlos Jesus, always open to discussion and sharing their *almost* limitless technical knowledge. Of course, I cannot skip the good of Francesco Gringoli who hosted me in Brescia and from whom I’ve learnt the meaning of hard work, always a source of inspiration. *Grazie mille*, Francesco.

During my *not that* long stay at UC3M and IMDEA, I’ve been enormously lucky of being surrounded by outstanding people. My special thanks and appreciation for those whom became more than lab mates, Sergio G. and Sergio T., Ginés, el *dúo italiano* Marco and Luca, Patri, Nuria, *los researchers de la calle*, Víctor, Jorge and Borja, and all those mates whom accompanied me during this journey. Thank you guys!

Expanding my gratitude, I thank my lifelong friends, those whom have traveled across all the places I have decided to stay whether has been shorter or longer periods: Sergio F. and Sergio C., Héctor, Kike and Arturo. You guys rock!

And finally to conclude, I want to thank the constant support of my family, wherever I go, whatever I decide, no judgements or regrets. This accomplishment is especially for my grandpas, I wish you feel as much proud as my parents do.

Published and submitted content

According to the Law 14/2011 about plagiarism and Code of Good Practices of the UC3M Doctoral School, I hereby include the bibliography of articles and/or other contributions I have (co)authored that are included as part of the thesis and that have been published or submitted for publication.

- F. Gringoli, P. Patras, **C. Donato**, P. Serrano, and Y. Grunenberger. Performance assessment of open software platforms for 5G prototyping. In *IEEE Wireless Communications*, 25(5), 10-15, 2018.

- This work is fully included in Chapter 1 and 2

- The role of author of this thesis was *i*) to design the set of experiments and methodology in order to analyze and characterize the presented SDR platforms, *ii*) to configure and deploy the testbed and tools of the system and, *iii*) to run the different experiments to finally validate the analysis and characterization of such platforms.

- The material from this source included in this thesis is not singled out with typographic means and references

- O. Font-Bach, N. Bartzoudis, M. Miozzo, **C. Donato**, P. Harbanau, M. Requena-Esteso and, M. Payaró. Design, implementation and experimental validation of a 5G energy-aware reconfigurable hotspot. In *Computer Communications*, 128, 1-17, 2018.

- This work is fully included in Chapters 1, 2 and 3

- The role of author of this thesis was *i*) to partially design the energy-awareness platform on top of the existing prototype, *ii*) to design the methodology for energy characterization, *iii*) to carry out the experimental measurements of the platform and, *iv*) to analyze and characterize the energy consumption of such prototype given the different available configurations.

- The material from this source included in this thesis is not singled out with typographic means and references

- N. Bartzoudis, O. Font-Bach, M. Miozzo, **C. Donato**, P. Harbanau, M. Requena, D. López, I. Ucar, A. Azcorra S., P. Serrano, J. Manges and M. Payaró. Energy footprint

reduction in 5G reconfigurable hotspots via function partitioning and bandwidth adaptation. In *Fifth International Workshop on Cloud Technologies and Energy Efficiency in Mobile Communication Networks (CLEEN)*, 2017.

- This work is fully included in Chapters 1, 2 and 3
- The role of author of this thesis was *i)* to partially design the energy-awareness platform on top of the existing prototype, *ii)* to design the methodology for energy characterization, *iii)* to carry out the experimental measurements of the platform and, *iv)* to analyze and characterize the energy consumption of such prototype given the different available configurations.
- The material from this source included in this thesis is not singled out with typographic means and references

■ I. Úcar, **C. Donato**, P. Serrano, A. Garcia-Saavedra, A. Azcorra and, A. Banchs. Re-visiting 802.11 rate adaptation from energy consumption's perspective. In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems* (pp. 27-34). 2016.

- This work is fully included in Chapter 1 and 3
- The role of the author of this thesis was *i)* to design the scenario and experiments that validated the analytic model and, *ii)* to carry out the experimental validation
- The material from this source included in this thesis is not singled out with typographic means and references

■ I. Úcar, **C. Donato**, P. Serrano, A. Garcia-Saavedra, A. Azcorra and, A. Banchs. On the energy efficiency of rate and transmission power control in 802.11. In *Computer Communications*, 117, 164-174, 2017

- This work is partially included in Chapter 1 and 3
- The role of the author of this thesis was *i)* to design the scenario and experiments that validated the analytic model and, *ii)* to carry out the experimental validation
- The material from this source included in this thesis is not singled out with typographic means and references

■ J. Ortín, P. Serrano and, **C. Donato**. Optimal configuration of a resource-on-demand 802.11 WLAN with non-zero start-up times. In *Computer Communications*, 96, 99-108, 2016.

- This work is partially included in Chapter 1 and 4
- The role of author of this thesis was *i)* to partially develop in Octave the analytic model and, *ii)* to validate such model with a simulator written in C++ for this work

- The material from this source included in this thesis is not singled out with typographic means and references

- J. Ortín, **C. Donato**, P. Serrano and, A. Banchs. Resource-on-Demand Schemes in 802.11 WLANs With Non-Zero Start-Up Times. In *IEEE Journal on Selected Areas in Communications*, 34(12), 3221-3233, 2016.

- This work is fully included in Chapter1 and 4

- The role of author of this thesis was *i*) to partially develop and extend the analytic model, *ii*) partially implement it in Octave and, *iii*) to validate such model with a simulator written in C++ for this work

- The material from this source included in this thesis is not singled out with typographic means and references

- **C. Donato**, P. Serrano, A. De La Oliva, A. Banchs and, C.J. Bernardos. An openflow architecture for energy-aware traffic engineering in mobile networks. In *IEEE Network*, 29(4), 54-60, 2015.

- This work is fully included in Chapter 1 and 5

- The role of the author of this was *i*) to design the end-to-end SDN platform for wireless network with support for RoD schemes, *ii*) implement such platform on a real world testbed and, *iii*) validate the system through experimentation

- The material from this source included in this thesis is not singled out with typographic means and references

Other research merits

European Projects

The core work of this thesis has been funded and developed during the execution of the following European Projects:

- **FP7-CROWD** (2013-2015)

CROWD (Connectivity management for eneRgy Optimised Wireless Dense networks) targets very dense heterogeneous wireless access networks and integrated wireless-wired backhaul networks. In this framework, CROWD pursues four key goals: *i*) bringing density-proportional capacity where it is needed, *ii*) optimising MAC mechanisms operating in very dense deployments by explicitly accounting for density as a resource rather than as an impediment, *iii*) enabling traffic-proportional energy consumption, and *iv*) guaranteeing mobile user's quality of experience by designing smarter connectivity management solutions.

The role and the activities of the author of this thesis in the project during the execution of the project are the following:

- Participating from 2014 until the end of the project in 2015
- Mainly involved in WP2, designing and developing a demonstrator that integrated a full SDN scenario with RoD schemes. This demo was based on the OF-TEN platform and optimal configurations found through the analytic models for RoD schemes.

- **H2020-Flex5GWare** (2015-2017)

The scope of Flex5GWare (Flexible and efficient hardware/software platforms for 5G network elements and devices) is to deliver highly reconfigurable hardware (HW) platforms together with HW-agnostic software (SW) platforms targeting both network elements and devices and taking into account increased capacity, reduced energy footprint, as well as scalability and modularity, to enable a smooth transition from 4G mobile wireless systems to 5G. This approach will be necessary so that 5G HW/SW platforms can meet the requirements imposed by the anticipated exponential growth in mobile data traffic (1000 fold increase) together with the large diversity of applications (from low bit-rate/power power

for M2M to interactive and high resolution applications). The role and the activities of the author of this thesis in the project during the execution of the project are the following:

- Participating from the beginning in 2015 until the end of the project in 2017
- Involved in all activities of WP5, especially in the software platform design, definition of the demonstrator and, conducting the research for energy consumption analysis
- Developing one of the eight demonstrators for flexible and multi-connectivity scenarios considering LTE, IEEE 802.11 (WiFi) and IEEE 802.15.1 (Bluetooth).

Other Publications

The following list of publications is not included in this thesis but it exists partial overlap with content of this work.

- J. Ortín, P. Serrano and **C. Donato**. Modeling the impact of start-up times on the performance of resource-on-demand schemes in 802.11 WLANs. In *Sustainable Internet and ICT for Sustainability (SustainIT)*, 2015.

Resumen

La inminente 5^a generación de sistemas móviles (5G) está a punto de revolucionar la industria, trayendo una nueva arquitectura orientada a los nuevos mercados verticales y servicios. Debido a esto, el 5G Infrastructure Public Private Partnership (5G-PPP) ha especificado una lista de Indicadores de Rendimiento Clave (KPI) que todo sistema 5G tiene que soportar, por ejemplo incrementar por 1000 el volumen de datos, de 10 a 100 veces más dispositivos conectados o consumos energéticos 10 veces inferiores. Con el fin de conseguir estos requisitos, se espera expandir los despliegues actuales usando mas Puntos de Acceso (PoA) incrementando así su densidad con múltiples tecnologías inalámbricas. Esta estrategia de despliegue masivo tiene una contrapartida en la eficiencia energética, generando un conflicto con el KPI de reducir por 10 el consumo energético. En este contexto, la comunidad investigadora ha propuesto nuevos paradigmas para alcanzar los requisitos impuestos para los sistemas 5G, siendo materializados en tecnologías como Redes Definidas por Software (SDN) y Virtualización de Funciones de Red (NFV). Estos nuevos paradigmas son el primer paso hacia la *softwarización* de los despliegues móviles, incorporando nuevos grados de flexibilidad y reconfigurabilidad de la Red de Acceso Radio (RAN).

En esta tesis, presentamos primero un análisis detallado y caracterización de las redes móviles *softwarizadas*. Consideramos el software como la base de la nueva generación de redes celulares y, por lo tanto, analizaremos y caracterizaremos el impacto en la eficiencia energética de estos sistemas. La primera meta de este trabajo es caracterizar las plataformas software disponibles para Radios Definidas por Software (SDR), centrándonos en las dos soluciones principales de código abierto: OpenAirInterface (OAI) y srsLTE. Como resultado, proveemos una metodología para analizar y caracterizar el rendimiento de estas soluciones en función del uso de la CPU, rendimiento de red, compatibilidad y extensibilidad de dicho software. Una vez hemos entendido qué rendimiento podemos esperar de este tipo de soluciones, estudiamos un prototipo SDR construido con aceleración hardware, que emplea una plataformas basada en FPGA. Este prototipo está diseñado para incluir capacidad de ser consciente de la energía, permiento al sistema ser reconfigurado para minimizar la huella energética cuando sea posible. Con el fin de validar el diseño de nuestro sistema, más tarde presentamos una plataforma para caracterizar la energía que será empleada para medir experimentalmente el consumo energético de dispositivos reales. En nuestro enfoque, realizamos dos tipos de análisis: *a pequeña escala de tiempo* y *a gran escala de tiempo*. Por lo tanto, para validar nuestro entorno de medidas, caracterizamos a través de análisis

numérico los algoritmos para la Adaptación de la Tasa (RA) en IEEE 802.11, para entonces comparar nuestros resultados teóricos con los experimentales. A continuación extendemos nuestro análisis a la plataforma SDR acelerada por hardware previamente mencionada. Nuestros resultados experimentales muestran que nuestra sistema puede en efecto reducir la huella energética reconfigurando el despliegue del sistema.

Entonces, la escala de tiempos es elevada y presentamos los esquemas para Recursos bajo Demanda (RoD) en despliegues de red ultra-densos. Esta estrategia está basada en apagar/encender dinámicamente los elementos que forman la red con el fin de reducir el total del consumo energético. Por lo tanto, presentamos un modelo analítico en dos sabores, un modelo exacto que predice el comportamiento del sistema con precisión pero con un alto coste computacional y uno simplificado que es más ligero en complejidad mientras que mantiene la precisión. Nuestros resultados muestran que estos esquemas pueden efectivamente mejorar la eficiencia energética de los despliegues y mantener la Calidad de Servicio (QoS). Con el fin de probar la plausibilidad de los esquemas RoD, presentamos un plataforma *softwarizada* que sigue el paradigma SDN, OFTEN (OpenFlow framework for Traffic Engineering in mobile Network with energy awareness). Nuestro diseño está basado en OpenFlow con funcionalidades para hacerlo consciente de la energía. Finalmente, un prototipo real con esta plataforma es presentando, probando así la plausibilidad de los RoD en despliegues reales.

Abstract

The upcoming 5th Generation of mobile systems (5G) is about to revolutionize the industry, bringing a new architecture oriented to new vertical markets and services. Due to this, the 5G-PPP has specified a list of Key Performance Indicator (KPI) that 5G systems need to support e.g. increasing the 1000 times higher data volume, 10 to 100 times more connected devices or 10 times lower power consumption. In order to achieve these requirements, it is expected to expand the current deployments using more Points of Attachment (PoA) by increasing their density and by using multiple wireless technologies. This massive deployment strategy triggers a side effect in the energy efficiency though, generating a conflict with the “*10 times lower power consumption*” KPI. In this context, the research community has proposed novel paradigms to achieve the imposed requirements for 5G systems, being materialized in technologies such as Software Defined Networking (SDN) and Network Function Virtualization (NFV). These new paradigms are the first step to softwarize the mobile network deployments, enabling new degrees of flexibility and reconfigurability of the Radio Access Network (RAN).

In this thesis, we first present a detailed analysis and characterization of softwarized mobile networking. We consider software as a basis for the next generation of cellular networks and hence, we analyze and characterize the impact on the energy efficiency of these systems. The first goal of this work is to characterize the available software platforms for Software Defined Radio (SDR), focusing on the two main open source solutions: OAI and srsLTE. As result, we provide a methodology to analyze and characterize the performance of these solutions in terms of CPU usage, network performance, compatibility and extensibility of the software. Once we have understood the expected performance for such platforms, we study an SDR prototype built with hardware acceleration, that employs a FPGA based platform. This prototype is designed to include energy-awareness capabilities, allowing the system to be reconfigured to minimize the energy footprint when possible. In order to validate our system design, we later present an energy characterization platform that we will employ to experimentally measure the energy consumption of real devices. In our approach, we perform two kind of analysis: at *short time scale* and *large time scale*. Thus, to validate our approach in short time scale and the energy framework, we have characterized through numerical analysis the Rate Adaptation (RA) algorithms in IEEE 802.11, and then compare our theoretical results to the obtained ones through experimentation. Next we extend our analysis to the hardware accelerated SDR prototype previously mentioned. Our

experimental results show that our system can indeed reduce the energy footprint reconfiguring the system deployment.

Then, the time scale of our analysis is elevated and we present Resource-on-Demand (RoD) schemes for ultradense network deployments. This strategy is based on dynamically switch on/off the elements that form the network to reduce the overall energy consumption. Hence, we present a analytic model in two flavors, an exact model that accurately predicts the system behaviour but high computational cost and a simplified one that is lighter in complexity while keeping the accuracy. Our results show that these schemes can effectively enhance the energy efficiency of the deployments and maintaining the Quality of Service (QoS). In order to prove the feasibility of RoD, we present a softwarized platform that follows the SDN paradigm, the OFTEN (Open Flow framework for Traffic Engineering in mobile Networks with energy awareness) framework. Our design is based on OpenFlow with energy-awareness functionalities. Finally, a real prototype of this framework is presented, proving the feasibility of the RoD in real deployments.

Table of Contents

List of Tables	XXIII
List of Figures	XXVII
List of Acronyms	XXIX
1. Introduction	1
1.1. Softwarization of Mobile Networks	1
1.2. Energy Efficiency on Massive Heterogeneous Deployments	4
1.3. Thesis Overview	6
2. Analysis of Software Platforms for Mobile Networking	7
2.1. Full Software SDR solutions	7
2.1.1. Testbed deployment, configuration, and validation	8
2.1.2. Performance evaluation	12
2.1.3. Extensibility and pitfalls	16
2.2. Hardware Accelerated SDR solutions	17
2.2.1. System description	18
2.2.2. System architecture	18
2.2.3. Dynamic hotspot prototype	20
2.3. Conclusions	23
3. Analysis and Characterisation of the Energy Consumption	25
3.1. Platform and Instrumentation for Energy Measurements	25
3.2. Energy Impact on the Rate Adaptation for IEEE 802.11	26
3.2.1. Joint Goodput-Energy Model	27
3.2.2. Numerical Results	30
3.2.3. Energy Consumption	31
3.3. Experimental Validation	33
3.3.1. Experimental Setup	33
3.3.2. Methodology and Results	34

3.4. Energy Consumption in a Energy-aware LTE Platform	36
3.4.1. Testbed setup and measurement devices	36
3.5. Experimental results and discussion	37
3.5.1. Methodology	38
3.5.2. Observed energy consumption results	39
3.6. Conclusions	46
4. Analysis and Optimization of Resource on Demand Schemes	49
4.1. System Model	50
4.1.1. Scenario	50
4.1.2. Resource on Demand policy	51
4.2. Exact Analysis	52
4.2.1. Model overview	52
4.2.2. Modelling the stages of the semi-Markov model	54
4.2.3. Computing the steady-state distribution	60
4.2.4. Performance figures	62
4.3. Simplified Analysis	63
4.3.1. Motivation and simplification	63
4.3.2. Model description	64
4.4. Numerical Results	68
4.4.1. Impact of network load	68
4.4.2. Impact of RoD configuration	70
4.4.3. Computational complexity	71
4.4.4. Realistic traffic model	72
4.4.5. Optimal Configuration of a RoD scheme	72
4.5. Conclusions	73
5. Proof of Concept	75
5.1. Architecture	75
5.1.1. Technology-agnostic operation	77
5.1.2. Maintaining the network vision	78
5.1.3. Installing a new configuration	79
5.1.4. Switching on/off resources	79
5.2. Mapping to technologies	80
5.2.1. Technology dispatcher	80
5.2.2. 802.11 mapping	81
5.2.3. Cellular mapping	82
5.2.4. Device to device mapping	82
5.3. Proof of concept	83
5.3.1. Implementing seamless NIHO	83

5.3.2. Software setup	84
5.3.3. Validation and support of infrastructure on demand	84
5.4. Conclusions	85
6. Summary and Conclusions	87
Appendices	91
A. FPGA energy aware design and system implementation	93
A.1. L2 and upper-layers	93
A.2. L2-L1 interface	96
A.2.1. Energy-aware HWA L1	99
B. Measurement Circuitry Schematics	103
C. Extension of Energy Model for Rate Adaptation	107
References	119

List of Tables

2.1. FPGA-resource utilization of the implemented HWA L1 entities.	22
3.1. Modes of the IEEE 802.11a PHY	31
4.1. Relative differences and computational times of the exact and simplified analyses.	72
4.2. Performance an optimal configurations of a RoD scheme.	74
5.1. Extensions to OpenFlow introduced by OFTEN.	77
C.1. Linear Regressions	108

List of Figures

1.1. Simplified virtual C-RAN deployment	3
2.1. Simplified architecture of the LTE-EPC network	8
2.2. Impact of eNB RX/TX gain on the throughput achievable in the uplink (left) and downlink (right) directions. The eNB employs the srsENB stack over a 10MHz channel, the UE is a Huawei USB dongle, and the propagation medium is coaxial cable.	11
2.3. Methodology validation: throughput performance over wireless vs. wired medium for different eNB/UE combinations, channel bandwidths, and transmission directions.	12
2.4. Throughput performance obtained with different eNB/UE configurations, UL/DL directions, and 5/10 MHz bandwidths. Experiments repeated at a second location (2) with SRS, to illustrate the sensitivity of this stack to PHY conditions. Shown with dashed line is the theoretical maximum throughput in each case.	13
2.5. Experimental results for CPU consumption and network bootstrap times for different combinations of the eNB/UE stacks, traffic direction and UEs	14
2.6. Considered NETCFGs and their related traffic loads.	19
2.7. Overview of the HW setup implementing the hotspot prototype, including the different supported communication function splits.	20
2.8. Picture of the FPGA based testbed	22
3.1. Simplified energy measurement platform	26
3.2. Optimal goodput (bold envelope) as a function of SNR.	30
3.3. Energy Efficiency vs. Optimal Goodput under fixed channel conditions.	32
3.4. Experimental setup.	33
3.5. Energy Efficiency vs. Transmission Power under fixed channel conditions for the Raspberry Pi case.	34
3.6. Experimental study of Fig. 3.5 for two AP configurations.	35
3.7. HW setup utilized to measure the energy consumption of the RF IC.	37
3.8. Picture of the testbed composed by the FPGA system and the energy framework .	38
3.9. Power consumption observed at the GigE NIC for different HeNB configurations.	39

3.10. CPU load resulting from the LENA HeNB process when using a 10 MHz DL BW configuration (MCS index 25).	40
3.11. Impact of the DL BW configuration on the consumption of the baseband SoC. . .	41
3.12. Impact of the RBG allocation on the energy consumption of the baseband SoC. .	42
3.13. Impact of the MCS index on the power drained by the baseband SoC.	42
3.14. Energy savings reported by adopting NETCFG3.	43
3.15. Variation of the power consumption at the RFIC as a result of changing the DL BW.	44
3.16. Variation of the power drained by the RFIC as a result of attenuating the RF output power.	45
3.17. Energy consumed by the RFIC under different RBG allocation cases.	45
4.1. Example of the policy with $K = 2$ APs	52
4.2. Semi-Markov process for an IoD scheme with $N = 4$ APs.	53
4.3. CTMC for stage 1: one active AP and no AP powering on	54
4.4. CTMC for Stage K : K active APs and no AP powering on	55
4.5. Markov chain when all APs are active	57
4.6. Markov chain for the case of K active APs and one AP powering on	58
4.7. Simplified model.	65
4.8. Average power consumption vs. network load.	69
4.9. Average service time vs. average power consumption.	70
4.10. Average service time (top) and power consumption (bottom) vs. target number of users per AP.	71
4.11. Average delay vs. power consumed with non-exponential service demands. . . .	73
5.1. Open Flow-based SDN architecture for heterogeneous networks.	76
5.2. Physical network (top) and abstract vision (bottom). Grey circles denote nodes in power-saving mode, dotted lines represent available links, and solid lines represent used links.	78
5.3. Simplified handover operation.	81
5.4. Deployed testbed and implemented modules and interfaces.	83
5.5. Validation of the proof of concept.	85
A.1. LTE-EPC data plane protocol stack.	93
A.2. Simplified view of the L2-L1 interfacing frame (with an example DL-DCI message).	96
A.3. General overview of the implemented L2-L1 interface.	97
A.4. PS to PL communication and data processing flow.	98
A.5. RTL architecture of the HWA L1.	99
A.6. Control the activity of the DSP stages through clock-gating techniques in order to save energy.	101
B.1. Circuitry schematics for inputs 0-5 V and 0-2 A up to 10 W	104

B.2. Circuitry schematics for inputs 12.5-20 V and 0-5 A up to 100 W	105
C.1. Linear regressions.	107
C.2. Expected energy consumption per frame in <i>millijoules per frame</i> (mJpf) under fixed channel conditions.	108
C.3. Impact of energy parameter scaling on the energy efficiency.	109

List of Acronyms

3GPP 3rd Generation Partnership Project

5G 5th Generation of mobile systems

5G-PPP 5G Infrastructure Public Private Partnership

AP Access Point

BBU Baseband Unit

C-RAN Cloud RAN

CN Core Network

eNB Evolved Node B

EPC Evolved Packet Core

FPGA Field Programmable Gate Array

HeNB Home eNB

HW Hardware

HSS Home Subscriber Server

IoT Internet of Things

IP Internet Protocol

ITU International Telecommunication Union

KPI Key Performance Indicator

LTE Long Term Evolution

MCS Modulation and Coding Scheme

MEC Multi-access Edge Computing

MME Mobility Management Entity

NETCFG Network Configurations

NFV Network Function Virtualization

NGFI Next Generation Fronthaul Interface

NIC Network Interface Card

OAI OpenAirInterface

P-GW PDN Gateway

PoA Points of Attachment

QoS Quality of Service

RA Rate Adaptation

RAN Radio Access Network

RFIC Radio Frequency Integrated Circuit

RRH Remote Radio Head

RoD Resource-on-Demand

S-GW Serving Gateway

SCB Shielded Connector Block

SDN Software Defined Networking

SDR Software Defined Radio

SoC System on a Chip

TCP Transmission Control Protocol

UE User Equipment

UDP User Datagram Protocol

WCP Wireless Communication Parameters

WLAN Wireless Local Area Network

Chapter 1

Introduction

The new revolution for businesses and industry is about to come with the arrival of the Industry 4.0, mainly empowered by the upcoming 5th Generation of mobile systems (5G) and Internet of Things (IoT) technologies where any kind of device will be meant to be connected. Due to the increase of connected and heterogeneous devices, this new generation of networks will need to face a massive growth of the data traffic (i.e., a seven-fold increase is expected in 2021 [1]), while accommodating a wide range of new applications, opening new vertical markets and providing ubiquitous communication services to mobile users across a heterogeneous Radio Access Network (RAN) infrastructure. To address the future network demands and services, the 5G Infrastructure Public Private Partnership (5G-PPP) has specified a set of Key Performance Indicator (KPI) [2] that the 5G systems are expected to support, e.g. the need of field trials, experimentation or energy reduction.

In order to achieve the listed KPIs requirements from the 5G-PPP, the research community has proposed new paradigms such as Network Function Virtualization (NFV), Software Defined Networking (SDN) [3] and Multi-access Edge Computing (MEC) as the core of the new infrastructure. These new technologies rely on software as a first step to open, make more flexible, scalable and reconfigurable cellular network infrastructure, a need for the upcoming services and demand. As consequence of this, experimenting with the new platforms is more affordable than ever, simplifying the hardware requirements to general purpose servers, leaving behind the expensive dedicated equipment [4].

1.1. Softwarization of Mobile Networks

The 3rd Generation Partnership Project (3GPP) has recently completed the specification of the 5G architecture [5] and industry stakeholders are now focusing on finalizing the Release 16 documentation. This will lay the foundations for the cornerstone IMT-2020 standard of the International Telecommunication Union (ITU), which will provide implementation guidelines that will underpin the next generations of mobile broadband and IoT connectivity [6].

Until recently, practical experimentation with cellular systems has been confined to the mobile operators community, primarily due to equipment costs and radio frequency licensing requirements imposed by regulators. In contrast, academics have gained vast experience by experimenting with technology that operates in unlicensed bands [7], such as IEEE 802.11 Wi-Fi and IEEE 802.15.4 ZigBee. Software Defined Radio (SDR) have further facilitated the implementation of novel communications paradigms and medium access schemes (e.g., full-duplex Wi-Fi [8], cognitive radio transceivers [9], and white spaces networking [10]), but the complexity of the “higher layers” of the cellular protocol stack has precluded the development of complete systems by the academic community. However, multiple affordable “open” comprehensive software platforms have emerged over the past years [11, 12]. These put academics almost on par with their industry counterparts, as they are now able to rapidly implement full protocol stacks and evaluate new cellular technologies such as unlicensed LTE [12, 13]. By joining forces in practical experimentation efforts, potential exists to accelerate progress towards 5G standardization and keep up with end-user demands. This ongoing quest towards flexibility and agile reconfiguration is one of the main drivers of the current efforts to softwarize 5G networks. The pairing of SDN and NFV techniques with programmable Hardware (HW) will play a critical role in the implementation of 5G systems and dynamic hotspots in particular.

In this context, centralized processing and very dense deployments are envisioned as key enablers of future 5G deployments. An example of this statement is SDN that over the last few years has gained a lot of popularity. SDN decouples the system that makes decisions about where traffic is sent (i.e., control plane) from the underlying system that forwards traffic to the selected destination (i.e., data plane). With SDN, network administrators can program the behaviour of the network in a centralised way, without requiring to access and configure each of the hardware devices independently, which greatly simplifies overall network management. Additionally, there is an imperative necessity to design 5G networks to be sustainable by bounding their energy consumption in spite of their increased service provision [2]. Because of all the previous, flexibility and reconfigurability need to be secured from conception and at different levels, starting from the underlying HW elements and arriving up to the adaptive management of the network operation. Therefore, flexibly distributing the communication functions across different network elements and adopting enhanced cloud computing schemes allows to transversely optimize the use of infrastructure resources. Yet, selecting the optimal functional split is a complicated task which requires carefully balancing the performance, delay and energy efficiency requirements [14]. From an architectural point of view, this reconfigurability needs to be enabled by intelligently combining a cloudified multi-layer network, embracing both Cloud RAN (C-RAN) and MEC paradigms, with dynamically operated small cells.

The centralization of resources is also a topic of elevated interest in the recent literature, with special attention to C-RAN architectures and communication function splits. Coincidentally, in this context the first prototyping efforts have been also identified, mostly based on pure SW implementations and not accounting for energy-related aspects. Fig. 1.1 shows an example of a

simplified virtual C-RAN deployment. Notably, the authors in [15] present an LTE-based testbed to analyze a medium access control (MAC)-PHY split featuring an Ethernet fronthaul (i.e., favoring the reuse of the existing packet based infrastructure). The analysis is conducted from a point of view of incurred latencies. A closely related contribution is found in [16] where an LTE-based prototype for flexible C-RAN deployments introduces an implementation of the Next Generation Fronthaul Interface (NGFI), which aims at redefining the functional split between the Baseband Unit (BBU) and the Remote Radio Head (RRH). In this respect the authors test two different PHY-layer splits from a functional point of view, with the second one executing part of the digital signal processing (DSP) functions on the RRH. A third relevant effort introduces the use of field programmable gate array (FPGA) devices in a prototype that combines SW and HW accelerated (HWA) functions to test different synchronization procedures in an Ethernet-based fronthaul [17].

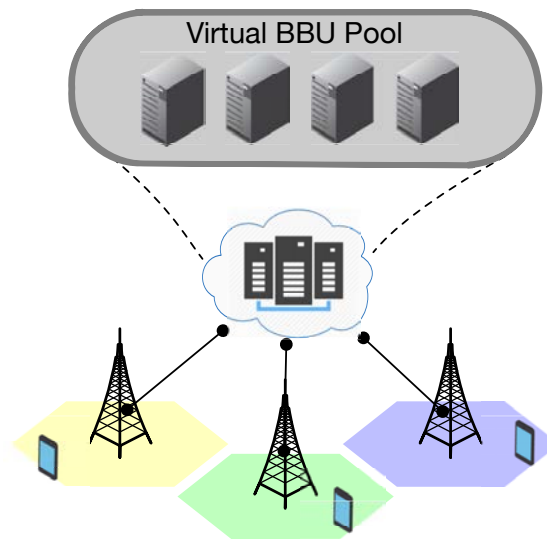


Figure 1.1: Simplified virtual C-RAN deployment

Numerous studies in the heterogeneous C-RAN network framework (i.e., combination of C-RAN and small cells) can also be found in the literature. Frequently, the authors of these studies propose an energy-aware management of the network that consists in switching off those parts which are not required given the current operative requirements. For instance, in [18] a flexible C-RAN prototype for small cells that is based on commercial WiMAX BBUs is described. The authors coarsely evaluate the energy savings obtained as a function of the number of BBUs that are deactivated during low activity periods. In contrast, the work detailed in [19] presents experimental energy measures from a similar traffic-aware C-RAN prototype based on standard server (i.e., computing) nodes. Another relevant development is found in [20], where an LTE-based C-RAN testbed featuring few HWA functions is described. No detailed power consumption numbers are given, but rather a qualitative energy savings analysis is provided.

Finally, the use of FPGAs to implement HWA DSP functions in a SDR context has been continuously growing with the increased capacity of modern programmable devices [21, 22]. More-

over the utilization of advanced digital design techniques to reduce the energy consumption of FPGA implementations is a well-established topic [23–25]. Recently, the emergence of FPGA-based system-on-chip (SoC) devices featuring an unprecedented combination of performance and flexibility, has introduced the means to efficiently implement SDN systems [3, 26, 27] and opened the door to the dynamic distribution of SW and HWA functions. Unfortunately, few contributions are found in the literature in this regard, which are mainly comprised by partial FPGA implementations, such as the common public radio interface (CPRI)-based C-RAN systems presented in [28, 29]. Furthermore, a fixed function split is commonly adopted, as showcased in [30], where the authors describe a simple C-RAN system featuring dynamic offloading of MAC functions onto the Cloud and analyze the energy gains obtained by optimizing the number of active processors.

1.2. Energy Efficiency on Massive Heterogeneous Deployments

The solution to cope with this growing traffic demand and the specified KPIs (e.g. 1000x higher mobile data volume per geographical area) necessarily entails using more Points of Attachment (PoA) such as IEEE 802.11 Access Point (AP) or LTE small cells, by increasing their density and by using different wireless technologies, as well as offloading the network infrastructure (through e.g., *device-to-device communication*) [31]. Besides the standard static small cell installations, dynamic hotspots are also required to provide on-demand services in localized spaces and time periods (e.g., due to a large increase of capacity requirements). In such heterogeneous environment, optimizing the energy footprint is key to ensure the sustainability and economic viability of small cells [32] (e.g., temporary deployments might be battery-powered) and AP based on IEEE 802.11. Adapting the transmission strategy is also fundamental from a practical point of view, since small cells need to coexist with other RAN infrastructure elements (e.g., interferences, limited spectrum resources) and to efficiently serve varying traffic and quality of service (QoS) requirements. Consequently, beyond providing a rigid macro-cellular traffic offloading scheme, small cells are principal facilitators of the intelligent distribution of communication functions with respect to the dynamically varying demands of the mobile users.

Besides the KPIs defined by 5G-PPP, there is a growing interest in the research literature to investigate small cell deployments which are aimed at addressing the capacity requirements in dense indoor and outdoor urban environments. In particular, many research efforts are focusing on proposing methods to efficiently reduce the amount of energy drained by these network installations. The great majority of such analyzes are based on computer simulations. A relevant example of the latter is found in [33] where a deployment of nomadic nodes is proposed (i.e., vehicle mounted battery-powered relays) in order to provide temporary demand-driven service provisioning to mobile users. The authors analyze the energy saving gains obtained by dynamically switching off those nomadic cells with no traffic demand. Similarly, the authors in [34] analyze a database-aided scheme that enables the use of deep sleep modes (i.e., completely switch off the transceiver HW) in clustered small cell deployments (e.g., train stations, shopping malls).

Energy-saving gains up to 30% are observed in the simulated scenarios. A low-latency time division duplex (TDD)-based physical (PHY) air interface is proposed and modelled in [35] to be used in 5G dense deployments. Focusing on the user equipment (UE), the authors argue that the battery lifetime can be largely extended by using the proposed scheme instead of LTE.

The side effect of this strategy though, is the increase of the power consumed, which can result in energy wastage if all the infrastructure is kept powered on when the load is low [36, 37]. Techniques to “green” the operation of the network include the design of more energy efficient hardware [38], the optimization of the radio transmission chain [39], or the implementation of Resource-on-Demand (RoD) strategies that dynamically adapt to the network load, activating resources as it grows and deactivating them when it shrinks [40]. When the number of hotspots and AP is increased to cope with the growth of the data demand, the total power consumption of the network linearly rises. RoD schemes are relatively easy to deploy, as they do not require the introduction of major changes in the network, and have been proposed to decrease the energy consumption of base stations in “traditional” mobile networks (GSM, UMTS) [41, 42], as these devices account for up to 60% of the total energy consumed [38]. Following [43], RoD policies can be divided into static and dynamic, depending on whether the switching on/off of the devices is scheduled or it follows real-time traffic patterns. In general, dynamic approaches are more efficient than static solutions, although they require higher switching on/off rates. In [40], a number of dynamic approaches are classified according to the wireless technology, performance metric, reaction time of the algorithm and control scheme (centralized or distributed).

Regarding Wireless Local Area Network (WLAN), it has been shown that RoD techniques can potentially provide substantial gains in energy savings when the number of considered APs increases (gains of approx. 37%, or 26 kW, can be achieved for a university campus [44] or even higher [45]). Several publications have shown the feasibility of RoD policies in campus networks [46, 47]. The first analytical model for these techniques [48] focuses on the case of “clusters” of APs covering the same area, and studies the impact of the strategy used to (de)activate APs on parameters such as the energy savings and the switch-off rate of the devices. In [49], the work is extended to account for the case when APs do not completely overlap their coverage areas. Following this interest in RoD schemes for WLANs, new publications analyze the performance when some assumptions are relaxed,¹ e.g., in [50] authors analyse the impact of using an accurate energy consumption model on performance.

When the energy efficiency of PoAs is dissected, it is generally assumed that optimality in terms of throughput also implies optimality in terms of energy efficiency. However, some recent work [51, 52] has shown that throughput maximisation does not result in energy efficiency maximisation, at least for 802.11n. However, we still lack a proper understanding of the causes behind this “non-duality”, as it may be caused by the specific design of the algorithms studied, the extra consumption caused by the complexity of MIMO techniques, or any other reason. In fact, it could be an inherent trade-off given by the power consumption characteristics of 802.11 interfaces, and,

¹In fact, in both [40] and [43] it is noted that current RoD policies are made using over-simplified models.

if so, RA techniques should not be agnostic to this case.

There is also the added motivation of a limited energy supply (i.e., batteries), which has triggered a relatively large amount of work on energy efficiency [41]. It turns out, though, that energy efficiency and performance do not necessarily come hand in hand, as some recent research has pointed out [53, 54], and that a criterion may be required to set a proper balance between them.

1.3. Thesis Overview

In this work, we present an analysis and characterization of the energy efficiency on software-defined mobile networks, presenting first a set of existing technologies for experimentation with mobile networking. Then, we introduce a high-accurate energy measurement framework that is the core of energy analysis for IEEE 802.11 and LTE systems. Later, we study other ways to reduce energy consumption in ultradense scenarios considering RoD schemes, providing optimal configurations for a trade-off between energy efficiency and network performance. Lastly, we present a proof of concept that implements a novel SDN platform designed to support RoD schemes and traffic engineering

The remainder of this thesis is organized as follows:

- In Chapter 2, we provide a detailed analysis and methodology to evaluate different software platforms for experimenting with real life LTE systems. Moreover, we study a FPGA based prototype with energy-awareness and support for function split of the LTE stack.
- In Chapter 3, we analyze the energy efficiency of the current rate adaptation algorithms for IEEE 802.11 based devices. We extend our analysis to the full software-defined LTE prototype
- In Chapter 4, we present an analytic model and performance evaluation of RoD schemes. We also study the trade-offs between energy consumption and network performances in ultradense PoA deployments
- In Chapter 5, we introduce the OFTEN framework, an SDN architecture with support for RoD schemes and traffic engineering for mobile networking. Then, a proof of concept is shown in order to validate first the proposed architecture and second, the strategies based-on RoD schemes.
- In Chapter 6, the final remarks and conclusions from this thesis are presented.

Chapter 2

Analysis of Software Platforms for Mobile Networking

The last generations of general purpose CPUs in combination with the reduction of the Software Defined Radio (SDR) cards cost have made possible to experiment and play around with radio stacks on relative cheap equipment. In the case of Long Term Evolution (LTE), several open source projects have proved that it is possible to deploy a whole platform that emulates a real network operator. In order to show this experimentation environment, we interconnect different platforms (including open software solutions and consumer-grade user equipment) to build end-to-end LTE systems, and assess their performance and adequacy towards 5G experimentation. Despite the rise of radio implementations by means of full software stack¹ for general purpose CPUs, we also consider a hardware accelerated platform empowered by Field Programmable Gate Array (FPGA) boards. In this chapter, we analyze and explore both means of experimentation and prototyped on LTE platforms.

2.1. Full Software SDR solutions

We first focus on two popular solutions that provide a complete version of the protocol stack, namely, OpenAirInterface (OAI) [11] and srsLTE [12].² OAI is licensed under the OAI Public License³ and implements a subset of the LTE Release 10 specification, including key elements such as User Equipment (UE), Evolved Node B (eNB), Mobility Management Entity (MME), Home Subscriber Server (HSS), Serving Gateway (S-GW), and PDN Gateway (P-GW) as shown in Fig 2.1, thus supporting a complete solution. OAI runs over commodity Linux-based computing equipment (Intel x86 PC architectures) and can be used with popular radio frequency (RF)

¹We denominate “full software implementation” to those radio stacks that fully run on the host PC which, it is the case for the srsLTE framework and OAI

²We do not include openLTE (<http://openlte.sourceforge.net/>) in our analysis due to its limited functionality (e.g., lack of a User Equipment software) and poor robustness, as compared with the other two solutions considered.

³http://www.openairinterface.org/?page_id=698

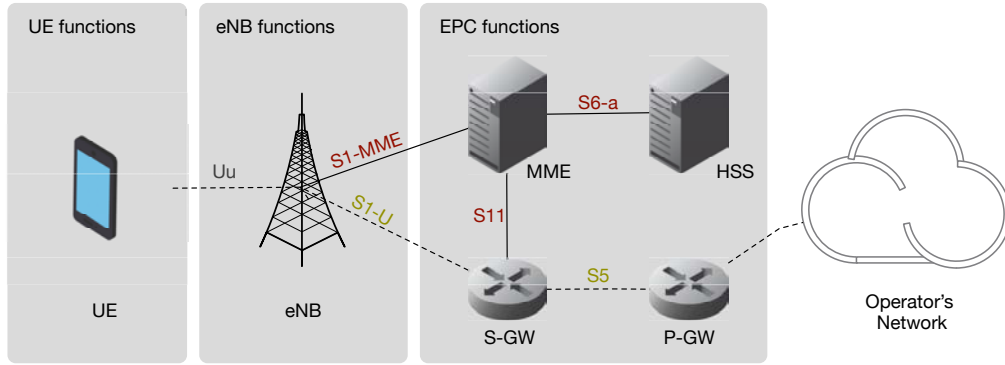


Figure 2.1: Simplified architecture of the LTE-EPC network

frontend equipment (e.g., National Instruments PXIe and Ettus USRP). srsLTE is licensed under the GNU Affero General Public License⁴ and also runs over a similar set of commodity equipment, providing implementations of the UE and eNB that are compliant with LTE Release 8. In addition to being compatible with commercial Core Network (CN) software solutions such as Amarisoft⁵ or open-source alternatives including the OpenAir-CN, as we demonstrate in this section, it has also recently released an implementation of the same CN elements.⁶ Despite not constituting complete 5G systems, these platforms have been employed successfully for the development of several 5G technologies, including LTE unlicensed [12, 13], network slicing [55], neutral host deployments, or RAN sharing [56].

While there is certain *unwritten consensus* about the features of each of these platforms, namely, OAI being more computationally efficient than srsLTE, srsLTE’s code base being more modular and easier to customize than OAI’s, there have been limited previous efforts to formally analyze their performance, or their interoperability. Existing work in this space only profiles the OAI components and measures its *emulation* execution time [57], or characterizes OAI’s base-band processing times under a range of conditions [58]. In this work, we set the record straight and carry out an extensive measurement campaign to characterize the performance of each platform under a variety of settings (including different UEs, channel bandwidths, and propagation conditions). We also analyze their interoperability, and discuss the degree of customization and extensibility they allow. We believe our results provide valuable insights and best practices to researchers and practitioners faced with deciding which platform(s) to consider for their experimental work.

2.1.1. Testbed deployment, configuration, and validation

We first describe the hardware and software used in our testbed. Then, we detail the methodology we designed to set up the experiments. Lastly, we experimentally confirm that the per-

⁴<https://github.com/srsLTE/srsLTE/blob/master/LICENSE>

⁵<https://www.amarisoft.com/>

⁶We were not able to assess the performance with these elements though, as they were not available by the time we ran our experiments.

formance observed in wireless scenarios is practically the same as then when coaxial cables are employed, which enables us to discard with confidence the impact of the transmission medium on the results.⁷

2.1.1.1. Testbed and tools

We conduct all the experiments reported using an eNB built with the Ettus USRP-B210 radio frontend boards connected using USB 3.0 to a Linux-based host computer that runs the different software suites considered. The frontend's up- and down-link interfaces are multiplexed on a single 2dB dipole antenna through a RF Diplexer compatible with LTE band 7. The host PC runs Ubuntu 16.04 and is equipped with an Intel Core i7-7700K CPU with four cores clocked at 4.2GHz, which is powered by an ASUS Z270-A motherboard and 16GB of DDR4 memory. We remark that a configuration with such high computing power is required to be able to run effortlessly the different software solutions that we use, since baseband processing is particularly CPU expensive.

Using the configuration described above, we conduct experiments with two types of eNB software, namely:

- **OAI** – version 0.6.1;
- **SRS** – version 2.0-17.09 of the srsENB application.⁸

We use an additional computer to host the CN functionality. It runs the OpenAir-CN stack, which implements an open-source Evolved Packet Core (EPC): MME, HSS, and P/S-GW modules. The computer providing eNB functionality through the different solutions considered is connect to the CN using a Gigabit Ethernet link. Although we could virtually run both the Radio Access Network (RAN) and CN stacks on the same physical equipment in order to reduce deployment costs, the configuration we adopt ensures the performance of the eNB will not be affected by the computing load placed by the processes specific to the CN.

In terms of UEs, we experiment with three different solutions: two of these are commercial off-the-shelf (COTS) equipment, and the third one is an SDR platform with open-source software commonly used for cellular testing purposes. Precisely, we work with an LG Nexus 5 smartphone (denoted 'Nexus' throughout our experiments), a Huawei e3272-153 USB dongle (denoted 'Huawei'), and respectively the Ettus USRP B210 board running the srsUE stack version 2.0-17.09 (denoted 'srsUE').⁹ To control the Nexus 5, we connect it through its USB interface to

⁷We note that we faced a number of performance issues (e.g., configuration-dependent incompatibilities, out-of-memory crashes) during our experiments, which was somehow expected as both open projects are actively evolving. We report the most relevant ones, for the software versions available at the time of testing, in Section 2.1.3.

⁸The srsLTE software suite comprises an eNB stack (srsENB) and the UE stack (srsUE). Hereafter, whenever there is no scope for confusion, we use SRS and srsENB interchangeably to refer to the eNB software.

⁹Although an OAI UE implementation exists, we found that this was incompatible with the SRS eNB software. Therefore, for a fair comparison of the eNB stacks, we do not report the performance of an OAI (eNB, UE) pair.

a PC-Engines APU embedded computer,¹⁰ which runs the Android Debug Bridge software¹¹ to access the internal command console of the smartphone. We use the same APU device with the Huawei dongle, which we supply with external power through an USB hub, to prevent power-related instability issues. To be able to test with the commercial equipment considered, we fit the smartphone and USB dongle with Sysmocom programmable SIM cards.¹²

During each experiment, only one UE is connected to the eNB. Furthermore, to avoid external interference on our experiments, as well as our experiments interfering with external networks, we placed our network setup within an RF shielding tent that is easy to set up and provides attenuation in the -50 to -60dB range.¹³ Finally, to generate and receive traffic, we install `iperf`-compatible software on the Nexus 5, the APU device hosting the Huawei dongle, and the computer running as SDR UE. During our experiments, we transmit UDP traffic in the uplink or downlink directions, and collect statistics at the receiver by sampling `iperf`'s output every second. We also measure the total CPU usage of the SDR process at the eNB by executing the `ps` command and querying the corresponding process ID.

2.1.1.2. Experiments set up and repeatability

For each combination of equipment and software we first devise a procedure for determining the optimal configuration of the TX and RX gains at the eNB. This turned out to be required to ensure the resolution of the ADC/DAC is not compromised or the receiver does not saturate. This is because both implementations do not adjust the output power level of the boards, but rather adjust signal sample amplitudes by scaling. The procedure consists of running very short experiments (e.g., UDP traffic sessions up to 5 seconds) and measuring the throughput obtained for each combination of gain. With these measurements we fill throughput matrices for the uplink (UL) and downlink (DL) directions, as shown in Fig. 2.2, which illustrates the case when the eNB runs `srsENB` over 10 MHz channels and the UE is the Huawei USB dongle connected with coaxial cables to the eNB. Subsequently, we employ the gain pair that yields the highest throughput. Depending on the direction of each experiment run, we may set different gain pairs. We note that the implementation of the OAI stack is more accurate, since in most cases setting these hardware gains to the maximum value led to highest throughput performance.

Before each repetition of a conducted test, we completely restart the network and measure the time required to establish a working client connection, i.e., until the UE has Layer 3 connectivity with the CN, as verified with ICMP echo request/reply. On the one hand this enables us to build statistical significance for the performance metrics of interest. On the other hand, we also collect data about the bootstrap instances that did not conclude successfully, in order to report on the reliability of specific configurations (this will be done in Section 2.1.2.4). As such, we bootstrap

¹⁰<http://www.pcengines.ch/apu.htm>

¹¹<https://developer.android.com/studio/command-line/adb.html>

¹²<http://shop.sysmocom.de/products/sysmousim-sjs1>

¹³<http://www.globalemc.co.uk/shielded-tents.php>

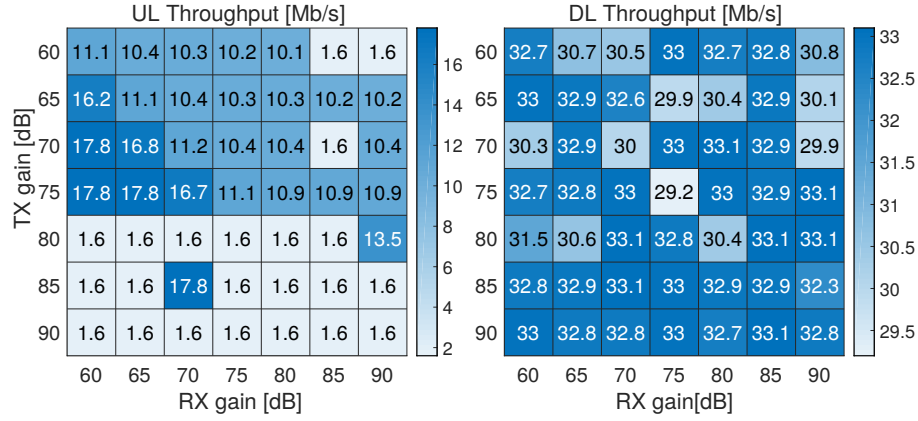


Figure 2.2: Impact of eNB RX/TX gain on the throughput achievable in the uplink (left) and downlink (right) directions. The eNB employs the srsENB stack over a 10MHz channel, the UE is a Huawei USB dongle, and the propagation medium is coaxial cable.

experiments through a set of Bash scripts that involve the following steps:

1. Starting the CN and verifying that all processes are working and remain alive;
2. Starting the eNB and waiting for connection to the HSS;
3. Starting the UE, waiting until this finds the cell, and initiating the attachment process;
4. Waiting until the UE obtains an IP address and receives an ICMP echo reply from the traffic generator running at the CN;
5. Measuring the achievable throughput by running five consecutive transmissions, each of 5 s duration.

After the network has been setup successfully, we run 30 s long transmission experiments, setting the offered close to the the maximum throughput measured at the last step of the bootstrap procedure.

2.1.1.3. Validation

To confirm that the proposed evaluation methodology is reliable and not susceptible to inaccuracies induced by practical signal attenuation and multipath propagation, we first compare the throughput attainable in both directions (UL and DL) over 5 and 10 MHz channels with all the possible SW/HW configurations, when the eNB and UE communicate over a wireless channel (air) and over coaxial cables using SMC connectors (cable), respectively. Note that our validation does not include experiments with the Nexus smartphone, as instrumenting wired connections would have voided the device's warranty. We illustrate the results of these preliminary experiments in Fig. 2.3.

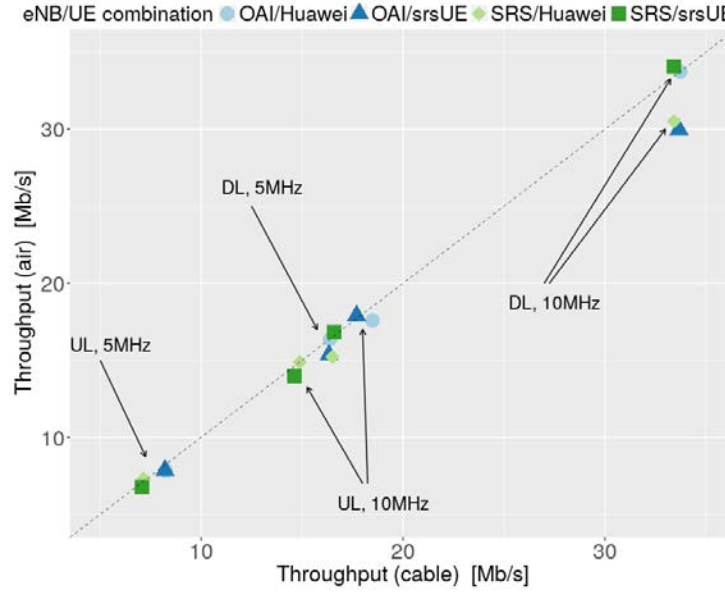


Figure 2.3: Methodology validation: throughput performance over wireless vs. wired medium for different eNB/UE combinations, channel bandwidths, and transmission directions.

In this figure, we plot the throughput performance over the air vs. that obtained over the wired link. We investigate this for each direction, bandwidth, and eNB/UE configuration considered. Observe that for each combination the results are highly correlated, with a Pearson correlation coefficient of $r = 0.993$, which proves that the communication medium has practically no impact on the achievable performance. Therefore, we are confident that the performance differences, which we report in the next section for all the possible configurations, have other root.

2.1.2. Performance evaluation

In this section, we compare the performance in terms of throughput, CPU consumption, and bootstrap time achievable with different eNB stacks (OAI vs srsENB), UEs (Nexus smartphone, Huawei USB dongle, and srsUE with USRP SDR), transmission directions (UL/DL), and bandwidth (5 and 10 MHz) configurations.

2.1.2.1. Throughput performance

We start our analysis by assessing the throughput performance when sending unidirectional UDP traffic at the maximum rate achievable in the uplink (from the UE) and downlink (from the eNB) directions. To this end, we use `iperf` to generate 1500 B frames from the corresponding side (the UE in the former case, and the computer hosting the CN stack in the latter), and measure the average throughput obtained every second, during 30 s trials. We plot in Fig. 2.4 the average and 95-percent confidence intervals obtained following 20 repetitions of each test. In this figure each direction is represented with a different color, and each subplot corresponds to a different

configuration of the eNB (OAI or SRS) and BW used (5 MHz or 10 MHz). To add perspective, we show with dashed lines the theoretical maximum throughput achievable in each case.

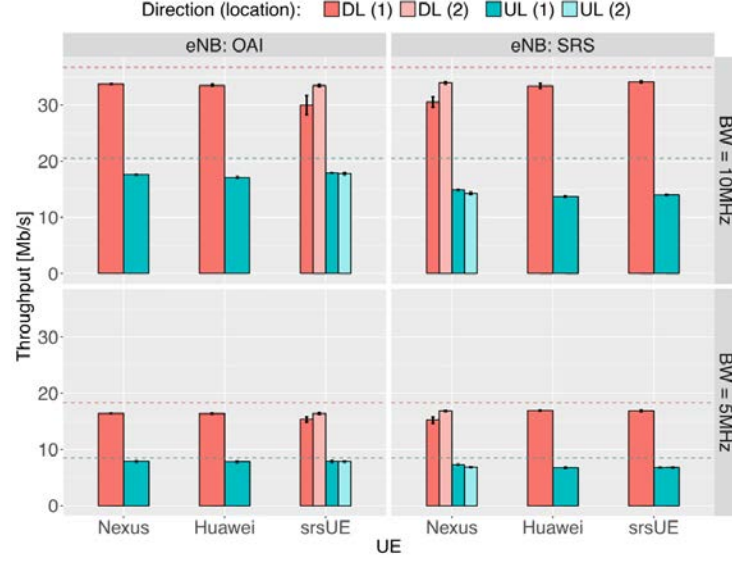


Figure 2.4: Throughput performance obtained with different eNB/UE configurations, UL/DL directions, and 5/10 MHz bandwidths. Experiments repeated at a second location (2) with SRS, to illustrate the sensitivity of this stack to PHY conditions. Shown with dashed line is the theoretical maximum throughput in each case.

There are two key results worthwhile remarking in the figure, apart from the obvious performance asymmetry in terms of the UL/DL rates, which is due to the fact that the DL operates with 64QAM and by default the UE transmits using 16QAM. Firstly, the type of UE used has practically no impact on the performance attainable with a given eNB, since in all cases the obtained throughput is almost “flat.” We only note minor differences between the commercial equipment’s performance (i.e., Nexus and Huawei) and that of the experimental platform (srsUE). In particular, for the case of {DL, 10 MHz} (top left subplot) we note a slight drop in performance when srsUE is employed with OAI. We also note that initially the throughput attained with the (SRS, Nexus) combination was lower. Our intuition is that the signal processing performed by the SRS stack (both in the eNodeB and UE implementations which are built on the same library) is different from that of OAI. Specifically, in some scenarios an SRS receiver may underestimate the channel quality, or produce a stream of samples that cannot be correctly decoded by a commercial receiver. To confirm this we repeated these tests with the UE placed at a second location, where the performance of SRS was at the same leveled with that of OAI – note the lighter bars labeled with “(2)” in the figure. As the eNB and UE use the same PHY stack with SRS, any mismatch is less likely to occur.

Secondly, we note the UL rates are slightly smaller when the eNB runs the SRS stack instead of OAI, something that is particularly noticeable for the 10 MHz configuration. As we analyze next, this can be related to the CPU demanded by the srsENB solution when decoding traffic,

particularly in the uplink direction. Despite these small variations, it is also worth remarking that SRS and OAI provide a similar throughput performance, with the difference over all cases being on average smaller than 7%.

2.1.2.2. CPU Usage

Next, we analyze the CPU usage of the cellular SW stack running at the eNB, aiming to quantify the differences in terms of resources consumed by the OAI and SRS solutions. For this purpose, we repeat the same experiments performed above, identifying the process ID corresponding to the stack of interest running at the eNB, then invoking the `ps` command every second, to estimate the CPU load. We illustrate the average and 95% confidence intervals of the CPU consumption for all combinations considered in Fig. 2.5a, where the subplots are arranged as in the case of the previous experiments.

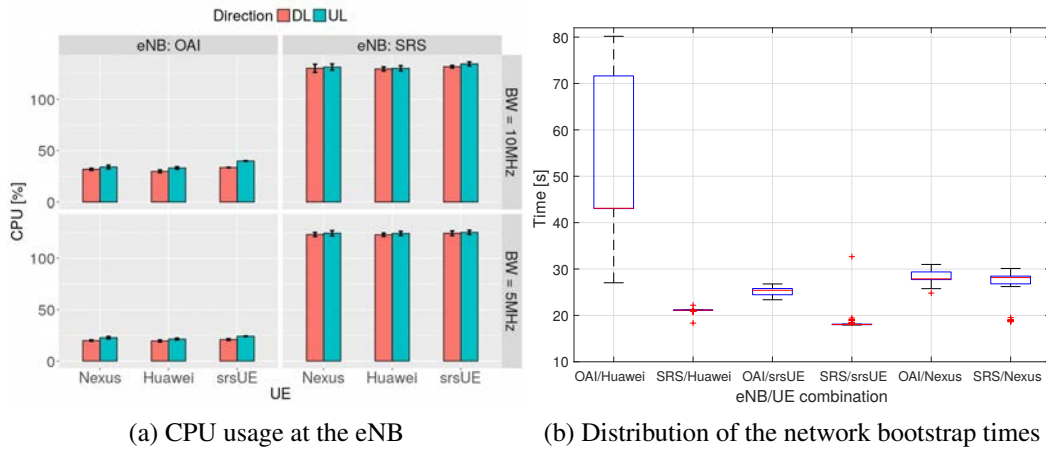


Figure 2.5: Experimental results for CPU consumption and network bootstrap times for different combinations of the eNB/UE stacks, traffic direction and UEs

The figure confirms that the choice of UE has little impact on resource consumption, as all results are very similar for a given configuration of eNB and bandwidth. It also confirms that there is a notable difference in terms of CPU usage between the SRS and OAI stacks, the former consuming more than four times more CPU cycles than the latter. These results suggest that OAI is more suitable for future 5G scenarios, where computational efficiency is of paramount importance, such as Cloud RAN. For all configurations, the UL direction is slightly more CPU demanding than the DL (9.7% on average), which we conjecture is caused by the decoding operations [59].

Finally, it is also worth observing the relative impact of the bandwidth configured on the CPU consumption. Using a simple linear regression model, admittedly built with limited data, we can roughly estimate that OAI may consume on average 2.5% CPU time for every additional MHz of bandwidth, whilst exhibiting an “idling” cost of 9.1% CPU usage. Interestingly, SRS only appears to add some 1.4% CPU consumption for every additional MHz of bandwidth, but the stack may

demand 116.8% CPU time even when the channel bandwidth would be virtually zero.

2.1.2.3. Multiple UEs

We also investigate whether the number of UEs attached to the eNB stacks considered might have an impact on the total throughput performance and CPU usage, due to additional processing that may be required. At the same time, we wish to understand if the schedulers implemented ensure the available resources are shared fairly among the UEs. Therefore, we deploy two UEs that implement the srsUE stack and measure the metrics of interest with both eNBs. The obtained results confirm that adding more clients does not impact on the total network throughput or the CPU usage of neither of the eNB stacks. In addition, both UEs obtain equal throughput, which confirms the accuracy of the schedulers. As a final note, following code inspection we find that OAI implements a proportional fair scheduler, whilst srsENB employs round robin scheduling.

2.1.2.4. Network bootstrap and reliability

Next, we take a closer look at the time required to successfully bootstrap the network setup and how often this process may fail on average. We argue this is particularly important to understand the degree of experiment automation and repeatability achievable with these platforms. To this end, based on the data collected prior to executing the previous experiments, we summarize in Fig. 2.5b the distributions of the time elapsed until the UE has obtained an IP address and thus has a Layer 3 connection (steps 2–4 described in Section 2.1.1.2) for the different eNB/UE combinations, over a wireless channel. We resort to box and whisker plots for this purpose, the central lines marking the medians, the boxes' lower and upper margins the 25th and respectively 75th percentiles, and the whiskers the minimum and maximum values, excluding outliers, which we plot separately (crosses).

Observe that the SRS stack displays the most deterministic behavior, especially with the Huawei dongle and the SDR running srsUE. Indeed, if we exclude outliers, the bootstrap time is constant and less than 22 s. With the same UE types, the OAI stack performs substantially different. In particular, while the bootstrap time is fairly constant and close in magnitude to that observed when the eNB runs the SRS stack (approximately 25 s median value) and the srsUE is used, the range of bootstrap times is very large when the UE used is the USB dongle (Huawei). In this case, the median is also higher than that measured in all the other setups and the 75th percentile is more than 70 s. This is particularly inconvenient, since when experimenting for performance assessment purposes, setting up the network takes more time than running the actual traffic. In case the Nexus smartphone is used as UE, both eNB types exhibit similar statistics, i.e., the median of the bootstrap time is approximately 28 s and the variations are relatively small (less than 2 s between the 1st and 3rd quartiles). We leave an analysis of the reasons behind these differences for future work.

We also count the number of times we detected a failure of the bootstrap procedure or during

experiment runtime for each of the eNB/UE combinations considered, in the process of conducting a total of 80 successful experiments in each case, which we reported earlier (20 repetitions for each direction and bandwidth setting). Interestingly, we find that the most reliable UE type is the srsUE, as we never encountered any failures when connecting this to either of the eNB types used and the network never failed during the experiments. We observed a similar behavior with the USB dongle only when connecting to the OAI eNB stack, while with SRS we measured a 5.9% failure rate. Here, failures occurred always after the network was formed successfully, during the initial short tests that we run to assess the sustainable throughput (step 5 of the set up procedure described earlier). In contrast, in the case of using the Nexus as UE, we observed a 7% failure rate with OAI, as the network did not bootstrap altogether, and a 2.5% failure rate with SRS, due to network breakdown during experiments.

2.1.3. Extensibility and pitfalls

2.1.3.1. Customization and extensibility

We next comment on ability to customize and extend the functionality of the platforms considered. To this end, we focus on particular issues related to scheduling and Modulation and Coding Scheme (MCS) assignment. To ensure that the two solutions studied are evaluated under the same conditions and all comparisons are fair, we decided to focus on the following customization: introduce the ability to fix during experiments the MCS assignments which the eNB MAC scheduler enforces on UEs.¹⁴ Achieving this turned out to be fairly intuitive with srsLTE, as we found the function responsible with scheduling and MCS assignment in `srsLTE/srsenb/src/mac/scheduler_ue.cc`, conveniently named `sched_ue` and the code easy to modify.

On the other hand, this task turned out to be less straightforward with OAI. The source code that implements the MCS assignment operation is located within the folder `openairinterface5g/openair2/LAYER2/MAC/`, where files related to MAC scheduling and MCS index assignation are located. After thorough code inspection, we found that all the files therein contain code that changes the MCS settings and unfortunately the MCS index is also often hard coded in places. Following debugging, we inferred that the files `eNB_scheduler_ulsch.c` and `eNB_scheduler_dlsch.c` contain the functions `schedule_ulsch_rnti` and `schedule_dlsch_rnti` that assign the MCS in the UL and DL directions. We developed a patch to enable dynamic MCS index assignment, though after applying our patch we discovered that the MCS will be later computed and altered by other functions. Therefore we were unable to achieve the desired behavior.

We believe the major differences between the OAI and srsLTE solutions in terms of extensibility are because they follow different software designs. In particular, OAI was developed for

¹⁴In the case of srsENB, the software includes code to fix the MCS index at start-up through a configuration file, but our aim is to dynamically change the MCS externally during execution time.

mock LTE network deployment with a built-in emulator, while srsLTE was designed from scratch as a framework to support building LTE applications on top, providing a set of common libraries, tools, and examples for PHY layer implementation and experimentation. As a result, UE and eNB apps were implemented on top of these libraries. From a software design perspective, srsLTE offers a modular framework that re-factors the code of common LTE functions for any application, whilst OAI is designed to offer an standalone eNB solution.

2.1.3.2. Software stack pitfalls

Working with open-source cellular stacks has important benefits, including speed of deployment, availability of documentation, affordable cost, and ability to extend functionality. Unfortunately, such solutions come with their own set of issues, some of which are more difficult to spot and which can hinder the reproducibility of results. Here we highlight the main pitfalls we identified while experimenting with the OAI and SRS tools:

- **Bandwidth incompatibilities:** While SRS supports operation with all the bandwidth settings specified by 3GPP, i.e., 1.4, 3, 5, 10, 15, and 20 MHz, OAI does not work with the 1.4, 3 and 15 MHz configurations. In addition, we find that the srsENB implementation (at least the Sept. 2017 version that we tested) does not work reliably with 20MHz channels. Therefore interoperability between eNBs and UEs running different stacks is limited to only two bandwidth settings, i.e., 5 and 10MHz.
- **Interconnection with CN:** the srsENB implementation employs the same subnetwork for both user plane (S1-U interface) and control plane (S1-C interface). On the other hand, the OpenAir-CN can be configured to use two different subnetworks in order to distinguish between the two planes; if such configuration is enabled, the srsENB stack will not work.
- **Problematic queue management:** We note that sending traffic in the downlink direction, at a rate that exceeds the maximum throughput supported on the channel, makes OAI crash. Following code inspection, we find that the different threads composing the software do not implement any packing dropping strategy at the queues/lists used for communication, which leads to out-of-memory issues. We have fixed this bug and are proposing a patch to the OAI developers community.

2.2. Hardware Accelerated SDR solutions

We have analyzed two full software implementations of the LTE stack that include the functionality of the eNB and UE. These software frameworks are designed and developed to be executed on general purpose CPUs allowing the virtualization of the whole stack. As we have seen in the previous section, OAI and srsLTE run in the host PC. We next present a SDR prototype

with hardware acceleration, empowered using a FPGA board. This prototype is an engineering challenge since the implementation of the functionalities in Hardware (HW) is not always an easy task. Often it requires an additional effort at the moment of debugging the code but with benefits in terms on the computational tasks: these are much faster when comparing to full software solutions. In this section, we introduce an alternative platform for LTE experimentation employing a FPGA-based solution with energy-awareness design. The results obtained with this platform are shown in Chapter 3. Additionally, we validate the energy-awareness design characterizing this prototype and studying the energy efficiency for the Wireless Communication Parameters (WCP) and configurations.

2.2.1. System description

As described in [60,61], there are many different function partitioning possibilities. However, our work considers three specially relevant Network Configurations (NETCFG), as shown in Figure 2.6, each one presenting a different degree of communication function offloading:

1. The first considered partition has all L1 functions executed locally at the HeNB, whereas all higher layer related processing is offloaded onto the Cloud. From this point onwards, this particular function split will be referred to as *NETCFG1*.
2. As in the previous case, the PHY-layer is executed locally at the HeNB. Nevertheless, in this second configuration, or *NETCFG2*, a MEC-like approach is adopted. That is, the remaining protocol stack functions will be virtualized as a specialized type of application in a server located in the vicinity of the small cell (i.e., MEC-like node).
3. *NETCFG3* considers the typical C-RAN setup, where the HeNB will act as a Remote Radio Head (RRH), while all protocol stack functions are placed onto the Cloud.

It is important to underline that, while the main focus of the proposed HeNB reconfigurations is set on the energy efficiency, this flexibility could also be exploited to satisfy a wide range of KPIs in different operative 5G scenarios (e.g., latency, availability or performance among others). The prototype combines a fully SW-based LTE emulator, implementing L2 and above stack functions, with a HWA DL PHY-layer based on a custom HW description language (HDL) realization¹⁵. This enables both the real-time operation and run-time reconfiguration of the hotspot.

2.2.2. System architecture

A real-time reconfigurable hotspot prototype has been developed with the objective to facilitate a realistic yet simple validation of the proposed concept described in the previous section. To make this possible, the prototype flexibly combines HWA and SW communication functions, by using standard computers and HW components. Regarding the implementation of HWA blocks,

¹⁵The uplink implementation is kept in the SW domain, making use of the native LENA functionalities.

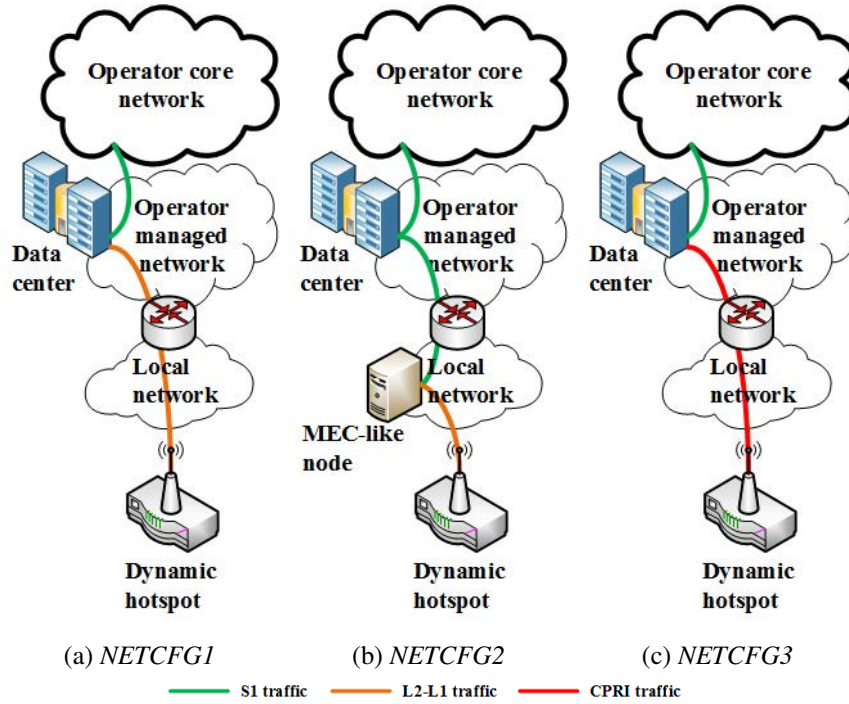


Figure 2.6: Considered NETCFGs and their related traffic loads.

the target platform is an FPGA-based SoC device: The latter embeds an integrated processing system (PS) and programmable logic (PL) on a single die, providing likewise high flexibility (i.e., run-time reconfigurability) and computational capacity (i.e., massive parallelism).

Focusing on the goal of obtaining an empirical assessment of the energy savings attainable by reconfiguring the system, the prototype has been kept as simple as possible. Hence, in the presented work the demonstrator includes a single LTE-based HeNB, which uses different HW elements to execute its functions, depending on the selected NETCFG, and a single UE¹⁶. Moreover the required evolved packet core (EPC) functionalities are also included, jointly with an emulated Internet, to complete the system implementation.

The presented scenario assumes that high-speed communication links are available where required, in order to communicate the small cell with the Cloud and/or MEC-like nodes. As it can be observed in Figure 2.6, depending on the adopted NETCFG, the communication links between the different 5G nodes present quite diverse latency and data-rate requirements [62]. In more detail, the interconnection of the HeNB to the Cloud or to the MEC-like node has stringent traffic requirements, which can be fulfilled using an ideal transport channel (i.e., 250 μ s of one-way latency and 2.5 Gbps for supporting a standardized LTE 2x2 20 MHz scheme) and the CPRI specification. The interconnection of the MAC with the PHY-layer (i.e., L2-L1 interface) poses more relaxed needs, that can be satisfied with sub-ideal transport channels (i.e., 6 ms of one-way

¹⁶Given that the current HeNB implementation supports multi-user transmissions, adding more UEs to the prototype only requires replicating its related HW setup.

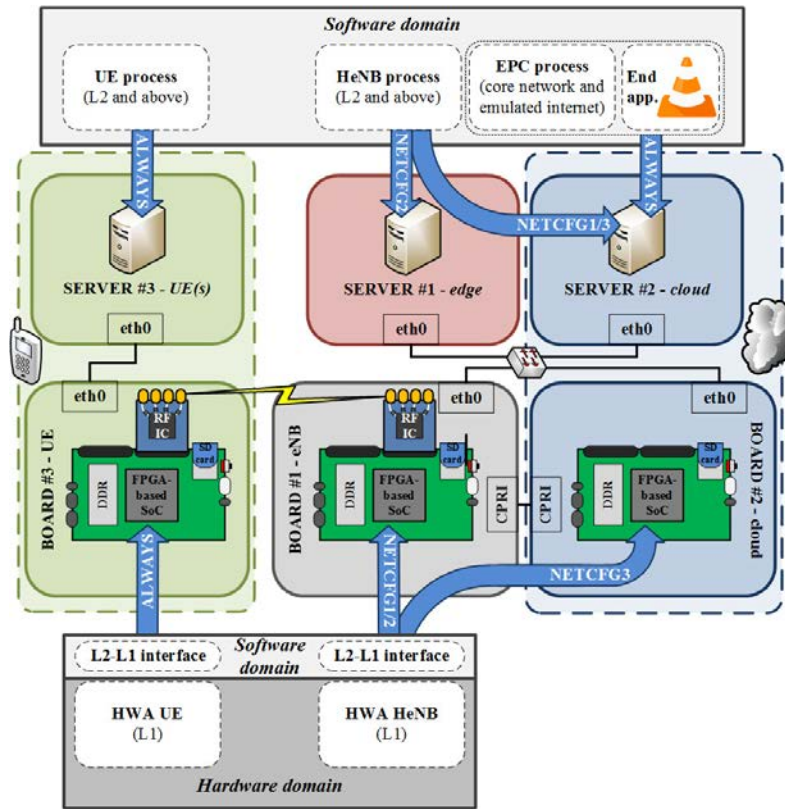


Figure 2.7: Overview of the HW setup implementing the hotspot prototype, including the different supported communication function splits.

latency and a capacity of 150 Mbps). Finally, the interconnection to the EPC (i.e., S1 traffic) is the less demanding and can be satisfied with a non-ideal transport channel (i.e., up to 30 ms of one-way latency and variable BW).

2.2.3. Dynamic hotspot prototype

A schematic of the basic HW setup of the energy-aware dynamic hotspot prototype is provided in Figure 2.7. Its flexible real-time operation supports the typical performance requirements described by the LTE standard, as well as the reconfiguration of the system to adopt different NETCFGs and/or wireless communication parameters (e.g., DL BW). The following section details the different HW components that are hosting the presented HWA and SW functions. Even though the focus of the described work resides in the adaptive HeNB prototype, its UE counterpart functions were also developed and included in the HW demonstrator in order to enable the end-to-end operation of the system under test. In this regard, the design and implementation details of the receiver subsystems are out of the scope of this work.

2.2.3.1. Hardware components

EXTREME[®] testbed

The EXTREME[®] Testbed is used to host the different SW processes of LENA (i.e. HeNB, UE and EPC). The EXTREME[®] Testbed [63] is an experimental framework for testing wireless access and backhaul/fronthaul architectures featuring generic purpose server pools (e.g., SDN control, NFVs such as vEPC), cellular and other wireless equipment, ns-3 emulation/simulation, and tools for fast prototyping and evaluation. Here, we only describe the components directly related to the hotspot prototype. The core of the EXTREME[®] Testbed lies on two central management servers, which act as interface between the final users and the SDN/NFV experimentation services. A series of reconfigurable multi-purpose servers and high-performance laptops can be customized and used as network elements for experimentation purposes. In this section, Super-micro servers have been the preferred option to execute the distributed LENA processes. These servers are equipped with two Intel Xeon E5-2640v4 processors (20 cores/40 threads running at 2.4 GHz), 64 GB of RAM, a hard disk of 2 TB and 6 Gigabit Ethernet (GigE) ports. Finally, the Cisco C6513 switch router is used to manage the GiGE connections and is dynamically reconfigured according to the adopted scenario to build the required fronthaul and backhaul networks.

FPGA-based SoC and RF boards

The Xilinx ZC706 board is used to host all HWA functions, as well as the related L2-L1 SW interfaces (for both HeNB and UE sides). In more detail, the board features the Zynq XC7Z045 all-programmable SoC, which integrates a dual-core ARM Cortex-A9 central processing unit (CPU; up to 1 GHz) on the PS side. These are paired with internal memory resources, a dedicated high-speed AXI-based bus and DMA interfaces to the PL, as well as with a set of standard input/output interfaces and peripherals (including Ethernet). Regarding the FPGA resources, the SoC also provides 350K logic cells, 545 embedded RAM blocks (RAMB) of 36Kb and 900 DSP slices (up to 18x25 multiply-and-accumulates each). The resource utilization metrics of the implemented HWA L1 functions can be observed in Table 2.1. It should be noted, that these metrics do not account for the HDL firmware utilized by the ZC706 board (i.e., they only account for the designed DSP functionality).

The Analog Devices (AD) AD-FMCOMMS3 board was used as the radio frequency (RF) front-end. It is connected to the ZC706 board through a FPGA Mezzanine Card (FMC) interface and includes the AD9361 RF integrated chip (RFIC). The latter features a 2x2 transceiver with integrated 12-bit DACs and analog-to-digital converters (ADCs), supporting a wide range of frequencies (up to 6 GHz), a tunable channel BW (up to 56 MHz) and programmable gain-control chains. Toward that end, a Linux kernel space application residing in the PS of the Zynq SoC, allows to fully tune and program the AD9361 RFIC. Optionally, power amplifiers, RF band filters and antennas are enabling over-the-air communications. The remaining implementation details can be found in Appendix A. In the next chapter, we will characterize the energy consumption of

Utilization	HWA eNB functions	HWA UE functions
LUT	22.30%	34.52%
LUTRAM	2.38%	13.38%
FF	9.57%	26.88%
BRAM	8.53%	61.65%
DSP	2.67%	60.78%
IO	28.73%	19.61%
BUFG	9.38%	21.88%
MMCM	0%	37.50%

Table 2.1: FPGA-resource utilization of the implemented HWA L1 entities.

this platform, considering the functional splits presented and hence, measuring and quantifying the energy footprint in each deployment.



Figure 2.8: Picture of the FPGA based testbed

2.2.3.2. Testbed Description

A different number of ZC706 (and AD-FMCOMMS3) boards is used depending on the specific NETCFG that is adopted, as it can be observed in Figure 2.7. When a C-RAN architecture is adopted, two FPGA-based platforms are required on the Home eNB (HeNB) side (i.e., one for the RRH and another one for the HWA L1), interfaced through a CPRI link (i.e., using coaxial cables as our ideal transport channel), and a third one is used to implement the HWA L1 of the UE. On

the contrary, when the L1 functions are executed locally at the HeNB (*NETCFG1/2*), then only two ZC706 boards are used (i.e., one for the HWA HeNB and another for its UE counterpart). In all cases, GigE links are utilized to provide the required interconnections between the DL signal and the EXTREME[®] testbed (i.e., sub-ideal and non-ideal transport channels) as shown in Fig. 2.8

2.3. Conclusions

This first contribution reports a performance assessment of the two most prevalent open source software solutions for mobile network prototyping, namely, srsLTE and OAI. We designed a methodology to characterize the performance of these stacks, quantifying their differences in throughput and resource consumption over a range of practical settings. Our findings formalize “word of mouth” knowledge among practitioners, and provide useful guidelines for deploying 5G testbeds with these tools.

Additionally, we have introduced a FPGA platform and prototype for future 5G connectivity with flexibility and reconfigurability as the two principal requirements. This platform supports centralized processing in a multi-layered Cloud architecture (i.e., C-RAN, MEC) and the massive deployment of small cells will play a central role toward providing a dynamic network management with energy-awareness capabilities.

Chapter 3

Analysis and Characterisation of the Energy Consumption

In the previous chapter, we have studied several softwarized platforms for mobile networking, presenting a methodology to characterize the performance of such platforms. Furthermore, we have presented a prototype with energy-awareness design. In this chapter, our objective is to analyze the energy efficiency in *short time scale*¹, introducing an energy measurement platform. This framework for energy characterization is validated considering the case of IEEE 802.11, using as a basis a analytic model and comparing it to experimental measurements. We next study the case of FPGA based prototype, analyzing the different subsystem components that form the platform and characterizing the energy impact for a set of configuration.

3.1. Platform and Instrumentation for Energy Measurements

The core of this chapter is based on the instrumentation platform for *ad-hoc* energy measurements. For such instrumentation, it is worth mentioning some features must be met to better understand how far we can go in our findings: *i*) the range of the measurements must go from mW up to tens of W, providing the flexibility of different levels of granularity according to the device to be dissected and, *ii*) a high sampling rate of measurements in time domain in order to analyze small energy changes between events as explained in [64]. This set of features will allow us to develop a methodology in order to analyze and characterize two different technologies with multiple levels of accuracy, whether we are characterizing the Modulation and Coding Scheme (MCS) adaptation for IEEE 802.11 Network Interface Card (NIC)s or the different subsystems of the LTE prototype presented in chapter 2: CPU, Ethernet NIC and the Radio Frequency Integrated Circuit (RFIC).

For these purposes, we chose the high-accuracy multifunction data acquisition (DAQ) device from National Instruments, more specifically the PCI-6289 model that includes up to 32 analogue

¹Let us define *short temporal scale* as measurements in a range of μs .

inputs and 7 input ranges with an accuracy of 18-bits. Each measurement is reported with a resolution of 50 ns, delivering an accuracy of 50 ppm of sampling rate.

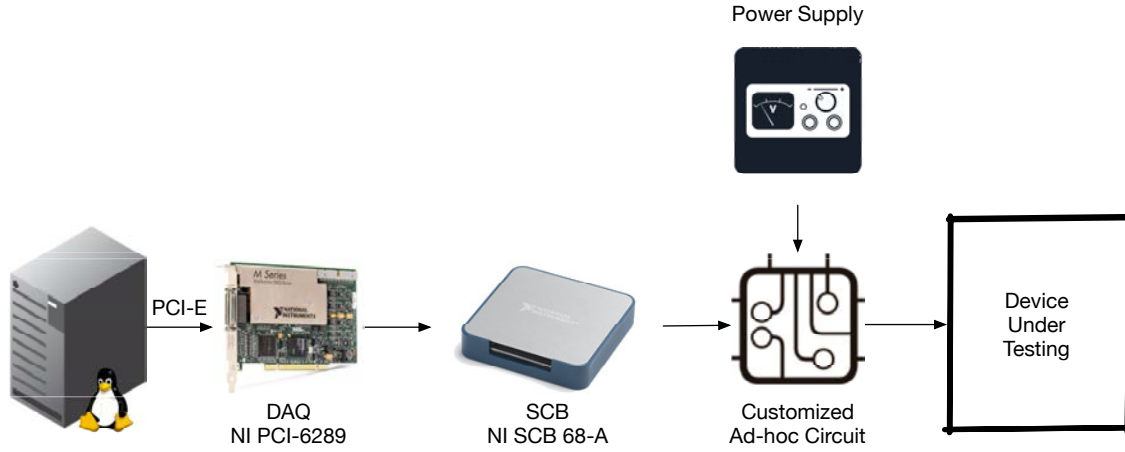


Figure 3.1: Simplified energy measurement platform

In addition to this equipment, the university’s Technical Office in Electronics designed a custom three-port circuit that samples the input signal and converts it to voltage, adapting the signal to the DAQ’s input minimizing the precision loss during the process. Considering two different voltage range specifications, the design was split in two different circuits. The full schematics can be consulted in Appendix B. For the sake of clarity, Fig 3.1 shows a simplified version of the energy measurement platform.

The software that empowers and brings of our platform to life is DAQ-Acquire² built on top of comedilib and Comedi drivers. As operating system, we have chosen Fedora GNU/Linux distribution.

3.2. Energy Impact on the Rate Adaptation for IEEE 802.11

This section tackles the problem of Rate Adaptation (RA) in 802.11 WLANs from the energy consumption’s perspective. RA algorithms are responsible for selecting the most appropriate modulation and coding scheme (MCS) and transmission power (TXP) to use, given an estimation of the link conditions, and have received a vast amount of attention from the research community (see e.g. [65] and references therein). In general, the challenge lies in distinguishing between those losses due to collisions and those due to poor radio conditions, because they should trigger different reactions. In addition, the performance figure to optimise is commonly the throughput or a related one such as, e.g., the time required to deliver a frame.

Building on this model, we provide the following contributions: *i*) we demonstrate through an extensive numerical evaluation that energy consumption and throughput performance are different optimization objectives in 802.11, and not only an effect of MIMO or certain algorithms’

²<https://github.com/Enchufa2/daq-acquire>

suboptimality; *ii*) we analyze the relative impact of each energy consumption component on the resulting performance of RA, which serves to identify the critical factors to consider for the design of RA algorithms, and illustrate that different hardware should employ different configurations; and *iii*) we experimentally validate our numerical results.

3.2.1. Joint Goodput-Energy Model

We develop a joint goodput-energy model for a single 802.11 spatial stream and the absence of interfering traffic. It is based on previous studies about goodput and energy consumption of wireless devices. As stated in the introduction, the aim of this model is the isolation of the relevant variables (MCS and TXP) to let us delve in the relationship between goodput and energy consumption optimality in the absence of other effects such as collisions or MIMO.

Beyond this primary intent, it is worth noting that these assumptions conform with real-world scenarios in the scope of recent trends in the IEEE 802.11 standard development, namely, the amendments 11ac and 11ad, where device-to-device communications (mainly through beamforming and MU-MIMO) are of paramount importance.

3.2.1.1. Goodput Model

We base our study on the work by Qiao *et al.* [66], which develops a robust goodput model that meets the established requirements. This model analyzes the IEEE 802.11a Distributed Coordination Function (DCF) over the assumption of an AWGN (Additive White Gaussian Noise) channel without interfering traffic.

Let us briefly introduce the reader to the main concepts, essential to our analysis, of the goodput model by Qiao *et al.*. Given a packet of length l ready to be sent, a frame retry limit $n_{\max}$ and a set of channel conditions $\hat{s} = \{s_1, \dots, s_{n_{\max}}\}$ and modulations $\hat{m} = \{m_1, \dots, m_{n_{\max}}\}$ used during the potential transmission attempts, the *expected effective goodput* $\mathcal{G}$ is modelled as the ratio between the expected delivered data payload and the expected transmission time as follows:

$$\mathcal{G}(l, \hat{s}, \hat{m}) = \frac{\mathbb{E}[\text{data}]}{\mathbb{E}[\mathcal{D}_{\text{data}}]} = \frac{\Pr[\text{succ} \mid l, \hat{s}, \hat{m}] \cdot l}{\mathbb{E}[\mathcal{D}_{\text{data}}]} \quad (3.1)$$

where $\Pr[\text{succ} \mid l, \hat{s}, \hat{m}]$ is the probability of successful transmission conditioned to $l, \hat{s}, \hat{m}$, given by Equation (5) in [66]. The expected transmission time is defined as follows:

$$\mathbb{E}[\mathcal{D}_{\text{data}}] = (1 - \Pr[\text{succ} \mid l, \hat{s}, \hat{m}]) \cdot \mathcal{D}_{\text{fail} \mid l, \hat{s}, \hat{m}} + \Pr[\text{succ} \mid l, \hat{s}, \hat{m}] \cdot \mathcal{D}_{\text{succ} \mid l, \hat{s}, \hat{m}}$$

where

$$\begin{aligned} \mathcal{D}_{\text{succ}|l,\hat{s},\hat{m}} = & \sum_{n=1}^{n_{\max}} \Pr[n \text{ succ} \mid l, \hat{s}, \hat{m}] \cdot \left\{ \sum_{i=2}^{n_{\max}} [\bar{T}_{\text{bkoff}}(i) \right. \\ & + T_{\text{data}}(l, m_i) + \bar{\mathcal{D}}_{\text{wait}}(i)] \\ & + \bar{T}_{\text{bkoff}}(1) + T_{\text{data}}(l, m_1) + T_{\text{SIFS}} \\ & \left. + T_{\text{ACK}}(m'_n) + T_{\text{DIFS}} \right\} \end{aligned} \quad (3.2)$$

is the average duration of a successful transmission and

$$\begin{aligned} \mathcal{D}_{\text{fail}|l,\hat{s},\hat{m}} = & \sum_{i=1}^{n_{\max}} [\bar{T}_{\text{bkoff}}(i) \\ & + T_{\text{data}}(l, m_i) + \bar{\mathcal{D}}_{\text{wait}}(i+1)] \end{aligned} \quad (3.3)$$

is the average time wasted during the $n_{\max}$ attempts when the transmission fails.

$\Pr[n \text{ succ} \mid l, \hat{s}, \hat{m}]$ is the probability of successful transmission at the n -th attempt conditioned to $l, \hat{s}, \hat{m}$, and $\bar{\mathcal{D}}_{\text{wait}}(i)$ is the average waiting time before the i -th attempt. Their expressions are given by Equations (7) and (8) in [66]. The transmission time (T_{data}), ACK time (T_{ACK}) and average backoff time ($\bar{T}_{\text{bkoff}}$) are given by Equations (1), (2) and (3) in [66]. Finally, T_{SIFS} and T_{DIFS} are 802.11a parameters, and they can be found also in Table 2 in [66].

3.2.1.2. Energy Consumption Model

The selected energy model is our previous work of [67], which has been further validated via ad-hoc circuitry and specialised hardware [68] and, to the best of our knowledge, stands as the most accurate energy model for 802.11 devices published so far, because it accounts not only the energy consumed by the wireless card, but the consumption of the whole device. While classical models focused on the wireless interface solely, this one demonstrates empirically that the energy consumed by the device itself cannot be neglected as a device-dependent constant contribution. Conversely, devices incur an energy cost derived from the frame processing, which may impact the relationship that we want to evaluate in this Chapter.

This model can be summarised as follows:

$$\bar{P} = \rho_{\text{id}} + \sum_{i \in \{\text{tx}, \text{rx}\}} \rho_i \tau_i + \sum_{i \in \{\text{g}, \text{r}\}} \gamma_{\text{xi}} \lambda_i \quad (3.4)$$

where $\rho_{\text{id}}, \rho_{\text{tx}}, \rho_{\text{rx}}$ are the power consumed by the device in idle, transmission and reception states respectively; $\tau_{\text{tx}}, \tau_{\text{rx}}$ are the airtime percentages in transmission and reception; $\gamma_{\text{xg}}, \gamma_{\text{xr}}$ are the so called *cross-factors*, a per-frame energy toll for generation and reception respectively; and $\lambda_{\text{g}}, \lambda_{\text{r}}$ are the frame generation and reception rates.

Therefore, the average power consumed $\bar{P}$ is a function of five device-dependent parameters (ρ_i, γ_{xi}) and four traffic-dependent ones (τ_i, λ_i) .

3.2.1.3. Energy Efficiency Analysis

Putting together both models, we are now in a position to build a joint goodput-energy model for 802.11a DCF. Let us consider the average durations (3.2) and (3.3). Based on their expressions, we multiply the idle time $(\bar{\mathcal{D}}_{\text{wait}}, \bar{T}_{\text{bkoff}}, T_{\text{SIFS}}, T_{\text{DIFS}})$ by ρ_{id} , the transmission time (T_{data}) by ρ_{tx} , and the reception time (T_{ACK}) by ρ_{rx} . The resulting expressions are the average energy consumed in a successful transmission $\mathcal{E}_{\text{succ}|l, \hat{s}, \hat{m}}$ and the average energy wasted when a transmission fails $\mathcal{E}_{\text{fail}|l, \hat{s}, \hat{m}}$:

$$\begin{aligned} \mathcal{E}_{\text{succ}|l, \hat{s}, \hat{m}} = & \sum_{n=1}^{n_{\text{max}}} \Pr[n \text{ succ} | l, \hat{s}, \hat{m}] \cdot \left\{ \sum_{i=2}^{n_{\text{max}}} [\rho_{\text{id}} \bar{T}_{\text{bkoff}}(i) \right. \\ & + \rho_{\text{tx}} T_{\text{data}}(l, m_i) + \rho_{\text{id}} \bar{\mathcal{D}}_{\text{wait}}(i)] \\ & + \rho_{\text{id}} \bar{T}_{\text{bkoff}}(1) + \rho_{\text{tx}} T_{\text{data}}(l, m_1) + \rho_{\text{id}} T_{\text{SIFS}} \\ & \left. + \rho_{\text{rx}} T_{\text{ACK}}(m'_n) + \rho_{\text{id}} T_{\text{DIFS}} \right\} \end{aligned} \quad (3.5)$$

$$\begin{aligned} \mathcal{E}_{\text{fail}|l, \hat{s}, \hat{m}} = & \sum_{i=1}^{n_{\text{max}}} [\rho_{\text{id}} \bar{T}_{\text{bkoff}}(i) \\ & + \rho_{\text{tx}} T_{\text{data}}(l, m_i) + \rho_{\text{id}} \bar{\mathcal{D}}_{\text{wait}}(i+1)] \end{aligned} \quad (3.6)$$

Then, by analogy with (3.2), the *expected energy consumed per frame transmitted*, $\mathbb{E}[\mathcal{E}_{\text{data}}]$, can be written as follows:

$$\begin{aligned} \mathbb{E}[\mathcal{E}_{\text{data}}] = & \gamma_{\text{xg}} + (1 - \Pr[\text{succ} | l, \hat{s}, \hat{m}]) \cdot \mathcal{E}_{\text{fail}|l, \hat{s}, \hat{m}} \\ & + \Pr[\text{succ} | l, \hat{s}, \hat{m}] \cdot \mathcal{E}_{\text{succ}|l, \hat{s}, \hat{m}} \end{aligned} \quad (3.7)$$

It is noteworthy that the receiving cross-factor does not appear in this expression because ACKs (acknowledgements) are processed in the network card exclusively, and thus its processing toll is negligible.

Finally, we define the *expected effective energy efficiency* μ as the ratio between the expected delivered data payload and the expected energy consumed per frame, which can be expressed in *bits per Joule* (bpJ):

$$\mu(l, \hat{s}, \hat{m}) = \frac{\mathbb{E}[\text{data}]}{\mathbb{E}[\mathcal{E}_{\text{data}}]} \quad (3.8)$$

3.2.2. Numerical Results

Building on the joint model presented in the previous section, here we explore the relationship between optimal goodput and energy efficiency in 802.11a. More specifically, our objective is to understand the behaviour of the energy efficiency in a range of devices in order to compare our experimental measurements with the analytic model and hence, validating our platform.

3.2.2.1. Optimal Goodput

We note that the main goal of RA, generally, is to maximise the effective goodput that a station can achieve by varying the parameters of the interface. In terms of the model discussed in the previous section, a rate adaptation algorithm would aspire to fit the following curve:

$$\max \mathcal{G}(l, \hat{s}, \hat{m}) \quad (3.9)$$

We provide the numerical results for this goodput maximisation problem in Fig. 3.2, which are in good agreement with those obtained in [66]. For the sake of simplicity but without loss of generality we fix $l = 1500$ octets and $n_{\max} = 7$ retries, and assume that the channel conditions and the transmission strategy are constant across retries ($\hat{s} = \{s_1, \dots, s_1\}$ and $\hat{m} = \{m_1, \dots, m_1\}$).

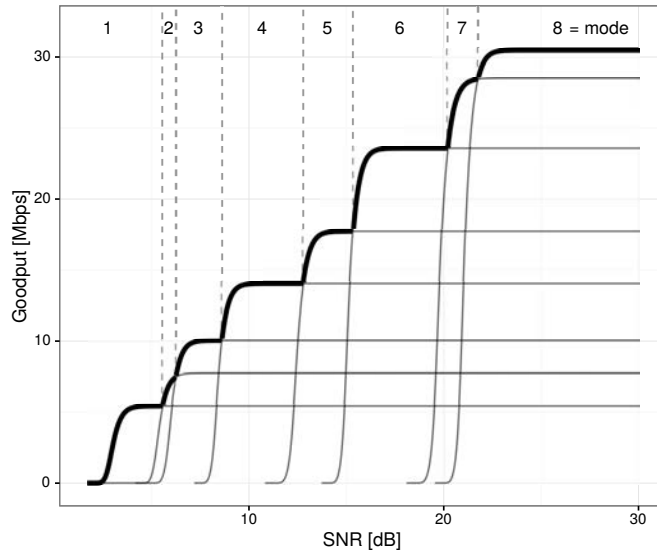


Figure 3.2: Optimal goodput (bold envelope) as a function of SNR.

Fig. 3.2 illustrates which mode (see Table 3.1) is optimal in terms of goodput, given an SNR level. We next address the question of whether this optimization is aligned with energy efficiency maximisation.

Table 3.1: Modes of the IEEE 802.11a PHY

Mode Index	1	2	3	4	5	6	7	8
MCS (Mbps)	6	9	12	18	24	36	48	54

3.2.2.2. Extension of the Energy Parametrisation

The next step is to delve into the energy consumption of wireless devices. [67] provides real measurements for five devices: three AP-like platforms (Linksys WRT54G, Raspberry Pi and Soekris net4826-48) and two hand-held devices (HTC Legend and Samsung Galaxy Note 10.1). Two of the four parameters needed are constant (ρ_{id} , γ_{xg}), and the other two (ρ_{tx} , ρ_{rx}) depend on the MCS and the TXP used. However, the characterisation done in [67] is performed for a subset of the MCS and TXP available, so we next detail how we extend the model to account for a larger set of operation parameters.

A detailed analysis of the numerical figures presented in [67] suggests that ρ_{tx} depends linearly on the MCS, and that ρ_{tx} depends linearly on the MCS and the TXP (in mW). Based on these observations, we define the following linear models:

$$\rho_{\text{tx}} = \alpha_0 + \alpha_1 \cdot \text{MCS} + \alpha_2 \cdot \text{TXP} \quad (3.10)$$

$$\rho_{\text{rx}} = \beta_0 + \beta_1 \cdot \text{MCS} \quad (3.11)$$

The models are fed with the data reported in [67], the results are shown in Appendix C.

3.2.3. Energy Consumption

To compute the energy consumption using the above parametrisation, first we have to define the assumptions for the considered scenario. We assume for simplicity a device-to-device communication, with fixed and reciprocal channel conditions during a sufficient period of time (i.e., low or no mobility). As we have discussed before, our primary goal is to isolate MCS and TXP as variables of interest, but we must not forget that these are also reasonable assumptions in scenarios targeted by recent 802.11 standard developments (11ac, 11ad).

For instance, given channel state information from a receiver, the transmitter may decide to increase the TXP in order to increase the receiver's SNR (and thus the expected goodput), or to decrease it if the channel quality is high enough. Although the actual relationship between TXP and SNR depends on the specific channel model (e.g., distance, obstacles, noise), without loss of generality, we choose a noise floor of $N = -85$ dBm in an office scenario with a link distance of $d = 18$ m in order to explore numerically the whole range of SNR while using reasonable values of TXP. The ITU model for indoor attenuation [69] gives a path loss of $L \approx 85$ dBm. Then, we can use (3.7) to obtain the expected energy consumed per frame and MCS mode, with TXP being directly related to the SNR level.

The results are reported in Fig. C.2. As the figure illustrates, consumption first falls abruptly

as the TXP increases for all modes, which is caused when the SNR reaches a sharp threshold level such that the number of retransmissions changes from 6 to 0 (i.e., no frame is discarded). From this threshold on, the consumption increases with TXP because, although the number of retransmissions is 0, the wireless interface consumes more power. We note that the actual value of the TXP when the consumption drops depends on the specifics of the scenario considered, but the qualitative conclusions hold for a variety of scenarios.

3.2.3.1. Energy Efficiency vs. Optimal Goodput

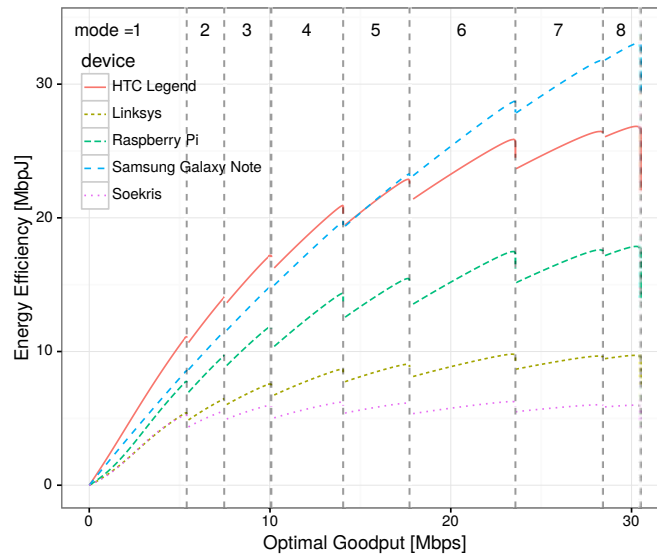


Figure 3.3: Energy Efficiency vs. Optimal Goodput under fixed channel conditions.

We can finally merge previous numerical analyses and confront energy efficiency, given by (3.8), and optimal goodput, given by (3.9), for all devices and under the aforementioned assumptions. To this aim, we plot in the same figure the energy efficiency for the configuration that maximises goodput given an SNR value vs. the obtained goodput, with the results being depicted in Fig. 3.3. We next discuss the main findings from the figure.

First of all, we can see that the energy efficiency grows sub-linearly with the optimal goodput (the optimal goodput for each SNR value) in all cases. We may distinguish three different cases in terms of energy efficiency: high (Samsung Galaxy Note and HTC Legend), medium (Raspberry Pi) and low energy efficiency (Linksys and Soekris). Furthermore, for the case of the Soekris, we note that the “central modes” (namely, 4 and 5) are more efficient in their optimal region than the subsequent ones.

Another finding (more relevant perhaps) is that it becomes evident that increasing the goodput does not always improve the energy efficiency: there are more or less drastic leaps, depending on the device, between mode transitions. From the transmitter point of view, in the described scenario, this can be read as follows: we may increase the TXP to increase the SNR, but if the

optimal goodput entails a mode transition, the energy efficiency may be affected.

As a conclusion, we have demonstrated that optimal goodput and energy efficiency do not go hand in hand, even in a single spatial stream, in 802.11. There is a trade-off in some circumstances that current rate adaptation algorithms cannot take into account, as they are oblivious to the energy consumption characteristic of the device.

3.3. Experimental Validation

This section is devoted to experimentally validate the results from the numerical analysis and, therefore, the resulting conclusions. To this aim, we describe our experimental setup and the validation procedure, first specifying the methodology and then the results achieved.

3.3.1. Experimental Setup

We deployed the testbed illustrated in Fig. 3.4, which consists of a station (STA) transmitting evenly-spaced maximum-sized UDP packets to an access point (AP). The AP is an x86-based Alix6f2 board with a Mini PCI Qualcomm Atheros AR9220 wireless network adapter, running Voyage Linux with kernel version 3.16.7 and the `ath9k` driver. The STA is a desktop PC with a Mini PCI Express Qualcomm Atheros QCA9880 wireless network adapter, running Fedora Linux 23 with kernel version 4.2.5 and the `ath10k` driver. We also installed at the STA a Mini PCI Qualcomm Atheros AR9220 wireless network adapter to monitor the wireless channel.

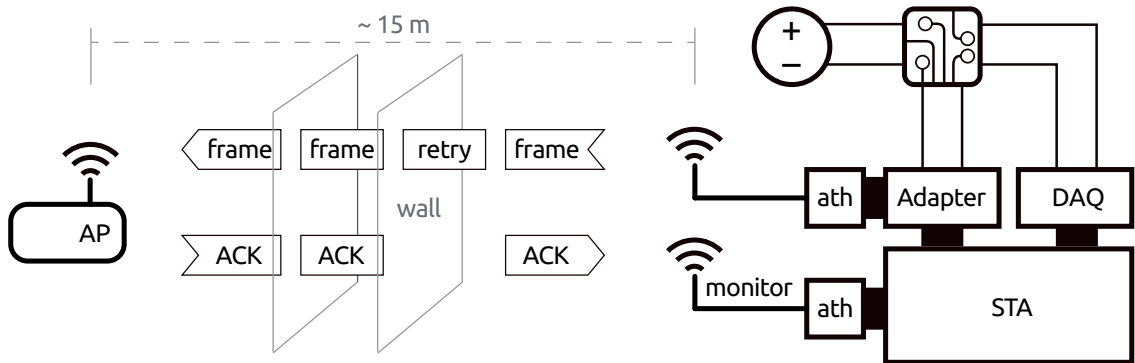


Figure 3.4: Experimental setup.

The QCA9880 card is connected to the PC through a *x1 PCI Express to Mini PCI Express* adapter from Amfeltec. This adapter connects the PCI bus' data channels to the host and provides an ATX port so that the wireless card can be supplied by an external power source. The power supply is a Keithley 2304A DC Power Supply, and it powers the wireless card through an *ad-hoc* measurement circuit that extracts the voltage and converts the current with a high-precision sensing resistor and amplifier. These signals are measured using a National Instruments PCI-6289 multifunction data acquisition (DAQ) device previously explained, which is also connected

to the STA. Thanks to this configuration, the STA can simultaneously measure the instant power consumed by the QCA9880 card³ and the goodput achieved.

As the figure illustrates, the STA is located in an office space and the AP is placed in a laboratory 15 m away, and transmitted frames have to transverse two thin brick walls. The wireless card uses only one antenna and a practically-empty channel in the 5-GHz band. Throughout the experiments, the STA is constantly backlogged with data to send to the AP, and measures the throughput obtained by counting the number of received acknowledgements (ACKs).

3.3.2. Methodology and Results

In order to validate our results, our aim is to replicate the qualitative behaviour of Fig. 3.3, in which there are energy efficiency “drops” as the optimal goodput increases. However, in our experimental setting, channel conditions are far from steady, which introduces a notable variability in the results as it affects both the x -axis (goodput) and the y -axis (energy efficiency). To reduce the impact of this variability, we decided to change the variable in the x -axis from the optimal goodput to the transmission power –a variable that is directly configured in the wireless card. In this way, the qualitative behaviour to replicate is the one illustrated in Fig. 3.5, where we can still identify the performance “drops” causing the loss in energy efficiency.

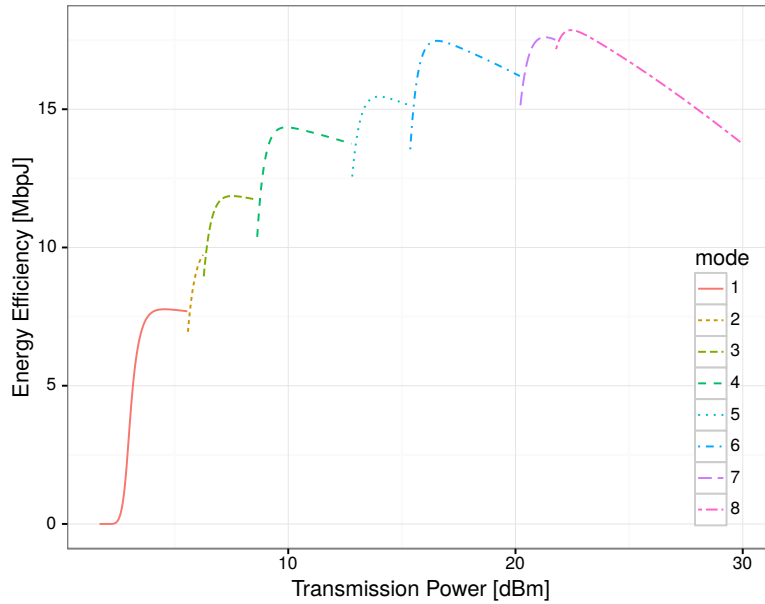


Figure 3.5: Energy Efficiency vs. Transmission Power under fixed channel conditions for the Raspberry Pi case.

On the other hand, experimental conditions are far from being ideal, and it is hard to achieve the channel steadiness required to get an adequate sweep along the optimal goodput values of

³Following the discussion on Section C, the device’s cross-factor is not involved in the trade-off, thus we will expect to reproduce it by measuring the wireless interface alone.

Fig. 3.3. The SNR's random variability causes that an ordered set of TXPs result in an unordered set of experimental goodputs, which derive in a figure impossible to interpret.

To solve this, we perform a change of the x axis to reflect the TXP instead of the optimal goodput, the result being illustrated in Fig. 3.5 for the case of the Raspberry Pi, which is easier to explore experimentally. A close look reveals that the downward slopes as the TXP grows for each mode (and their length until the next mode takes over) are responsible for the efficiency drops shown in Fig. 3.3. Therefore, our experimental validation is based on the presence of such indicators.

Building on this figure, we perform a sweep through all available combinations of MCS (see Table 3.1) and TXP⁴. Furthermore, we also tested two different configurations of the AP's TXP at different times of the day, to confirm that this qualitative behaviour is still present under different channel conditions. For each configuration, we performed 2-second experiments in which we measure the total bytes successfully sent and the energy consumed by the QCA9880 card with sub-microsecond precision. From this data, the energy efficiency is computed, with the results depicted in Fig. 3.6 (each figure corresponds to a different TXP value configured at the AP).

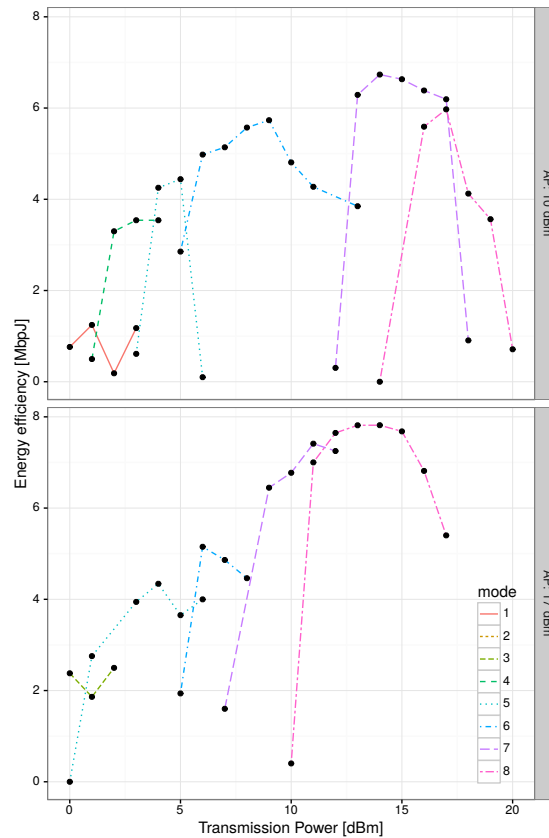


Figure 3.6: Experimental study of Fig. 3.5 for two AP configurations.

⁴The model explores a range between 0 and 30 dBm to get the big picture, but this particular wireless card only allows us to sweep from 0 to 20 dBm.

Our model assume reciprocal channel conditions, which means, among other things, that receiver's TXP (for the ACKs) should be the same as the transmitter's. In our setup, with different network cards on both sides, we cannot ensure this condition. So in order to overcome this limitation and remove this variable, we tested two different AP configurations: low and high TXP, 10 and 17 dBm respectively.

In the figure, each line type represents the STA's mode that achieved the highest goodput for each TXP interval, therefore in some cases low modes do not appear because a higher mode achieved a higher goodput. Despite the inherent experimental difficulties, namely, the low granularity of 1-dBm steps and the random variability of the channel, the experimental results validate the analytical ones, as the qualitative behaviour of both figures follows the one illustrated in Fig. 3.5. In particular, the performance "drops" of each dominant mode can be clearly observed (especially for the 36, 48 and 54 Mbps MCSs) despite the variability in the results.

3.4. Energy Consumption in a Energy-aware LTE Platform

The first part of this chapter presents a measurement platform for the energy characterization, framework that we have employed for an experimental validation of the proposed analytical model for the RA algorithms in IEEE 802.11. In this section, we reuse such framework to extend our analysis to the LTE prototype previously introduced in Chapter 2, dissecting the energy consumption across the different subsystems that form the prototype. For a better understanding of this section and the architecture design with energy-awareness, we refer the reader to the Appendix A.

The main objective of the development presented in the previous chapter is to allow the realistic evaluation of the gains that could be provided by applying energy-aware reconfigurations at the dynamic hotspot. For this purpose, a real-time prototype has been designed and implemented, where the communication functions can be moved among different 5G nodes in order to improve the energy efficiency of the system, accounting for the actual network conditions as well.

3.4.1. Testbed setup and measurement devices

In order to accurately assess the energy-saving benefits of the proposed dynamic reconfiguration of both Wireless Communication Parameters (WCP)s and function splits in Chapter 2, specialized HW has also been utilized in order to obtain experimental measurements at different key subsystem elements of the hotspot prototype. Namely the energy consumption of the HeNB has been analyzed at the System on a Chip (SoC) baseband processor (i.e., HWA L1), at the RF stage and at the GigE interfaces. Additionally, the central processor unit (CPU) load resulting from the different LENA configurations has also been monitored in order to (indirectly) assess its effect on the energy consumption of the Supermicro servers. In more detail, the servers provide an intelligent platform management interface (IPMI), which among others enables monitoring the usage of CPU and memory resources, as well as the overall server consumption. Nevertheless,

given the very low granularity of this embedded energy measurement HW⁵, the CPU loads resulting from different system configurations have been captured with the objective to complete the presented analysis.

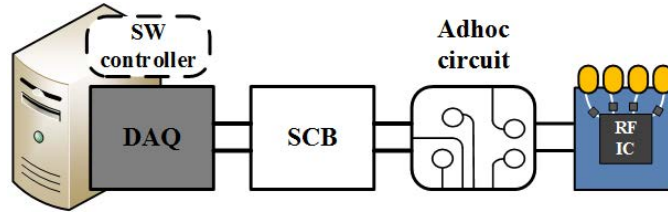


Figure 3.7: HW setup utilized to measure the energy consumption of the RF IC.

As for the HWA L1, the Xilinx ZC706 board hosts a power system based on the Texas Instruments (TI) UCD90120A power supply sequencer. The latter integrates a 12-bit ADC enabling to monitor up to 12 power-supply voltage lines. Moreover, a power management bus (PMBus) compliant controller is also included. By using a proprietary universal serial bus (USB)-based cable and the TI Fusion Digital Power Designer graphical user interface (GUI), the voltage and current utilized by the baseband SoC can be measured at run-time. In our case, the *VCCINT* rail has been used for the energy measurements, taking into account that it is the one powering both the internal logic of the PL and the PS.

Regarding the RFIC, a custom measurement setup was implemented as it can be observed in Figure 3.7 whilst the Figure 3.8 shows the real testbed deployment. As it was explained at beginning of this chapter, two ad-hoc measurement circuits were specially designed to work with the 3.3 V power rail of the AD9361 chip. A Shielded Connector Block (SCB), based on the National Instrument (NI) SCB-68A device, allows to interconnect the ad-hoc measurement circuits to a data acquisition (DAQ) card. Specifically, the NI PCI-6289 multifunction DAQ device is used to quantify the measurements. Toward that end, the DAQ is hosted in a general purpose computer through the peripheral component interconnect express (PCIe) bus. The `DAQ-acquire` SW application tool is then in charge of demultiplexing the measurements of the DAQ card and dumping the measured values onto a file, enabling its posterior post-processing. A similar setup was employed to obtain energy measurements at the GigE NIC of the server hosting the HeNB LENA processes.

3.5. Experimental results and discussion

This section presents the experimental evaluation of the energy-aware hotspot prototype under different function splits and WCP configurations. It must be noted that the aim of the presented results is to prove that important energetic benefits can be obtained by applying the proposed HeNB reconfigurations. In that sense, we are aware that the energy measurements detailed herein

⁵The IPMI measurement solution provided the compound power consumption of the Supermicro server (i.e., including not only the CPU, but the hard disk, fans and remaining HW components).

are strongly dependent on the specific HW elements comprising the prototype. Consequently, the focus of our evaluation is laid on the trade-offs and energy savings observed when comparing the power consumption of the prototype under different system configurations. From our past implementation experiences, we believe that similar results in terms of the presented figures (i.e., same order of magnitude) should be obtained when using different HW setups.

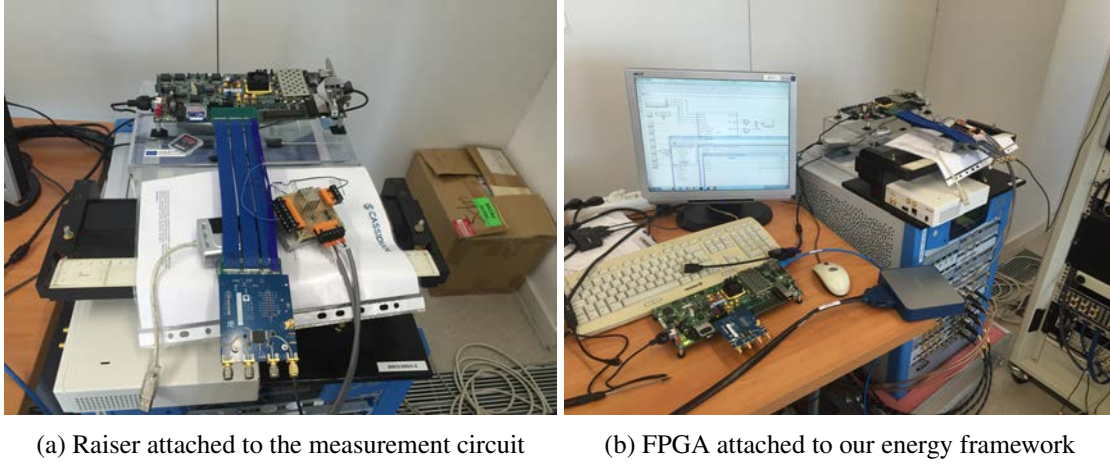


Figure 3.8: Picture of the testbed composed by the FPGA system and the energy framework

3.5.1. Methodology

An exhaustive measurement campaign has been carried out with the objective to characterize the energy savings reported by reconfiguring the dynamic hotspot. In that regard, a set of operating scenarios have been defined by modifying the applied NETCFG and the values of the WCPs. Namely, different DL BW configurations, RBG allocation loads, MCS indexes and RF transmit power settings are considered. For each given scenario a series of power measurements were then performed. In order to procure statistically significant data the time resolution of the DAQ card responsible for gathering the energy measurements was fixed at $1\ \mu\text{s}$ (i.e., 1 MHz sampling frequency). Moreover, the data has been captured in uninterrupted sequences of 30 seconds, with several repetitions per experiment. Similar values have been also considered for the measurement resolution of the baseband, in spite of its inferior specifications with respect to the time resolution of the samples when compared to the DAQ card.

In all experiments the HeNB has been configured and operating according to the defined scenario, and with the primary objective of setting the focus on a given reconfiguration parameter on each experiment. That is, only a single parameter has been modified at a time, whereas the remaining ones are kept fixed. This allows to observe the result of adapting that isolated parameter (e.g., DL BW) with respect to the power consumption of the HeNB. Considering the utilized measurement setup, 30 million samples have been obtained for each experimental iteration. Nevertheless, only selected sets of data have been used in order to bound the cost of the

post-processing, without compromising the validity of the analysis. Finally, the evaluation of the results is performed offline using custom scripts based on R, an open-source SW environment for statistical computing [70]. The obtained curves represent the cumulative distribution function (CDF) of the energy consumed by the HeNB under each implemented scenario.

3.5.2. Observed energy consumption results

The energy consumption of the hotspot prototype is analyzed for the different measured sub-systems, according to the results obtained after post-processing the captured data for the different experiments.

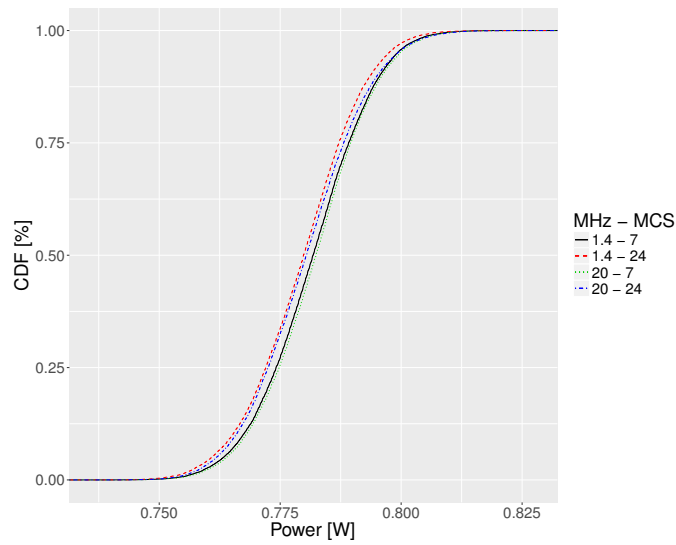


Figure 3.9: Power consumption observed at the GigE NIC for different HeNB configurations.

3.5.2.1. Ethernet

Several tests have been conducted in order to characterize the energy consumption of the GigE NICs of the servers⁶ hosting LENA. Each experiment utilized a different hotspot configuration. For instance, Figure 3.9 depicts the consumption observed at the server hosting L2 and above functionalities (HeNB side) for different DL BW and MCS index settings. The figure illustrates no variations in the consumed energy due to changes in the traffic load (i.e., MCS index, RBG allocation).

While this behaviour was expected for legacy Ethernet devices [71]], it is also observed when NICs are equipped with modern power saving features, such as those defined by the energy efficient Ethernet (EEE; which is the case of the Supermicro servers). The reason is that energy savings are linked to low and bursty data activity, which supports the use of aggregation techniques (i.e., coalescing) [72]. However, this is not the case for the dynamic hotspot traffic. First,

⁶A similar behaviour is expected on the local HeNB HW (e.g., Xilinx board), but it was less cumbersome measuring the NIC of a standard server.

the traffic load of the CPRI link is not low in those NETCFGs where L2 and above functions are offloaded onto the Cloud. Moreover, coalescing can introduce jitter, which might result in a disrupted system performance. Therefore, the contribution of the network interfacing HW elements to the energy budget of the dynamic hotspot can be hardly reduced and its energy-aware reconfigurations should target other subsystems.

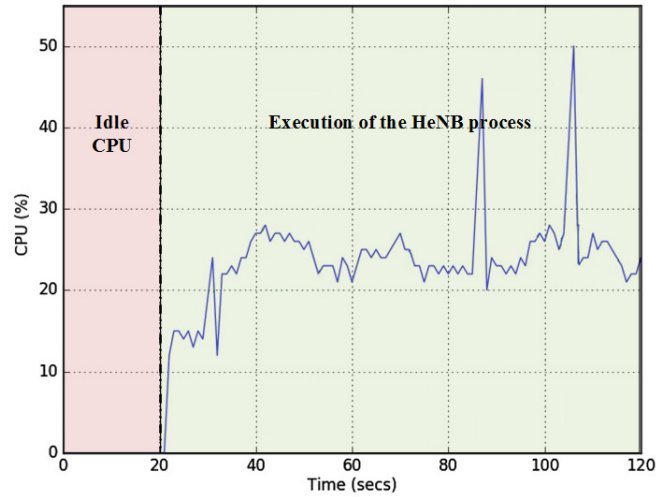


Figure 3.10: CPU load resulting from the LENA HeNB process when using a 10 MHz DL BW configuration (MCS index 25).

3.5.2.2. SW L2 (CPU)

Independently of the particular WCP settings being adopted (i.e., DL BW, MCS index and RBG allocation), the CPU utilization observed for the HeNB process (i.e., L2 and upper-layers) of LENA is always in the range between 25% and 30%, as shown in Figure 13. This is due to the massive computation capacity of the Supermicro servers (i.e., a single UE is attached to the HeNB; e.g., resulting in the simplest scheduling), as well as to the low measurement granularity provided by its monitoring interface. Accordingly, from the point of view of the PS, it seems that the only strategy to save energy in the HeNB would be to offload the processing of L2 and above functions, which is actually the case in all considered NETCFGs. In this respect, the energy consumption of a PS can be analyzed based on its activity. In more detail, an important fraction of its consumption is related to the usage of its CPU(s) and memories, including their related cooling systems. Regarding the processors, they present an elevated baseline energy consumption when in idle state (i.e., the idle energy consumption represents an important fraction of its peak consumption when at full load). This consumption increases with every added workload (i.e., process). On top of that, the drained energy depends on its operating frequency [73]. Moreover, the more accesses that these processes require to the memory system, the more energy it will be utilized. Based on that, it can be argued that minimizing the workload of a processor at any given time, will help reducing the overall consumed system energy. Even more, if the workload

is sufficiently low (i.e., low number of attached UEs), a reduced clock frequency could be used in the CPU, helping likewise to reduce its energy footprint [74]. This latter fact is empirically verified in the evaluation of the energy consumed by the HWA L1.

3.5.2.3. HWA L1 (FPGA-based SoC)

In the analysis of the power consumption observed at the Zynq XC7Z045 SoC it must be first noted that *NETCFG1* and *NETCFG2* are indistinguishable (i.e., both feature a locally executed HWA L1). Consequently only the power measurements for *NETCFG2* and *NETCFG3* are presented. In detail, the first three experiments assume a locally executed HWA L1, whereas the fourth examines the benefits of moving the DSP computation to the Cloud.

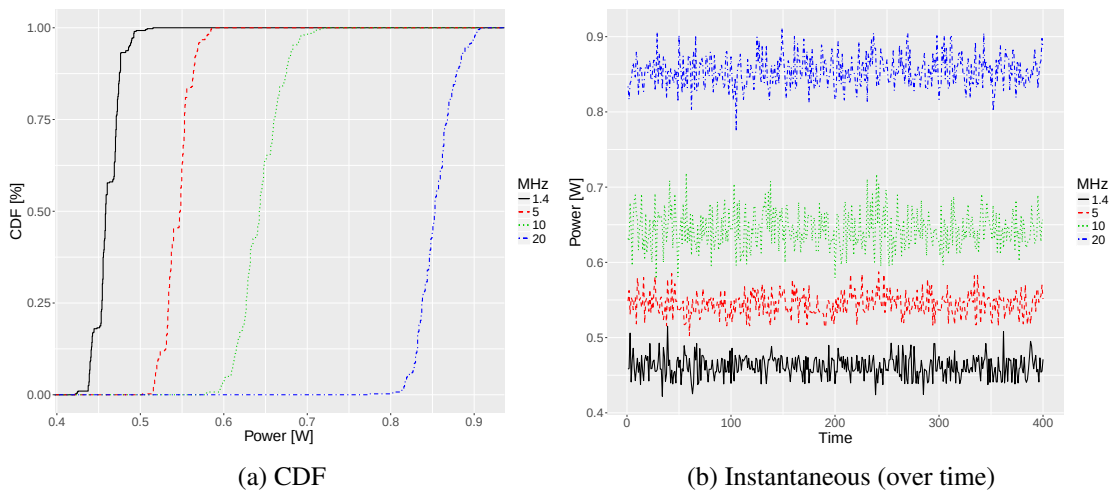


Figure 3.11: Impact of the DL BW configuration on the consumption of the baseband SoC.

The first experiment analyzes the variation of the power consumption at the baseband processor as a function of the utilized DL BW configuration. In that respect, a fixed MCS index of 24 (i.e., highest modulation order) and a fully allocated DLSCH (i.e., user-data in all available RBGs) is combined with the four considered DL BW values (i.e., 1.4, 5, 10 and 20 MHz). As it can be seen in Figure 3.11 downscaling the signal BW can provide important power savings, which scale up to 44% in the extreme situation of changing from 20 MHz to 1.4 MHz (e.g., end of the venue, when most attendees leave the hotspot coverage area). The experiments have shown that these gains remain stable over time (Figure 3.11b).

The impact that different RBG allocations have on the energy consumption of the SoC is evaluated on the second experiment for two different DL BW settings and MCS indexes. Three different RBG loads are considered, ranging from a very low utilization of the available DLSCH resources to a fully allocated case. Figure 3.12a shows a fixed baseline MCS configuration with index 7 (i.e., lowest modulation order) for the 10 and 20 MHz DL BW cases. Similarly, Figure 3.12b adopts a fixed MCS index of 24. In both cases minimal variations in the energy consumption of the HWA L1 are reported (i.e., around 2% in the best case).

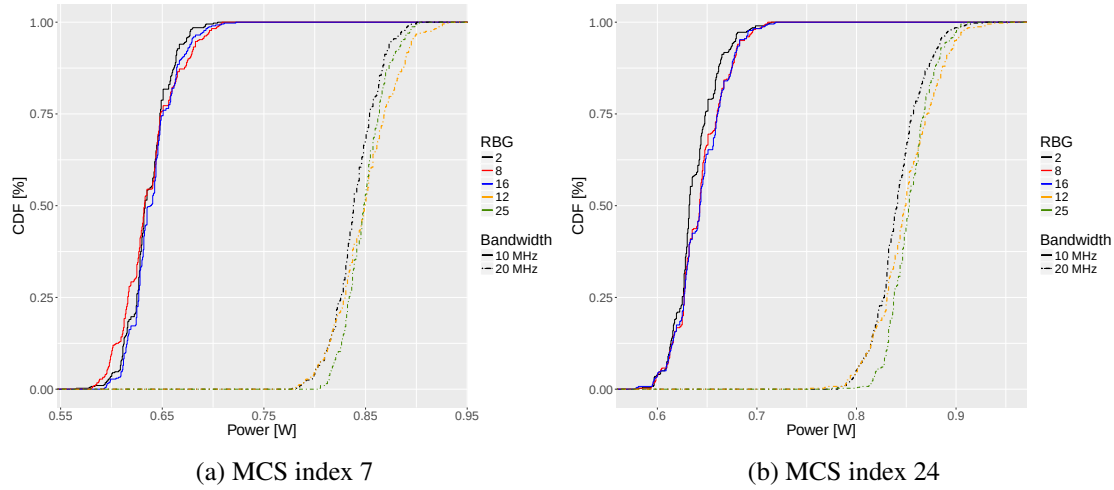


Figure 3.12: Impact of the RBG allocation on the energy consumption of the baseband SoC.

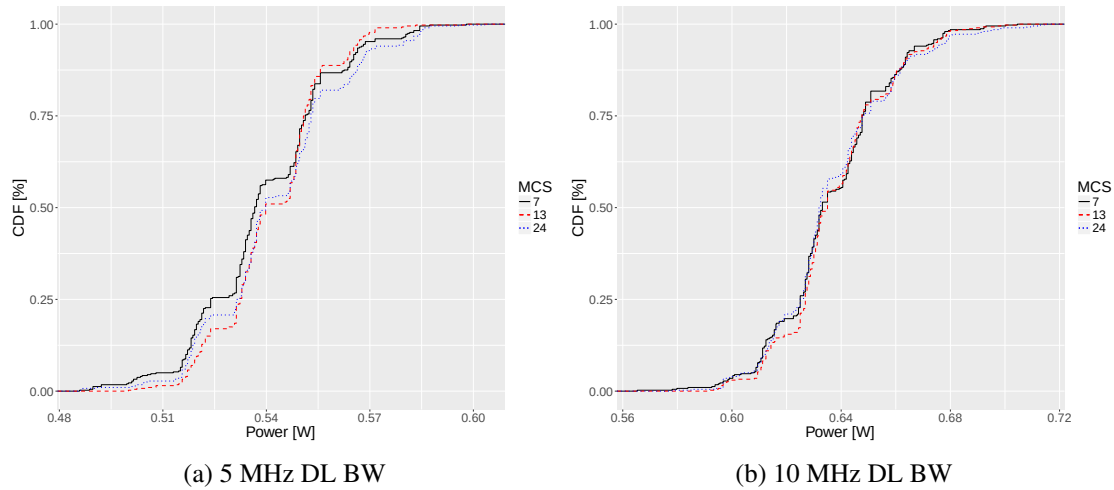


Figure 3.13: Impact of the MCS index on the power drained by the baseband SoC.

A third experiment investigates the relation between the MCS index and the power consumption observed at the HWA L1. This is done for the 5 MHz and 10 MHz DL BW configurations and with only 2 RBGs allocated. Then, three different MCS index values are used, namely 7, 13 and 24 (i.e., ranging from the lowest to the highest modulation order). As it can be observed in Figure 3.13, nearly identical results to the previous experiment are reported in this case, with the MCS index having little impact on the energy being drained by the baseband processor.

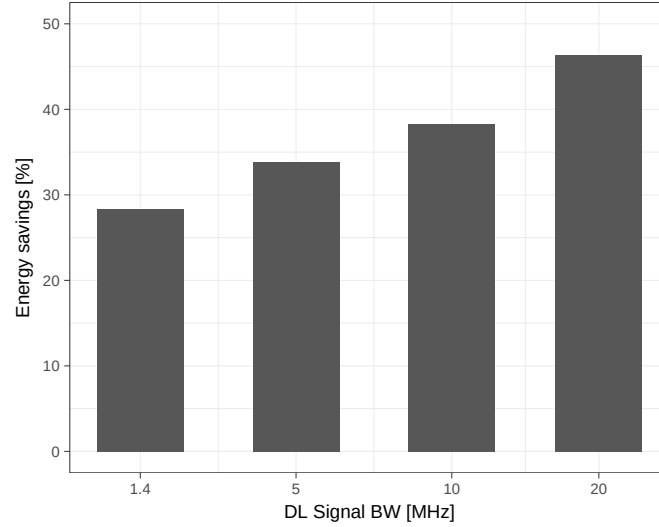


Figure 3.14: Energy savings reported by adopting NETCFG3.

After analysing the influence of reconfiguring the WCPs on the energy budget of the HWA L1, the fourth experiment is focusing on the dynamic split of communication functions. Specifically, the energy saving gains obtained by moving from *NETCFG1/2* to *NETCFG3* are evaluated (i.e., adoption of a C-RAN scheme, where the HeNB acts as a Remote Radio Head (RRH)). As reported in Figure 3.14, the consumed energy is considerably reduced (i.e., up to 46.38%) independently of the utilized DL BW configuration. In this case, the clock-gated design can be fully exploited, as the energy-hungry DSP logic can be put to an idle state, minimizing likewise the switching activity of the digital circuit.

From the point of view of the baseband processor, as it was expected, most energy saving benefits come from reducing the activity of the circuit. Toward that end, dynamically adapting the WCPs is not always the most efficient way to reduce the energy. More specifically, we have found that the reconfigurations affecting the MCS index or RBG allocation settings should focus on satisfying the frequently changing QoS requirements. In that case, although the energy-aware RTL design is capable of preventing the unnecessary operation of some parts of the system, this can only be done a fraction of the time and results in a minimal reduction of the energy drained by the PL. On the other end, reducing the operating frequency of the circuit results in a notable reduction of the consumed energy, exactly as it is expected for the CPU case. Thus, downscaling the DL BW has been proved as an effective means to minimize the energy consumption of the HeNB

during those periods where a reduced performance can serve the needs of the attached users. Similarly, distributing the underlying DSP functions across the network can be also exploited to effectively minimize the energy consumption of the HeNB.

3.5.2.4. Radio Frequency Integrated Circuit

The energy measurements presented for the AD9361 RFIC apply to all three considered NETCFGs (i.e., the RF stage is always active and operates locally, independently of the underlying distribution of functions). As in the HWA L1 case, different settings for the MCS index, DL BW and RBG allocation load are considered. Additionally, variations in the output power of the RFIC are also evaluated.

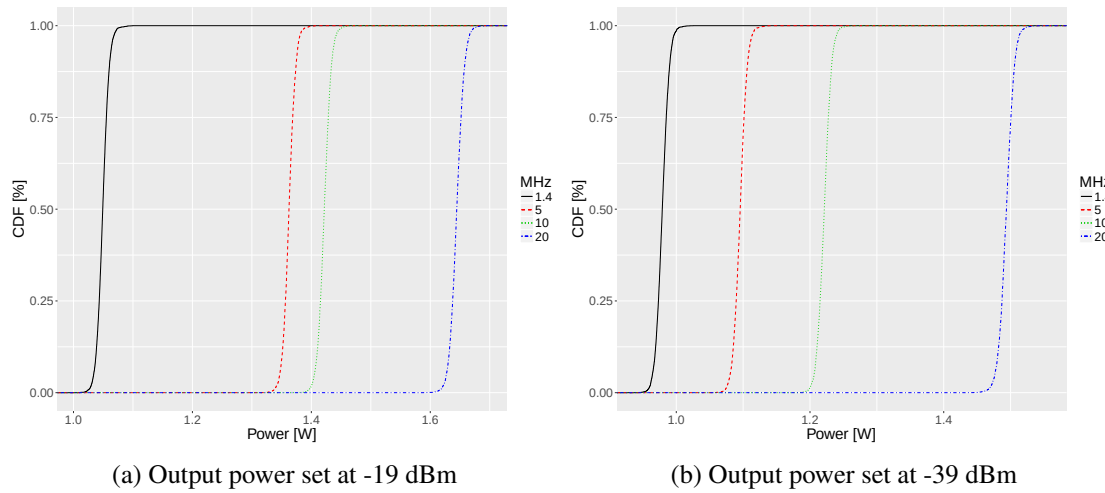


Figure 3.15: Variation of the power consumption at the RFIC as a result of changing the DL BW.

The weight of the DL BW configuration in the power drained by the RF stage is investigated in the first experiment presented here. Figure 3.15a shows the results for all four signal BW settings, when using a fixed MCS index of 7 (i.e., QPSK modulation), a complete allocation of the available RBG resources and an RF output power of -19 dBm. Similarly, the output power of the RFIC is attenuated to -39 dBm in Figure 3.15b. Analogous and elevated savings are observed in both cases, which grow above 34% in the limit situation of downscaling the DL BW from 20 MHz to 1.4 MHz.

In the second experiment conducted at the RF stage, the effect of attenuating the RF output power on the power consumption is investigated. As in the previous case, a fixed MCS index of 7 and a fully allocated DSLCH is considered. Moreover, three different RF output power settings are utilized (i.e., -19, -29 and -39 dBm). In Figure 3.16a, the resulting energy consumption for the 1.4 MHz DL BW case is shown, where modest energy savings can be observed (around 6%). Correspondingly, the 5 MHz setting is depicted in Figure 3.16b, where higher gains are reported (i.e., up to 20%).

A third experiment analyzes the impact of allocating a different number of RBGs on the power

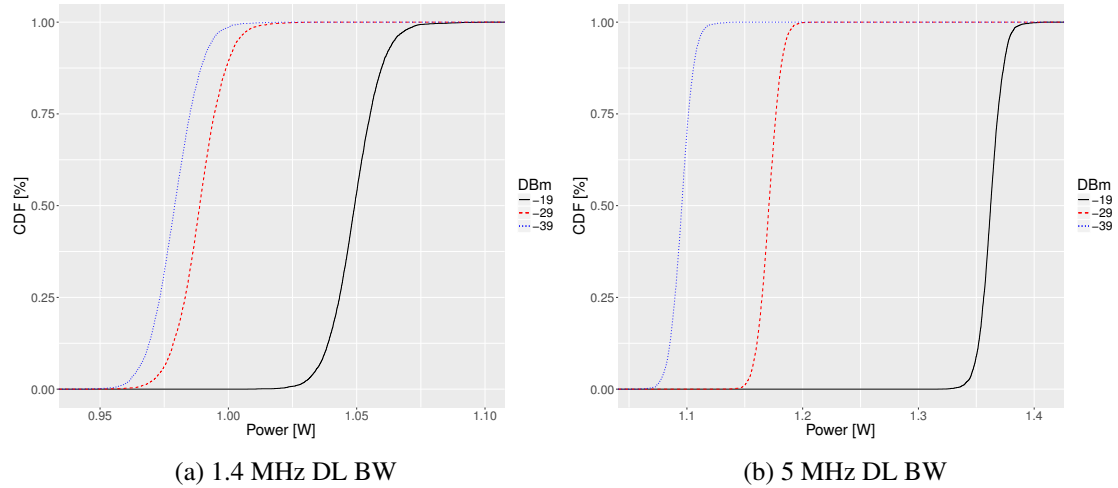


Figure 3.16: Variation of the power drained by the RFIC as a result of attenuating the RF output power.

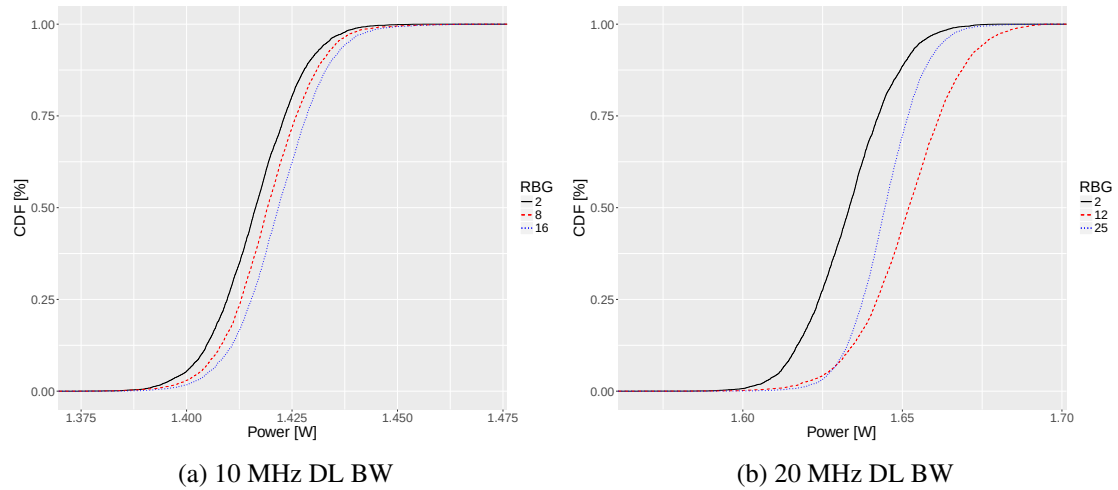


Figure 3.17: Energy consumed by the RFIC under different RBG allocation cases.

consumption of the RFIC. A fixed MCS index of 7 and output power of -19 dBm is used, for two different DL BW settings (i.e., 10 MHz and 20 MHz). As it can be observed in Figure 3.17 the variation of the energy consumption resulting from varying the RBG allocation is negligible. The exact same behaviour has been found when adapting the MCS index WCP.

As a summary, adapting the MCS index or the RBG allocation does not help in reducing the energy footprint of the RFIC. On the contrary, energy saving gains can be obtained by attenuating the RF output power in those use cases where the link quality allows it. Furthermore, adopting the most efficient DL BW configuration given the actual hotspot requirements is the optimum reconfiguration in power saving terms.

3.6. Conclusions

In this section, we have first introduced a high accuracy energy measurement platform that we have employed to characterize two different technologies. As a basis for our analysis, we have revisited 802.11 rate adaptation by taking energy consumption into account. While some previous studies pointed out that MIMO rate adaptation is not energy efficient, we have demonstrated through numerical analysis that, even for single spatial streams without interfering traffic, energy consumption and throughput performance are different optimization objectives. Furthermore, we have validated our results via experimentation. Our findings show that this trade-off emerges at certain “mode transitions” when maximising the goodput, suggesting that small goodput degradations may lead to energy efficiency gains. For instance, a station at the edge of a mode transition may decide to reduce the transmission power a little in order to downgrade the modulation coding scheme. Or an opportunity to achieve a better goodput by increasing the transmission power and modulation coding scheme could be delayed if the expected gain is small. Moreover, our analysis have showed that these trade-offs arise as a consequence of the power consumption behaviour of wireless cards and does not depend on the energy consumed in the rest of the device. In this way, energy-aware rate adaptation may be achieved building on information local to the wireless interface. Still, to develop energy-aware rate adaptation algorithms, further research is needed to understand how the findings of this work can be leveraged in suboptimal conditions, and how other effects, such as collisions and MIMO, affect the established trade-off.

As second objective of this chapter but not least important, we have extended our analysis to the energy saving benefits that yield from the dynamic reconfiguration of the hotspot for LTE deployments. This work has a strong applied component and revolves around the development of a real-time hotspot prototype. Furthermore, actual power consumption measurements of the different subsystems comprising the HeNB have been presented. The analysis of the energy consumption of the prototype has also allowed to select the most interesting reconfiguration strategies in the considered scenario. In more detail, both distributing the communication functions and/or downscaling the utilized signal BW can help to greatly reduce the energy consumption of the HeNB (i.e., savings up to 50% at a subsystem level). Hence, the energy-aware reconfiguration

of the hotspot might help optimizing its energy footprint during those periods where the system presents low performance requirements or is in need to save energy (e.g., battery powered HeNB). It has also been observed that adapting the MCS index or RBG load WCPs results in minimal energy savings and, thus, should not be used as a main trigger to reconfigure the system (from an energy saving perspective).

Chapter 4

Analysis and Optimization of Resource on Demand Schemes

In the previous chapter, we have first analyzed the energy impact in the Rate Adaptation (RA) algorithms in IEEE 802.11. Next, we have addressed the characterization of a reconfigurable and flexible LTE prototype, dissecting the energy consumption for *i*) multiple wireless communication parameters (MCS, RGB, BW) and, *ii*) different functional split configurations considering a Cloud RAN (C-RAN) scenario. Both analyses are quantified in a short temporal scale¹, dissecting the energy consumption at almost packet level scale.

In this chapter, we present an analysis, performance evaluation and optimization in a *larger temporal scale* of an entire wireless network deployment², considering that the system is capable to dynamically switch on/off the elements that form the network. We will denominate these schemes as Resource-on-Demand (RoD). To analyze this kind of deployment, we will consider real world constraints such as the start-up time of the devices, dissecting its impact on the overall performance. By “start-up time” we mean the time it takes between the AP is activated until the WLAN is announced. According to the seminal work of [75], typical start-up times of an AP range between 12 and 35 seconds.

More specifically, we analyze a general case of a RoD scenario consisting of N overlapping APs with non-zero start-up times. To achieve that, we present two flavors of the analytic model: *i*) an *exact* model that accurately predicts performance in terms of energy consumption and service time, but with a high computational complexity; and *ii*) a *simplified* model that sacrifices some numerical accuracy in exchange for more affordable computational times. Finally, we present one possible use of this simplified model, namely, the design of a simple configuration algorithm for RoD, based on the minimization of the average service time. As the results show, the simplified model supports the design of optimization policies that trade-off performance for significant gains in energy efficiency.

¹We define *short temporal scale* as measurements in a range of μs .

²By *large temporal scale* we define measurements in a range of seconds or even minutes.

4.1. System Model

4.1.1. Scenario

We consider a *cluster model* like the one analyzed in [48], consisting of N identical APs serving the same area but using non-overlapping channels. Although in typical high-density deployments the APs may not be located exactly at the same position, the high level of overlapping allows making this assumption, which simplifies the theoretical analysis. Indeed, as will be seen in Section 4.4, this assumption does not impact the validity of the RoD strategy.

The need for a dense deployment such as the one addressed in this section is motivated by the current trends in the increase of traffic demand. This trend has been forecasted by a number of sources. According to [76], the number of devices and connections per user is steadily growing, which increases user densities; in addition, the throughput required per user is also increasing, as new services such as HD video streaming are becoming ubiquitous. Along the same lines, the forecasts for future 5G networks,³ predict data rates 100 times higher than today's. Even now, a recent research estimates that typical densities in the deployment of APs may exceed 4000 APs per square kilometre [45].

The scenario considered could be mapped to a very-dense 802.11a setup, where there are many available channels in the 5 GHz band (the specific number depending on the country). One of the APs is always on, in order to maintain the WLAN coverage, while the other APs are opportunistically powered on (off) as users arrive (leave) the system. Powering an AP takes a deterministic time T_{on} and, during this time, the AP is not available, so arriving requests are served by any of the other APs. We neglect the time required to power off an AP.

Each AP consumes P_{AP} units of power when active (i.e., during start-up and when powered on) and zero otherwise. Although commodity hardware can support an intermediate state (i.e., switching on/off the wireless card), this does not bring as much savings as powering on/off the complete device [67]. A “user” is a new connection generated by a wireless client. Following [77] and [47], these are generated according to a Poisson process at rate λ and are always served by the less loaded AP. Also following [47], we further assume that users' demands are exponentially distributed (i.e., each user downloads an amount of data that is exponentially distributed) and that the AP bandwidth is evenly shared among all the users.⁴

Based on the above assumptions, service times (i.e., the time elapsed since a user arrives to the WLAN until it has fully downloaded its demanded data) would be exponentially distributed (with mean $1/\mu$) if every user got all the bandwidth of an AP, and the service rate (i.e., the inverse of the average service time) is μ when there is only one serving AP, 2μ when there are two APs

³<http://5g-ppp.eu/>.

⁴The assumption on the Poissonian nature of user arrivals is aligned with the characterisation driven by measurements provided by [78] and [79]. Furthermore, [47] shows that, while the distribution of the duration of user connections is not a memory-less process, it can still be approximated by an exponential distribution with reasonable accuracy. In the numerical evaluation, we will rely on a more accurate traffic model in order to assess the impact of the simplifying assumptions upon which our analysis relies.

serving, etc. (i.e., we neglect the impact of channel sharing via contention). The total load is given by $\rho = \lambda/N\mu$.

We also assume a load-balancing algorithm such that users (re)associate while they are being served, and that this (re)association time is negligible –note that this can be achieved with the recent 802.11v and 802.11r amendments [80], which support triggering re-associations and performing fast transitions, respectively, with minor disruption of the service.

4.1.2. Resource on Demand policy

In order to power on/off the APs we assume there is a “target” number $M > 1$ of users per AP, i.e., the system will opportunistically power on/off APs in order to keep that “target” number across resources (except for one AP that will be always on, to guarantee coverage). Based on this, we assume a threshold-based policy with hysteresis, namely:

- An AP will be powered on when the number of user per AP is ρ_h higher than this target value.
- An AP will be powered off when the number of user per AP is ρ_l lower than this target value.

In this way, with K APs powered on, the $K + 1$ -th AP will be powered on when the number of users reaches

$$\text{Threshold to power on another AP } (N_K): \lceil (1 + \rho_h)KM \rceil$$

while with K APs powered on, one AP will be powered off when the number of user reaches

$$\text{Threshold to power off an AP } (n_K): \lfloor (1 - \rho_l)KM \rfloor$$

We next impose some conditions on these thresholds to support an efficient operation. On the one hand, with K APs powered on we impose that there are at least K associated users, so all APs are serving traffic. This results in that the threshold to power off an AP with K users has to be at least K , i.e.,

$$n_K = \lfloor (1 - \rho_l)KM \rfloor \geq K, \quad (4.1)$$

which results on the following condition for ρ_l

$$\rho_l < 1 - \frac{1}{M}. \quad (4.2)$$

On the other hand, to prevent (or, at least, reduce) “flip-flop” effects in the WLAN (i.e., to power on an AP and, once active, immediately power it off), we assume that the ρ_h and ρ_l thresholds are set such that the number of users to power on an AP when K of them are already

serving traffic is larger than the number of users required to power off an AP when $K + 1$ are serving traffic, i.e.,

$$\lceil (1 + \rho_h)KM \rceil > \lfloor (1 - \rho_l)(K + 1)M \rfloor . \quad (4.3)$$

Based on the above condition, and neglecting the rounding operations, to prevent the flip-flop effects the following condition between ρ_h and ρ_l should hold

$$\rho_h > \frac{1 - \rho_l}{K} - \rho_l , \quad (4.4)$$

where the rhs of (4.4) is maximum for $K = 1$, i.e., the case of one AP, and therefore the ρ_h threshold should be set to at least⁵

$$\rho_h > 1 - 2\rho_l . \quad (4.5)$$

In addition to the above, for analytical tractability we introduce the following restriction on the RoD policy: at any point in time there will be at most one AP being powered on. More specifically, while one AP is powering on there will be no decisions taken w.r.t. powering on or off other resources, and only once the AP is available the system will decide on the amount of resources needed. Fig. 4.1 exemplifies this policy for $K = 2$.

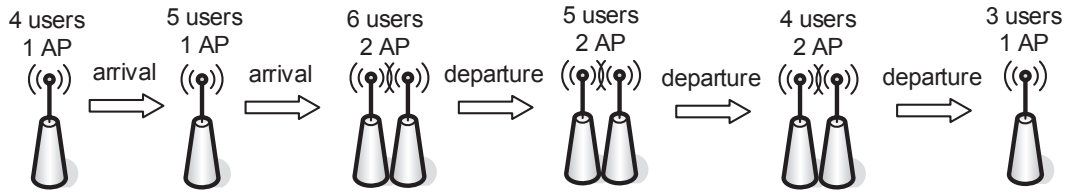


Figure 4.1: Example of the policy with $K = 2$ APs

4.2. Exact Analysis

4.2.1. Model overview

We model the system with the semi-Markov process illustrated in Fig. 4.2. The label in each arrow corresponds to the number of users (or range of users) in the system that triggers the transition between the corresponding stages. There are four types of *stages*, depending on the transitions that could happen between them:

- Stage 1, which is the initial situation with only one AP active. The only possible transition is to stage 1* (another AP is powered on), that is triggered when the number of users reaches N_1 .

⁵Note that for simplicity our policy is set on fixed values of ρ_h and ρ_l , i.e., they do not change with the number of active APs.

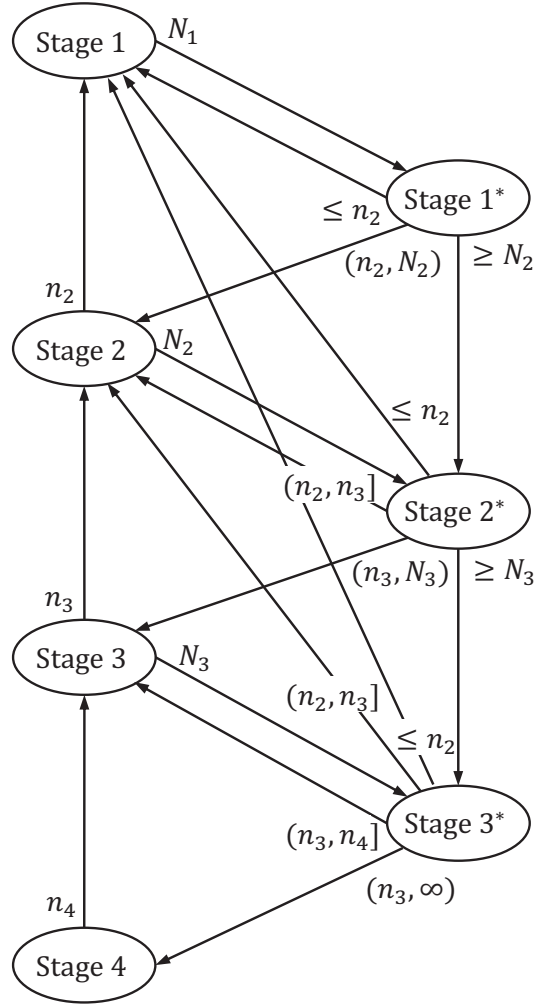


Figure 4.2: Semi-Markov process for an IoD scheme with $N = 4$ APs.

- Stage N , when all APs are active and serving traffic. The only possible transition is to stage $N - 1$ (one AP is powered off), what happens when the number of users is n_N .
- Stages K (with $1 < K < N$), where there are K active APs. In this case there are two possible transitions: one to stage K^* , triggered when the number of users reaches N_K and another AP is powered on (label N_K in Fig. 4.2); and other to stage $K - 1$, triggered when the number of users in the system is n_K and one AP is powered off (label n_K in Fig. 4.2).
- Stages K^* (with $1 \leq K < N$), where in addition to the K active APs there is another AP booting up. For this type of stage there is a larger number of possible transitions, which are determined by the number of users in the system after T_{on} :
 - If there are N_{K+1} or more users, the system will move to stage $K + 1^*$, as the

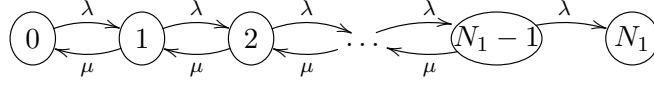


Figure 4.3: CTMC for stage 1: one active AP and no AP powering on

number of users is already above the threshold to switch on an additional AP. These transitions are marked with the label $\geq N_{K+1}$ in Fig. 4.2.

- If there are between $n_{K+1} + 1$ and $N_{K+1} - 1$ users, the system will move to stage $K + 1$. These transitions are marked with the label (n_{K+1}, N_{K+1}) in Fig. 4.2.
- If there are n_{K+1} users or less, the next stage will depend on whether the number of users is also less than or equal to n_2 (and therefore the next stage will be '1'), between $n_2 + 1$ and n_3 (the next stage will be '2'), and so on. These transitions are marked with the labels $\leq n_2, (n_2, n_3], \dots$ in Fig. 4.2.

With the above, we have introduced the different stages of the semi-Markov process. We next analyze each type of stage, their holding times, and the transition probabilities between them.

4.2.2. Modelling the stages of the semi-Markov model

4.2.2.1. Stage 1 (S_1)

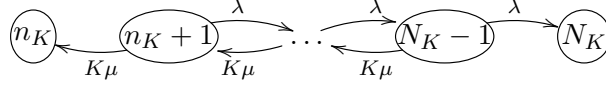
For the initial situation with only one AP active, following our assumptions the system can be modelled with the continuous-time Markov chain (CTMC) illustrated in Fig. 4.3, where each state represents the number of users being served and therefore reaching the absorbing state N_1 corresponds to the case when another AP will be powered on (and stage 1 will be left).

The average time in this stage H_1 corresponds to the *time until absorption* of the Markov chain, i.e., the time since the system arrived to the chain until it reaches the absorbing state N_1 . If we define $L_i(t)$ as the expected total time that a CTMC spends in state i during the interval $[0, t)$, H_1 can be expressed as the sum of the terms $L_i(t)$ for all the non-absorbing states of the CTMC when $t \rightarrow \infty$ [81]

$$H_1 = \sum_{i=0}^{N_1-1} L_i^{(1)}(\infty). \quad (4.6)$$

The values of $L_i^{(1)}(\infty)$ (the superscript (1) indicates that we are referring to the CTMC modelling stage 1) can be obtained solving the following system of equations:

$$\mathbf{L}^{(1)}(\infty)\mathbf{Q}^{(1)} = -\boldsymbol{\pi}^{(1)}(0), \quad (4.7)$$

Figure 4.4: CTMC for Stage K : K active APs and no AP powering on

where

$$\mathbf{L}^{(1)}(\infty) = \left[L_0^{(1)}(\infty), L_1^{(1)}(\infty), \dots, L_{N_1-1}^{(1)}(\infty) \right], \quad (4.8)$$

$$\boldsymbol{\pi}^{(1)}(0) = \left[\pi_0^{(1)}(0), \pi_1^{(1)}(0), \dots, \pi_{N_1-1}^{(1)}(0) \right], \quad (4.9)$$

with $\pi_i^{(1)}(0)$ the initial probability of state i , and $\mathbf{Q}^{(1)}$ a $N_1 \times N_1$ matrix with the following non-zero elements:

$$q_{ij} = \begin{cases} -\lambda & i = 1, j = 1 \\ -\lambda - \mu & i = 2, \dots, N_1, j = i \\ \lambda & i = 1, \dots, N_1 - 1, j = i + 1 \\ \mu & i = 2, \dots, N_1, j = i - 1 \end{cases} \quad (4.10)$$

The computation of $\boldsymbol{\pi}^{(1)}(0)$ is not straightforward, as it depends on the stage the system was before arriving to stage 1, which could be stage 2 or any other stage K^* , with $K \geq 1$. We detail how to compute $\boldsymbol{\pi}^{(1)}(0)$ for this and the other cases in the next section, after we present the modelling of the other stages of the semi-Markov process.

Finally, let $P(S_T | S_{T'})$ denote the transition probability from stage T' to stage T , with T and T' referring indistinctly to any stage K , including stages 0 and N , or K^* . For the case of stage 1, we have that

$$P(S_{1*} | S_1) = 1. \quad (4.11)$$

4.2.2.2. Stages K (S_K), $1 < K < N$

For these stages, the resulting CTMC is illustrated in Fig. 4.4. In this case, while the arrival rate is also λ , the service rate accounts for the total number of powered-on APs, which is constant and equal to $K \cdot \mu$ for all states.⁶ As described above, there are two absorbing states: one corresponding to the powering on of another AP (when the system reaches N_K users), and another corresponding to the de-activation of one AP (when the number of users is n_K).

⁶Note that we have imposed $n_K > K$ with (4.1).

Similarly to the previous case, the average time in a stage K can be computed as

$$H_K = \sum_{i=n_K+1}^{N_K-1} L_i^{(K)}(\infty). \quad (4.12)$$

In order to compute $L_i^{(K)}(\infty)$ we use the same expression as in the previous case

$$\mathbf{L}^{(K)}(\infty)\mathbf{Q}^{(K)} = -\boldsymbol{\pi}^{(K)}(0), \quad (4.13)$$

where

$$\mathbf{L}^{(K)}(\infty) = \left[L_{n_K+1}^{(K)}(\infty), L_{n_K+2}^{(K)}(\infty), \dots, L_{N_K-1}^{(K)}(\infty) \right], \quad (4.14)$$

$$\boldsymbol{\pi}^{(K)}(0) = \left[\pi_{n_K+1}^{(K)}(0), \pi_{n_K+2}^{(K)}(0), \dots, \pi_{N_K-1}^{(K)}(0) \right], \quad (4.15)$$

and $\mathbf{Q}^{(K)}$ is a $(N_K - n_K - 1) \times (N_K - n_K - 1)$ matrix, whose non-zero elements are

$$q_{ij} = \begin{cases} -\lambda - K\mu & i = 1, \dots, N_K - n_K - 1, j = i \\ \lambda & i = 1, \dots, N_K - n_K - 2, j = i + 1 \\ K\mu & i = 2, \dots, N_K - n_K - 1, j = i - 1 \end{cases} \quad (4.16)$$

Again, the computation of $\boldsymbol{\pi}^{(K)}(0)$ requires the knowledge of the stage of the system before entering stage K , which we will address in the next section.

To finalise the analysis of this stage, we have to compute the two transition probabilities from this stage to stage K^* (when the chain ends in the absorbing state N_K) and to stage $K - 1$ (when the chain falls into the absorbing state n_K), denoted as $P(N_K)$ and $P(n_K)$ respectively,

$$P(S_{K^*} | S_K) = P(N_K), \quad (4.17)$$

$$P(S_{K-1} | S_K) = P(n_K). \quad (4.18)$$

These can be computed as [82]

$$[P(n_K) \ P(N_K)] = \boldsymbol{\pi}^{(K)}(0)\mathbf{B}^{(K)}, \quad (4.19)$$

with $\mathbf{B}^{(K)}$ a $(N_K - n_K - 1) \times 2$ matrix whose element b_{ij} is the probability of ending in the absorbing state j , given that the chain starts in the transient state i . This matrix can be computed as

$$\mathbf{B}^{(K)} = [\mathbf{I} - \mathbf{T}^{(K)}]^{-1} \mathbf{R}^{(K)}, \quad (4.20)$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{T}^{(K)}$ is a $(N_K - n_K - 1) \times (N_K - n_K - 1)$ matrix with the transition probabilities between non-absorbing states, and $\mathbf{R}^{(K)}$ is a $(N_K - n_K - 1) \times 2$ matrix denoting the

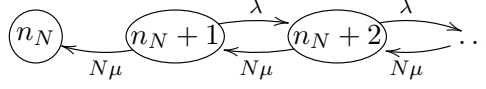


Figure 4.5: Markov chain when all APs are active

transition probabilities from non-absorbing to absorbing states. Both matrices are obtained from the associated discrete-time Markov chain (DTMC) of the CTMC, and their non-zero elements are

$$t_{ij} = \begin{cases} \frac{\lambda}{\lambda + K\mu} & i = 1, \dots, N_K - n_K - 2, j = i + 1 \\ \frac{K\mu}{\lambda + K\mu} & i = 2, \dots, N_K - n_K - 1, j = i - 1 \end{cases} \quad (4.21)$$

$$r_{ij} = \begin{cases} \frac{K\mu}{\lambda + K\mu} & i = 1, j = 1 \\ \frac{\lambda}{\lambda + K\mu} & i = N_K - n_K - 1, j = 2 \end{cases} \quad (4.22)$$

4.2.2.3. Stage N (S_N)

When all APs are active and serving traffic the resulting CTMC is the one depicted in Fig. 4.5, where the arrival rate is λ and the service rate is $N \cdot \mu$. As in the case of stage 1, there is only one absorbing state, the one corresponding to the switching off of one AP when there are n_N users in the system and all the APs are on, but now the chain has an infinite number of states.

To compute the holding time in this stage, we assume that the system is stable (i.e., $\lambda < N\mu$), so there is a state n_D with $n_D > n_N$ such that

$$\sum_{i=n_D+1}^{\infty} L_i^{(N)}(\infty) \approx 0, \quad (4.23)$$

and therefore the holding time is

$$H_N \approx \sum_{i=n_N+1}^{n_D} L_i^{(N)}(\infty), \quad (4.24)$$

where $\mathbf{L}^{(N)}(\infty)$ is obtained from

$$\mathbf{L}^{(N)}(\infty)\mathbf{Q}^{(N)} = -\boldsymbol{\pi}^{(N)}(0), \quad (4.25)$$

with

$$\mathbf{L}^{(N)}(\infty) = [L_{n_N+1}^{(N)}(\infty), L_{n_N+2}^{(N)}(\infty), \dots, L_{n_D}^{(N)}(\infty)], \quad (4.26)$$

$$\boldsymbol{\pi}^{(N)}(0) = [\pi_{n_N+1}^{(N)}(0), \pi_{n_N+2}^{(N)}(0), \dots, \pi_{n_D}^{(N)}(0)], \quad (4.27)$$

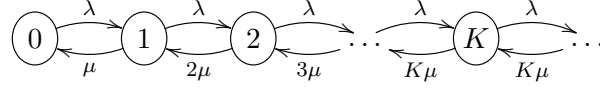


Figure 4.6: Markov chain for the case of K active APs and one AP powering on

and $\mathbf{Q}^{(N)}$ is a $(n_D - n_N) \times (n_D - n_N)$ matrix with the following non-zero elements

$$q_{ij} = \begin{cases} -\lambda - N\mu & i = 1, \dots, n_D - n_N, j = i \\ \lambda & i = 1, \dots, n_D - n_N - 1, j = i + 1 \\ N\mu & i = 2, \dots, n_D - n_N, j = i - 1 \end{cases} \quad (4.28)$$

The computation of $\pi^{(N)}(0)$ is described in the next section. From this stage, the only possible transition is to stage $N - 1$, i.e.,

$$P(S_{N-1} | S_N) = 1. \quad (4.29)$$

4.2.2.4. Stages K^* (S_{K^*})

For the stages with K active APs and one AP being powered on, the resulting Markov chain is illustrated in Fig. 4.6. In these stages there are no absorbing states that trigger the transition to other stages, since this happens when the amount of time spent in the stage is T_{on} . Because of this, the number of users that can be in the system during this stage varies between zero and infinity. Additionally, the service rate depends on the number of users as there should be at least one user per active AP for the total rate to be $K\mu$.⁷ For completeness, the time spent in a stage K^* is given by

$$H_{K^*} = T_{on}. \quad (4.30)$$

In this case, we need to obtain the expected time that the system spends in each state i during the T_{on} seconds that a stage K^* lasts, $L_i^{(K^*)}(T_{on})$, and the probability of each state after T_{on} , $\pi_i^{(K^*)}(T_{on})$. These terms are required to compute the transition probabilities from stage K^* to the other stages and to obtain the performance figures of the system. The values for $L_i^{(K^*)}(T_{on})$ and $\pi_i^{(K^*)}(T_{on})$ can be obtained with the expressions of the transient analysis of an M/M/K queue, which are

$$L_i^{(K^*)}(t) = \int_0^t \pi_i^{(K^*)}(u) du, \quad (4.31)$$

where $\pi_i^{(K^*)}(t)$ is the probability of being in state i at time t , which is determined by the funda-

⁷In fact, the CTMC corresponds to the classic M/M/K queue

mental equations of the CTMC

$$\frac{d\boldsymbol{\pi}^{(K*)}(t)}{dt} = \boldsymbol{\pi}^{(K*)}(t) \mathbf{Q}^{(K*)}, \quad (4.32)$$

with $\boldsymbol{\pi}^{(K*)}(t) = [\pi_i^{(K*)}(t)]_i$ the transient state probability vector and $\mathbf{Q}^{(K*)}$ the infinitesimal generator matrix of the CTMC. The non-zero elements of $\mathbf{Q}^{(K*)}$ are

$$q_{ij} = \begin{cases} -\lambda - (i-1)\mu & i = 1, \dots, K, j = i \\ -\lambda - K\mu & i = K+1, \dots, j = i+1 \\ \lambda & i = 1, \dots, j = i+1 \\ (i-1)\mu & i = 2, \dots, K, j = i-1 \\ K\mu & i = K+1, \dots, j = i-1 \end{cases} \quad (4.33)$$

Note that to solve (4.31) and (4.32), we need again the vector of initial state probabilities $\boldsymbol{\pi}^{(K*)}(0)$. On the other hand, as there are no closed expressions for the transient behaviour of an M/M/K queue, we need to use approximate methods (such as *uniformization* [81]) to solve it and compute $L_i(T_{on})$ and $\pi_i(T_{on})$. Like in the previous cases, the computation of $\boldsymbol{\pi}^{(S_{K*})}(0)$ is described in the next section.

Finally, as noted before, from stage $K*$ the system can go to any other stage K' , with $K' \leq K+1$, and to stage $K+1*$. In this way, after T_{on} the system can have K or less APs powered on, with the following probabilities

$$P(S_{K'} | S_{K*}) = \begin{cases} \sum_{i=0}^{n_2} \pi_i^{(K*)}(T_{on}), & K' = 1 \\ \sum_{i=n_{K'}+1}^{n_{K'+1}} \pi_i^{(K*)}(T_{on}), & 1 < K' \leq K \end{cases} \quad (4.34)$$

while the probability of having more APs on (or being powered on) after T_{on} depends on whether there are more APs to be powered on, i.e., if $K < N-1$, or not ($K = N-1$). For the former case, we have that

$$P(S_{K+1} | S_{K*}) = \sum_{i=n_{K+1}+1}^{N_{K+1}-1} \pi_i^{(K*)}(T_{on}), \quad (4.35)$$

$$P(S_{K+1*} | S_{K*}) = \sum_{i=N_{K+1}}^{\infty} \pi_i^{(K*)}(T_{on}), \quad (4.36)$$

while for the case of $K = N-1$ there are no more APs to activate, and therefore

$$P(S_N | S_{N-1*}) = \sum_{i=n_N+1}^{\infty} \pi_i^{(N-1*)}(T_{on}). \quad (4.37)$$

4.2.3. Computing the steady-state distribution

To complete the analysis of the steady-state distribution of the semi-Markov process, we have to express the set of initial conditions for every stage in terms of the final state probabilities of the other stages. To this end, we denote $\pi_i^{(T)}(0)$ as the probability that the initial state is i for the stage T (again we use T for generalization purposes, with stage T we refer indistinctly to any stage K , including stages 0 and N , or K^*). This probability can be computed with the law of total probability as

$$\pi_i^{(T)}(0) = \sum_{S_{T'} \in \mathcal{T}_T} \pi_i^{(T|T')}(0) P_T(T'), \quad (4.38)$$

where

- $\mathcal{T}_T$ is the set of stages that can reach stage T in one transition between stages,
- $\pi_i^{(T|T')}(0)$ is the probability that the initial state of the CTMC modelling stage T is i , given that the system was in stage T' and transitioned to state T ,
- $P_T(T')$ is the probability that the system was in stage T' before the stage transition, given that it is now in stage T .

The set $\mathcal{T}$ can be easily derived for each stage from the Semi-Markov model described in Section 4.2.1. Specifically we have,

$$\mathcal{T}_1 = \{S_2, S_{1*}, \dots, S_{N-1*}\}, \quad (4.39)$$

$$\mathcal{T}_K = \{S_{K+1}, S_{K-1*}, \dots, S_{N-1*}\} \text{ for } 1 < K < N, \quad (4.40)$$

$$\mathcal{T}_N = \{S_{N-1*}\}, \quad (4.41)$$

$$\mathcal{T}_{1*} = \{S_1\}, \quad (4.42)$$

$$\mathcal{T}_{K*} = \{S_K, S_{K-1*}\} \text{ for } 1 < K < N - 1. \quad (4.43)$$

As an example, for the case of Fig. 4.2 with 4 APs, we have $\mathcal{T}_1 = \{S_2, S_{1*}, S_{2*}, S_{3*}\}$, $\mathcal{T}_2 = \{S_3, S_{1*}, S_{2*}, S_{3*}\}$, $\mathcal{T}_3 = \{S_4, S_{2*}, S_{3*}\}$, $\mathcal{T}_4 = \{S_{3*}\}$, $\mathcal{T}_{1*} = \{S_1\}$, $\mathcal{T}_{2*} = \{S_2, S_{1*}\}$, $\mathcal{T}_{3*} = \{S_3, S_{2*}\}$.

The computation of $\pi_i^{(T|T')}(0)$ depends on whether stage T' corresponds to a stage with an AP being powered on or not. For the latter case, the transition is triggered because the number of

stations reached a (de)activation threshold (i.e., an absorbing state), and therefore we have

$$\pi_i^{(K^*|K)}(0) = \begin{cases} 1, & i = N_K \\ 0, & \text{otherwise} \end{cases} \quad (4.44)$$

for $1 \leq K < N$ (note that we have included here the transition from stage 1 to stage 1* as well), and

$$\pi_i^{(K-1|K)}(0) = \begin{cases} 1, & i = n_K \\ 0, & \text{otherwise} \end{cases} \quad (4.45)$$

for $1 < K \leq N$ (we have included the transition from stage N to stage $N - 1$ as well). On the other hand, when stage T' is a K^* stage, there are multiple states that can result in a transition to a stage, which results in the following cases:

(i) If the transition is to stage 1 ($S_T = S_1$), then

$$\pi_i^{(1|K^*)}(0) = \begin{cases} \frac{\pi_i^{(K^*)}(T_{on})}{\sum_{j=0}^{n_2} \pi_j^{(K^*)}(T_{on})}, & 0 \leq i \leq n_2 \\ 0, & n_2 < i < N_1 \end{cases} \quad (4.46)$$

(ii) If the transition is to a stage $1 < K' \leq K$, then

$$\pi_i^{(K'|K^*)}(0) = \begin{cases} \frac{\pi_i^{(K^*)}(T_{on})}{\sum_{j=n_{K'}+1}^{n_{K'+1}} \pi_j^{(K^*)}(T_{on})}, & n_{K'} < i \leq n_{K'+1} \\ 0, & n_{K'+1} < i < N_{K'} \end{cases} \quad (4.47)$$

(iii) If $K < N - 1$ (i.e. $S_{T'} \neq S_{N-1}$) and the transition is to stage $K + 1$, then

$$\pi_i^{(K+1|K^*)}(0) = \frac{\pi_i^{(K^*)}(T_{on})}{\sum_{j=n_{K+1}+1}^{N_{K+1}-1} \pi_j^{(K^*)}(T_{on})}. \quad (4.48)$$

(iv) If $K < N - 1$ (again $S_{T'} \neq S_{N-1}$) and the transition is to stage $K + 1^*$, then

$$\pi_i^{(K+1|K^*)}(0) = \begin{cases} 0, & 0 \leq i \leq N_{K+1} \\ \frac{\pi_i^{(K^*)}(T_{on})}{\sum_{j=N_{K+1}}^{\infty} \pi_j^{(K^*)}(T_{on})}, & i > N_{K+1} \end{cases} \quad (4.49)$$

(v) If $K = N - 1$ and the transition is to stage N , then

$$\pi_i^{(N|N-1^*)}(0) = \frac{\pi_i^{(N-1^*)}(T_{on})}{\sum_{j=n_N+1}^{\infty} \pi_j^{(K^*)}(T_{on})}. \quad (4.50)$$

Finally, the computation of $P_T(T')$ can be done with the law of total probability again

$$P_T(T') = \frac{P(S_T | S_{T'})\phi_{T'}}{\sum_{S_Q \in \mathcal{T}_T} P(S_T | S_Q)\phi_Q}, \quad (4.51)$$

where $P(S_T | S_{T'})$ denotes the stage transition probability computed in (4.11), (4.17), (4.18), (4.29), (4.34)-(4.37), and ϕ_T is the stationary probability of stage T in the embedded Markov chain of the semi-Markov process. The computation of ϕ_T is done via the system

$$\phi = \phi \mathbf{P}, \quad (4.52)$$

where ϕ is a row vector whose components are the values of ϕ_T , and $\mathbf{P}$ is a matrix composed of the stage transition probabilities of the embedded Markov chain.

With the above, we have completed the analysis that enables in the next section the computation of the steady state probabilities of the semi-Markov model. We also address there how to compute performance figures based on these probabilities.

4.2.4. Performance figures

We characterize the performance of the system with two figures:

- The average power consumed by the infrastructure P .
- The average service time of a user T_s .

The average power consumed by the infrastructure can be expressed in terms of the average number of APs that are powered on, N_{AP} , as follows

$$P = N_{AP}P_{AP}. \quad (4.53)$$

N_{AP} is computed as the weighted sum of the number of APs powered on in each stage times the probability of being in that stage

$$N_{AP} = \sum_{K=1}^N K\mathcal{P}_K + \sum_{K=1}^{N-1} (K+1)\mathcal{P}_{K*}, \quad (4.54)$$

where $\mathcal{P}_K$ and $\mathcal{P}_{K*}$ are the stationary probabilities of the stages of the semi-Markov process, i. e., the probability of being in stage K (including stages 0 and N) or $K*$ at a specific moment. These probabilities are related to the stage probabilities of the embedded Markov chain as follows

$$\mathcal{P}_T = \frac{H_T\phi_T}{\sum_{K=1}^N \phi_K H_K + \sum_{K=1}^{N-1} \phi_{K*} H_{K*}}. \quad (4.55)$$

The average service time T_s , which corresponds to the time between the instant when a user generates a service request and when this request is completely served, can be obtained via Little's

formula

$$T_s = \frac{N_u}{\lambda}, \quad (4.56)$$

with N_u the average number of users in the system. This can be computed with the law of total probability as follows

$$N_u = \sum_{i=1}^{\infty} i \left(\sum_{K=1}^N \pi_i^{(K)} \mathcal{P}_K + \sum_{K=1}^{N-1} \pi_i^{(K*)} \mathcal{P}_{K*} \right), \quad (4.57)$$

where $\pi_i^{(K)}$ and $\pi_i^{(K*)}$ are the average probabilities of having i users, given that the system is in stage K or $K*$, respectively. This can be computed, for each type of stage, as

$$\pi_i^{(K)} = \frac{L_i^{(K)}(\infty)}{H_K}, \quad (4.58)$$

$$\pi_i^{(K*)} = \frac{L_i^{(K*)}(T_{on})}{T_{on}}. \quad (4.59)$$

As can be seen, all the performance metrics depend on the variables ϕ_T , H_T , $L_i^{(K)}(\infty)$ and $L_i^{(K*)}(T_{on})$, whose relationships have been described through Sections 4.2.2 to 4.2.3. In order to obtain an exact solution for them, we should solve a system of non-linear equations with the additional problem that there are no closed expressions for the transient analysis of the CTMC modelling stages $K*$. To solve this, we propose the iterative algorithm described in *Algorithm 1*. In this algorithm, the initial values of $\pi^{(K)}(0)$ can be set assuming that all the states with non-zero probabilities according to (4.46)-(4.50) have the same initial probability. Regarding $\pi^{(K*)}(0)$, a good starting guess is to assume that $\pi_i^{(K*)}(0) = 1$ for $i = N_K$ and 0 otherwise (this is what would happen if $T_{on} = 0$). Finally, a common stopping criterion is that the norm of the vector difference between the old and updated version of vectors $\pi^{(K)}(0)$ and $\pi^{(K*)}(0)$ is below a threshold ϵ .

4.3. Simplified Analysis

4.3.1. Motivation and simplification

The main weaknesses of the model derived in the previous section is that the initial probabilities of a stage depends on the “final” probabilities of the rest of stages, which depend in turn of their initial probabilities. This causes a loop that requires the use of an iterative algorithm with non-negligible computational complexity as the one described above. We next describe how to simplify the analytical model of Section 4.2 to enable an efficient computation of the performance figures, at the cost of some numerical inaccuracy.

The proposed simplification affects exclusively the transitions from stages $K*$. As can be seen in Fig. 4.2, from these stages the system could go to stage $K + 1*$ or any other stage K' ,

Algorithm 1 Solution to the exact model

-
- 1: Set initial estimations of $\pi^{(K)}(0)$ and $\pi^{(K^*)}(0)$
 - 2: **repeat**
 - 3: Compute $\mathbf{L}^{(K)}(\infty)$ with (4.7), (4.13) and (4.25)
 - 4: Obtain H_K with (4.6), (4.12) and (4.24)
 - 5: Solve (4.31)-(4.32) to obtain $\mathbf{L}^{(K^*)}(T_{on})$ and $\pi^{(K^*)}(T_{on})$
 - 6: Compute $P(T | T')$ with (4.17)-(4.20) and (4.34) - (4.37)
 - 7: Solve (4.52) to obtain ϕ
 - 8: Obtain $P_T(T')$ with (4.51)
 - 9: Compute $\pi_i^{(T|T')}(0)$ with (4.44)-(4.50)
 - 10: Update $\pi^{(K)}(0)$ and $\pi^{(K^*)}(0)$ with (4.38)
 - 11: **until** Stopping criterion is met
 - 12: Obtain $\pi_i^{(K)}$ and $\pi_i^{(K^*)}$ with (4.58) and (4.59)
 - 13: Compute $\mathcal{P}_T$ with (4.55)
 - 14: Obtain N_u with (4.57) and N_{AP} with (4.54)
 - 15: Compute T_s with (4.56) and P with (4.53)
-

with $K' \leq K+1$. The direct transitions between stages K^* make that the initial state probabilities for these stages $\pi_i^{(K^*)}(0)$ could be non-zero for $i \geq N_K$.

To break the coupling between stages K^* , we assume that the initial state probabilities of stages K^* are fixed and equal to

$$\pi_i^{(S_{K^*})}(0) = \begin{cases} 1, & i = N_K \\ 0, & \text{otherwise} \end{cases} \quad (4.60)$$

This implies that the system enters into stages K^* always with N_K users. This assumption holds as long as the transition probability between a stage K^* and the stage $K+1^*$ is small, which is true for typical T_{on} values.

Once this assumption is made, the transition probabilities from stages K^* to other stages are fixed and independent of the initial state probabilities of the rest of stages. Now, we also have to tackle the same apparent coupling for the initial state probabilities of stages K . To solve this, we build a new semi-Markov model derived from the one depicted in Fig. 4.2 substituting stages K by the embedded DTMC of their corresponding CTMC. The description of this new model is performed in the next Section.

4.3.2. Model description

Fig. 4.7 shows the embedded DTMC of the semi-Markov process described above. The leftmost states, which model stages K (including 0 and N), are defined by the pair (i, K) , with i the number of users in the system and K the number of powered-on APs. The holding time of these states is an exponential random variable with mean $(\lambda + K\mu)^{-1}$. The rightmost states model the stages K^* and their holding time is constant and equal to T_{on} . To keep a uniform notation, we

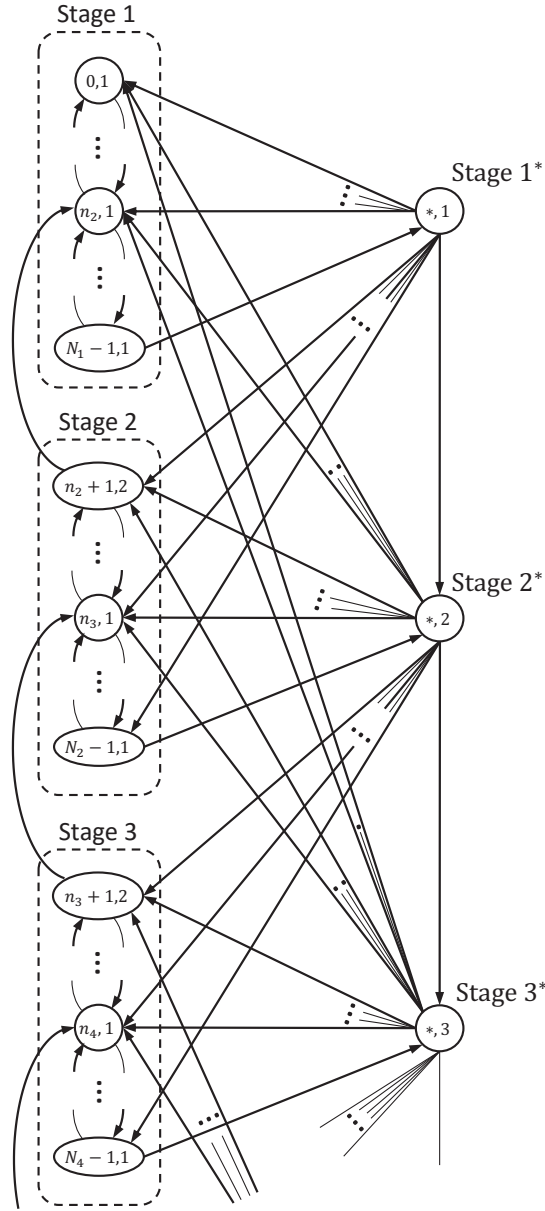


Figure 4.7: Simplified model.

note these states as $(*, K)$. The non-null transitions probabilities are described in (4.61).

The first two equations model the transitions between states of Stage 1, the first one corresponds to the departure of a user and the second one its arrival. The third and fourth equations model the transitions between states of stages $2 \leq N \leq N - 1$ and the fifth and sixth the transitions between states of stage N . The seventh equation corresponds to the switch off of an AP when a user departs and stage K remains with n_K users, which triggers the transition to stage $K - 1$. The eighth equation models the switching on of a new AP (i.e. the transition to stage $K*$) when the N_K -th user arrives and K APs are on. Note that in all the cases the transition

$$\left\{ \begin{array}{ll}
P(i-1, 0 \mid i, 0) = \frac{\mu}{\lambda+\mu} & i = \{1, \dots, N_1 - 1\} \\
P(i+1, 0 \mid i, 0) = \frac{\lambda}{\lambda+\mu} & i = \{0, \dots, N_1 - 2\} \\
P(i-1, K \mid i, K) = \frac{K\mu}{\lambda+K\mu} & i = \{n_K + 2, \dots, N_K - 1\}, \\
& K = \{2, \dots, N - 1\} \\
P(i+1, K \mid i, K) = \frac{\lambda}{\lambda+K\mu} & i = \{n_K + 1, \dots, N_K - 2\}, \\
& K = \{2, \dots, N - 1\} \\
P(i-1, N \mid i, N) = \frac{N\mu}{\lambda+N\mu} & i = \{n_N + 2, \dots\} \\
P(i+1, N \mid i, N) = \frac{\lambda}{\lambda+N\mu} & i = \{n_N + 1, \dots\} \\
P(n_K, K-1 \mid n_K+1, K) = \frac{K\mu}{\lambda+K\mu} & K = \{2, \dots, N\} \\
P(*, K \mid N_K - 1, K) = \frac{\lambda}{\lambda+K\mu} & K = \{1, \dots, N - 1\} \\
P(*, K+1 \mid *, K) = \sum_{i=N_{K+1}}^{\infty} \pi_i^{(K*)}(T_{on}) & K = \{1 \dots, N - 2\} \\
P(i, K+1 \mid *, K) = \pi_i^{(K*)}(T_{on}) & i = \{n_{K+1} + 1, \dots, N_{K+1} - 1\}, \\
& K = \{1 \dots, N - 2\} \\
P(i, N \mid *, N-1) = \pi_i^{(N-1*)}(T_{on}) & i = \{n_N + 1, \dots\} \\
P(i, K' \mid *, K) = \pi_i^{(K*)}(T_{on}) & i = \{n_{K'} + 1, \dots, n_{K'+1}\}, \\
& K = \{2 \dots, N - 1\}, K' = \{2 \dots, K\} \\
P(i, 1 \mid *, K) = \pi_i^{(K*)}(T_{on}) & i = \{0, \dots, n_2\}, K = \{1 \dots, N - 1\}
\end{array} \right. \quad (4.61)$$

probabilities only depend on the parameters λ , μ and the number of APs that are serving traffic at the moment of the transition.

The next equations model the transitions from states $(*, K)$, (i.e., from stages $K*$). Now the transition probabilities are of the form $\pi_i^{(K*)}(T_{on})$ and can be computed solving (4.31) and (4.32) assuming the initial state probabilities given in (4.60). Specifically, the ninth equation corresponds to the transition from stage $K*$ to stage $K+1*$ because the system reaches N_{K+1} users during the booting-up of the $K+1$ -th AP. The tenth equation models the transition from stage $K*$ to a state where there are $K+1$ APs powered on and a number of users ranging between $n_{K+1} + 1$ and $N_{K+1} - 1$. The eleventh equation is similar to the previous one but for stage $N-1*$. In this case, there is no upper limit in the number of users since there is not any remaining AP to boot up. The twelfth equation models the transition from stage $K*$ to states where the number of APs on is below $K+1$. This implies that during the booting up of the $K+1$ -th AP several users have left forcing the system to switch off some APs. The last equation is similar to the previous one and corresponds to transitions to states where only one AP is on.

With the previous equations, the DTMC can be easily solved to obtain the stationary distribution of the state probabilities, that we name $P(i, K)$ (or $P(*, K)$) hereafter. With these, the

stationary probability of each state of the semi-Markov process, $\Phi(i, K)$ (or $\Phi(*, K)$) are

$$\Phi(i, 1) = \frac{P(i, 1)}{\Omega \cdot (\lambda + \mu)}, \quad 0 \leq i < N_1 \quad (4.62)$$

$$\Phi(i, K) = \frac{P(i, K)}{\Omega \cdot (\lambda + K\mu)}, \quad n_K < i < N_K, \quad 1 < K < N \quad (4.63)$$

$$\Phi(i, N) = \frac{P(i, N)}{\Omega \cdot (\lambda + N\mu)}, \quad i > n_N \quad (4.64)$$

$$\Phi(*, K) = \frac{P(*, K)T_{on}}{\Omega}, \quad 1 \leq K < N \quad (4.65)$$

with

$$\begin{aligned} \Omega = & \sum_{j=0}^{N_1-1} \frac{P(j, 1)}{\lambda + \mu} + \sum_{K'=2}^{N-1} \sum_{j=n_{K'}+1}^{N_{K'}-1} \frac{P(j, K')}{\lambda + K'\mu} \\ & + \sum_{j=n_N+1}^{\infty} \frac{P(j, N)}{\lambda + N\mu} + \sum_{K'=1}^{N-1} P(*, K')T_{on}. \end{aligned} \quad (4.66)$$

The stationary probabilities of stages $K*$ are directly $\mathcal{P}_{K*} = \Phi(*, K)$, while for stages K we have

$$\mathcal{P}_1 = \sum_{i=0}^{N_1-1} \Phi(i, 1), \quad (4.67)$$

$$\mathcal{P}_K = \sum_{i=n_K+1}^{N_K-1} \Phi(i, K), \quad (4.68)$$

and

$$\mathcal{P}_N = \sum_{i=n_N+1}^{\infty} \Phi(i, N). \quad (4.69)$$

Once these terms are known, we can compute the average power P with (4.55) and (4.54). The average service time T_s is obtained with (4.56) as well, but in this case N_u is

$$N_u = \sum_{i=1}^{\infty} i \left(\sum_{K=1}^N \Phi(i, K) + \sum_{K=1}^{N-1} \pi_i^{(K*)} \mathcal{P}_{K*} \right), \quad (4.70)$$

with $\pi_i^{(K*)}$ the same as in (4.59).

To end this Section, we present in *Algorithm 2* the different steps required to obtain the performance figures of the system. As can be seen, in this case we avoid the presence of loops.

Algorithm 2 Solution to the approximate model

-
- 1: Set $\pi^{(K^*)}(0)$ with (4.60)
 - 2: Solve (4.31) and (4.32) to obtain $\mathbf{L}^{(K^*)}(T_{on})$ and $\pi^{(K^*)}(T_{on})$
 - 3: Solve the DTMC with transitions given by (4.61) to obtain $P(i, K)$ and $P(*, K)$
 - 4: Compute (4.62)-(4.66) to obtain $\Phi(i, K)$ and $\Phi(*, K)$
 - 5: Obtain $\mathcal{P}_1$, $\mathcal{P}_K$ and $\mathcal{P}_N$ with (4.67)-(4.69)
 - 6: Obtain N_u with (4.70) and N_{AP} with (4.54)
 - 7: Compute T_s with (4.56) and P with (4.53)
-

4.4. Numerical Results

We next present a numerical evaluation of a RoD system in terms of the performance figures considered, namely, the average service time T_s and the power consumed by the infrastructure P . To this end, we compute these two variables for a variety of scenarios, these being defined in terms of the network load or the configuration of the RoD scheme (given by the parameters M , ρ_h , ρ_l). In the simulation results presented, we compare the results of our approximate model against the ones obtained via simulation,⁸ while in Section 4.4.3 we assess the computational complexity of this model against the accurate one.

Throughout all simulations, we consider the following scenario:⁹ (i) various APs can be simultaneously activated (instead of only one, as assumed in the analysis); (ii) there is no complete overlap of the coverage areas: we assume a deployment centred around one AP with a 10 m coverage radius that is always on, and $N - 1$ APs with the same coverage radiuses that are randomly deployed within a 4 m circle centred around the first AP and that will be opportunistically (de)activated; and (iii) users are not static but follow the classical *random waypoint model* [83], selecting a novel destination at random after reaching the previous one, and moving at a speed that is randomly chosen between 0.3 and 0.7 m/s. We further assume that there are up to $N = 10$ APs available,¹⁰ that a single AP consumes 3.5 W when active (which corresponds to the average power consumed by a Linksys device [67]) and zero otherwise, and that $\mu = 0.1 \text{ s}^{-1}$.

4.4.1. Impact of network load

We first analyze the power consumption as the network load $\rho = \lambda/(N\mu)$ varies. To this end, we fix a target distribution of $M = 5$ users per AP and the following two configurations of the (de)activation thresholds $\{\rho_h, \rho_l\}$: $\{100\%, 30\%\}$ and $\{50\%, 25\%\}$, the former being more “reluctant” to increase the number of APs when the network load increases. To understand the

⁸Our approximate model is solved numerically using Octave (<https://www.gnu.org/software/octave/>), while simulation results are obtained from a discrete event simulator written in C++.

⁹Note that this scenario relaxes some of the simplifying assumptions behind our model, and thus allows to assess the impact of such assumptions on the results.

¹⁰This is a reasonable number for dense scenarios: for instance, an auditorium with 360 users, each of them demanding 3 Mbps for HD video, would require 31 802.11n APs with a throughput of 35 Mbps (data taken from [76]). Results of the same order of magnitude are obtained in [84, 85].

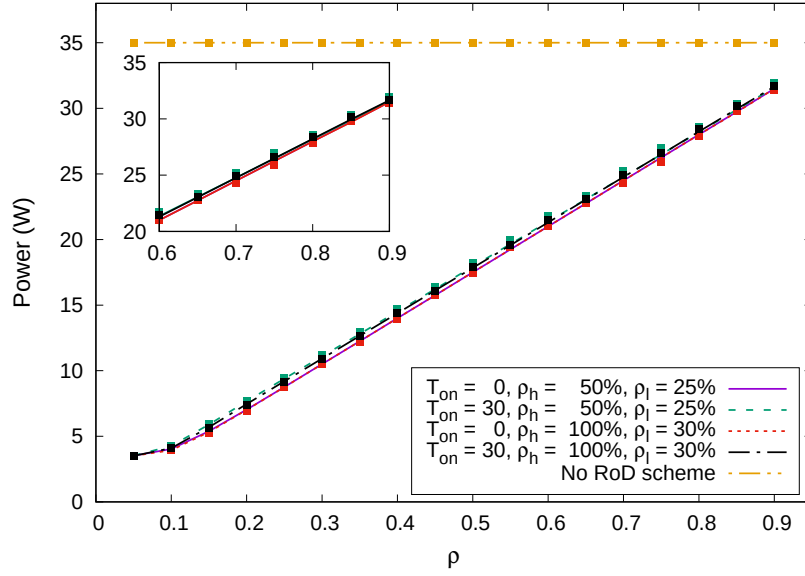


Figure 4.8: Average power consumption vs. network load.

impact of T_{on} on performance, we consider the cases of zero and 30 s start-up times. We plot the computed figures of P in Fig. 4.8, where we use squares for the simulation values (average of 10 simulation runs, each consisting of more than 100k users) and lines for the analysis.

According to the results, the power consumption is monotonously increasing with the network load, with the analysis practically coinciding with the simulation values, with some minor deviations (approx. 1.8%) for high loads (we depict a zoomed version of the figure for these values). Considering the relative performance of each configuration, for the case of $T_{on} = 0$ the results overlap, while for the case of $T_{on} = 30$ s, the policy that is “more eager” to power APs leads to higher power consumption.

We next analyze the performance in terms of service time and the trade-off with power consumption. To this end, we plot T_s vs. P in Fig. 4.9, with each simulation point corresponding to a different value of ρ , which varies from 0.05 to 0.9 in steps of 0.05. Here we also provide for comparison the “ordinary” case of no RoD scheme (all APs always on), which leads to the smallest service times and the largest power consumptions. As in the previous case, the analysis accurately predicts simulation results, with differences below 2.5%. The figure also illustrates that the service time is a monotonous increasing function of the load: steep for $\rho \leq 0.3$, which is caused by the “drastic” impact of powering on an AP when the number of active resources is relatively low, and then more gradual until $\rho \approx 0.9$. Concerning the impact of the considered configurations, for the same value of the $\{\rho_l, \rho_h\}$ parameters, the non-zero start time has an impact of approx. 5 s for the more dynamic configuration, and approx. 2 s for the more “reluctant” configuration, while the impact of the activation policy results in differences of approx. 12 s.

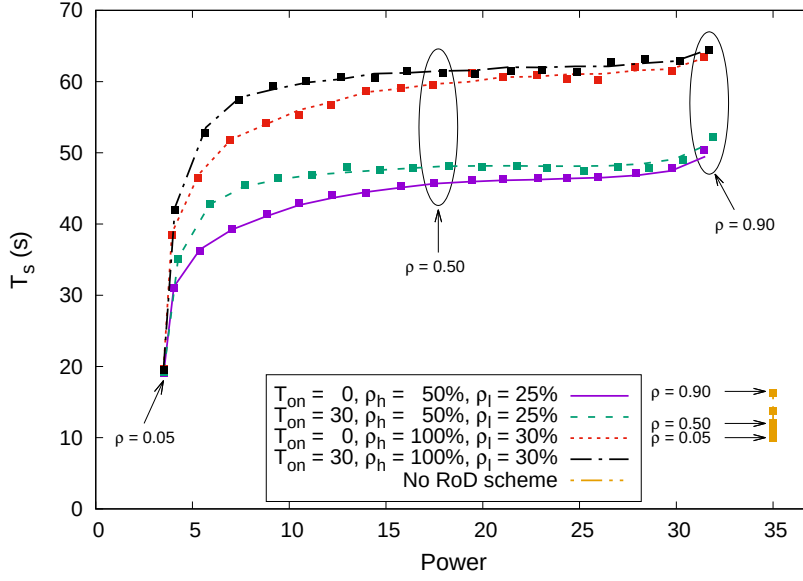


Figure 4.9: Average service time vs. average power consumption.

4.4.2. Impact of RoD configuration

Next, we consider the case of a fixed value of $\rho = 0.5$, and compute the service time and power consumed for the two considered $\{\rho_h, \rho_l\}$ configurations and different values of the target number of users per AP M . We plot the service time and the power consumption as a function of M , with the results being depicted in Fig. 4.10.

For the case of the service time (Fig. 4.10, top), again simulation results practically coincide with the analysis. The larger M is, the longer the service times are, as users are more likely to share the capacity of a single AP before activating new resources. In fact, the relation is practically linear, e.g., when M changes from 5 to 10, the service time doubles for all considered scenarios: as there are, on average, more users per AP, the service times will be longer.

For the case of the power consumption, the resulting values are depicted in Fig. 4.10 (bottom). Here, we note that the results for both RoD configurations for $T_{on} = 0$ overlap, and result in a constant power consumption regardless of the value of M . The reason for this behaviour is that, as M increases, more users per AP are required to power on additional resources, but also longer service times will result, leading to more users in the system. In fact, the power consumption of 17.5 W implies that, on average, 5 out of the 10 available APs are on, which matches the $\rho = 0.5$ load. When the start-up times are non-zero, there is a small reduction of P as M increases. The reason for this is that, on average, the system is less likely to power on additional APs, which incurs in the overhead of the start-up process. Finally, we also note that simulation values are very close to those from the analysis, with relative differences of approx. 4% (the smaller M is, the larger the differences are, as the impact of non-perfect overlap of coverage areas is more noticeable for a small number of users).

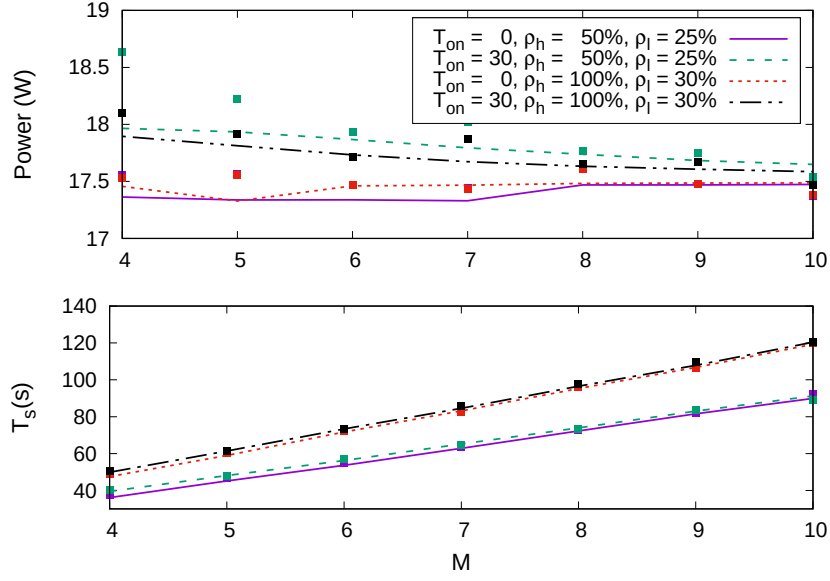


Figure 4.10: Average service time (top) and power consumption (bottom) vs. target number of users per AP.

4.4.3. Computational complexity

We next estimate the computational complexity of obtaining the numerical solution for the exact and the simplified analysis. To this end, we assume a scenario with $N = 10$ APs, fix $M = 4$, and consider different configurations of T_{on} , ρ , and $\{\rho_h, \rho_l\}$ parameters. For each set of parameters, we compute the average service time T_s and power consumption P using the exact and the simplified analysis, as well as the time required to compute these values for each case. We note that we use Octave to compute the numerical solution for these analysis, running over an Intel® Xeon® X5550 @2.67GHz with 48 GB RAM, and therefore our comparison serves to illustrate the relative differences in complexity, and not absolute values.

We provide in Table 4.1 the results of the above computation. More specifically, we provide in the Table, for each considered configuration, the relative difference between the two analyses in terms of service time (denoted as ΔT_s) and power consumption (denoted as ΔP), and the corresponding computation times. There are two main observations from the results: (i) on the one hand, for both power and service time figures, the resulting differences between the numerical analyses are at most 3%, and in many cases well below 1%; and (ii) on the other hand, for the computational times, there are two orders of magnitude of difference between them in all but for two cases. Finally, it is also worth noting that, for the case of the exact analysis, computational times grow with T_{on} , which confirms to some extent that the K^* stages are responsible for the computational burden.

Table 4.1: Relative differences and computational times of the exact and simplified analyses.

T_{on} (s)	ρ_h, ρ_l	ρ	Error		Comp. time (s)	
			ΔT_s	ΔP	Exact	Simpl.
0	{0.5, 0.75}	0.25	$\approx 0\%$	$\approx 0\%$	104.28	2.35
		0.75	$\approx 0\%$	$\approx 0\%$	102.34	2.35
	{1, 0.7}	0.25	$\approx 0\%$	$\approx 0\%$	102.78	2.40
		0.75	0.06%	0.04%	104.58	2.30
15	{0.5, 0.75}	0.25	0.22%	0.14%	167.69	3.62
		0.75	0.91%	0.58%	269.43	5.77
	{1, 0.7}	0.25	0.02%	0.01%	166.94	3.9
		0.75	0.17%	0.14%	276.20	5.65
30	{0.5, 0.75}	0.25	1.97%	1.05%	320.52	6.75
		0.75	3.09%	1.80%	519.96	11.36
	{1, 0.7}	0.25	0.41%	0.23%	316.62	7.03
		0.75	1.17%	0.66%	541.04	11.72

4.4.4. Realistic traffic model

To analyze the impact of the simplifying assumptions on the traffic model of our analysis, in the following we compare the results obtained with our analysis against those obtained from simulations with a “realistic” traffic model. In particular, we follow [?] and assume that when a station joins the WLAN, it performs a random number of download requests that follows a BiPareto distribution. The length of each download also follows a BiPareto distribution, and the interarrival time of requests follows a lognormal distribution. We fix the average number of requests to 10, with the following parameters of the BiPareto distribution: $\alpha = 0.06$, $\beta = 1.73$, $c = 6.61$ and $k = 1$; the lognormal distribution is simulated with parameters $\mu = 0.34$ and $\sigma = 0.63$; and the request lengths are initially modelled with parameters $\alpha = 0.0$, $\beta = 2.13$, $c = 20.0$ and $k = 1.5$ (which leads to an average download size of 30 MB), while the user arrival rate is Poissonian at a rate ranging from 0.05 to 0.9 s⁻¹.

We show in Fig. 4.11 the resulting average service time vs. power consumption for different configurations of the RoD scheme and values of T_{on} , where (like in Fig. 4.9) we vary the load from 0.05 to 0.9 in steps of 0.05. We observe that the accuracy of the model worsens as the service rate increases: the deviations are smaller than 5% for $\rho < 0.8$ but notably higher as the system gets closer to saturation. We conclude from these results that overall the accuracy of the model is reasonable for the range of loads of interest (i.e., sufficiently far from congestion).

4.4.5. Optimal Configuration of a RoD scheme

While the exact analysis incurs in a notable complexity, we have seen that the simplified analysis is able to compute the performance figures of a RoD scheme in an affordable manner while keeping a notable accuracy. In this way, it can be used, for instance, to compute the optimal configuration of a RoD algorithm, given a set of estimated network conditions, these being expressed

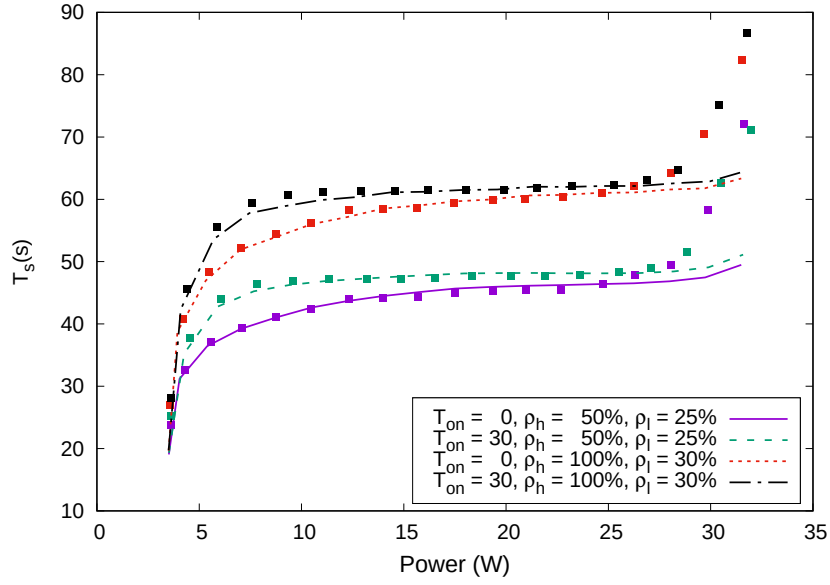


Figure 4.11: Average delay vs. power consumed with non-exponential service demands.

in terms of λ and μ . In the following, we present one example of such configuration algorithms, although we restrict ourselves for simplicity to the considered RoD policy (although there could be many others) and a simple optimization criterion. Our optimization scheme works as follows. Given an estimation of the network conditions, we set a bound on the maximum service time $T_{\max}$, and perform a sweep on the configuration space $\{M, \rho_h, \rho_l\}$ to look for the configuration that minimises power while guaranteeing an average service time T_s below $T_{\max}$. In our search, M goes from 2 to 10 in steps of one, while ρ_h and ρ_l go from 0.05 to 1.25 in steps of 0.05.

The configuration resulting from this search and the corresponding performance figures are given in Table 4.2 for three different service rates $\mu = \{0.05, 0.1, 0.2\} s^{-1}$ and the corresponding three service time bounds $T_{\max} = \{80, 40, 20\} s$, respectively. If we compare the consumed power with a reference scenario of the 10 APs always on (i.e., consuming 35 W), the reduction is quite considerable, ranging between 25% and 75% depending on the network load. Finally, it is also worth remarking that T_{on} has a non-negligible effect on the resulting configuration parameters.

4.5. Conclusions

As shown in the previous chapter, reconfiguring the hotspots of network deployment will lead in energy efficient gains under certain condition. In this chapter, we have gone a step further analyzing the Resource-on-Demand schemes. We have proved that these schemes are required in dense networks to adapt to the varying load while maintaining an energy efficient performance. Moreover, we have developed an analytical model of these schemes that, in contrast to previous publications, accounts for the non-zero start-up times of real hardware. We have also presented a

Table 4.2: Performance an optimal configurations of a RoD scheme.

$\mu(s^{-1})$	ρ	$T_{on}(s)$	M	ρ_h	ρ_l	$T_s(s)$	$P(W)$
0.05	0.25	0	3	1.20	0.55	75.93	8.76
		15	4	0.75	0.30	79.17	8.96
		30	3	1.20	0.30	74.77	9.16
	0.5	0	3	1.20	0.30	76.89	17.33
		15	3	1.15	0.30	76.60	17.55
		30	3	1.15	0.30	78.41	17.81
	0.75	0	3	1.20	0.30	77.44	25.35
		15	3	1.20	0.30	79.75	26.00
		30	2	0.95	0.45	53.00	25.34
0.10	0.25	0	3	1.20	0.55	37.96	8.76
		15	3	1.20	0.30	37.38	9.16
		30	3	1.20	0.30	39.98	9.50
	0.5	0	3	1.20	0.30	38.44	17.33
		15	3	1.15	0.30	39.21	17.81
		30	2	0.95	0.45	28.41	17.88
	0.75	0	3	1.20	0.30	38.72	25.35
		15	2	0.95	0.45	26.50	26.34
		30	2	0.95	0.45	28.20	25.98
0.20	0.25	0	3	1.20	0.55	18.98	8.76
		15	3	1.20	0.30	19.99	9.50
		30	3	0.80	0.30	19.87	10.12
	0.5	0	3	1.20	0.30	19.22	17.33
		15	2	0.95	0.45	14.21	17.88
		30	2	1.00	0.45	16.36	17.61
	0.75	0	3	1.20	0.30	19.36	25.35
		15	2	0.95	0.45	14.10	25.98
		30	2	0.95	0.45	15.90	25.42

simplified model, whose computational times are approx. 50x shorter while maintaining relative errors below 3%. We have illustrated the practicality of this simplified model with a simple algorithm to derive the optimal configuration of a RoD scheme. In the next chapter, we define an Software Defined Networking (SDN) architecture that validates the feasibility of the RoD operation.

Chapter 5

Proof of Concept

In the previous chapter, we have presented an analytic model and performance evaluation in large temporal scale for dense mobile network deployments, presenting additionally the optimal configurations of the model in order to find a trade-off between energy consumption and network performance. In this chapter, we introduce the OFTEN framework (Open Flow framework for Traffic Engineering in mobile Networks with energy awareness), a novel platform based Software Defined Networking (SDN) paradigm with some additional facilities that allows us to implement Resource-on-Demand (RoD) schemes and Traffic Engineering (TE).

Such framework is design to *i*) support heterogeneous technologies (such as cellular or WLAN) in a transparent way, hiding the specificities of the technologies to the TE tool, *ii*) integrate in a single decision point different types of communication that have traditionally been handled separately, such as device-to-device and infrastructure communications and, *iii*) address new functions that are not needed in wired networks, such as switching off those points of access that are not needed at a given point in time, and thus reduce energy consumption.

The key point of this framework is the centralised controller that (a)periodically performs TE optimizations based on a given set of policies. By using a common API for all technologies, this controller issues commands that are oblivious to the technologies underneath. Furthermore, the controller does not only serve to manage the mobile network infrastructure but can also reach mobile terminals and even trigger device-to-device communications between them in order to offload the network infrastructure.

To conclude, we have implement this framework in real world prototype to validate our approach.

5.1. Architecture

We propose the architecture illustrated in Fig. 5.1, which consists of a number of technology-agnostic elements plus some technology-specific modules. We start with the former:

- The first element is the database containing the current **network vision**, i.e., the *nodes*

of the network, the active *links* connecting them, the potential links that can be used and their capacity as well as the traffic demand.

- Based on this network vision, the **optimiser** module runs (a)periodically (e.g., every 5 minutes or after a major change in network conditions) to obtain the configuration that maximises performance, according to a set of given policies that trade-off energy consumption and performance.
- The configuration resulting from the optimiser module is passed to the **controller** via the *northbound* interface; this interface is not shown in Fig. 5.1 and depends on the specific OpenFlow controller chosen (we leave it outside the scope of our architecture).
- Finally, the controller implements this configuration through **OpenFlow++**, a technology-agnostic interface that extends the OpenFlow protocol (OFPT) to support the required functionality for mobile networks (described in the following sections).

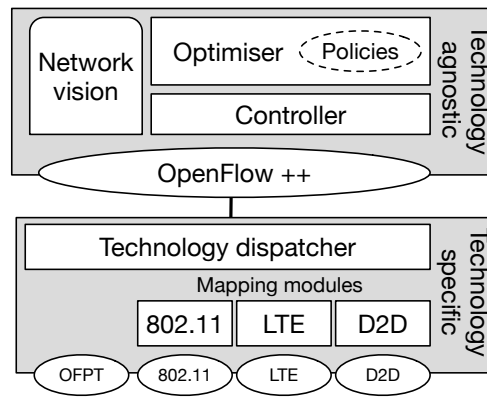


Figure 5.1: Open Flow-based SDN architecture for heterogeneous networks.

One key feature of the proposed architecture is the ability of the upper layer modules described above to operate on a technology-agnostic manner, which allows to (i) manage the entire heterogeneous network in an integrated way, allowing for a joint optimization of all the technologies; and (ii) easily adapt network operation to new requirements/objectives, by simply modifying the technology-agnostic modules. This functionality is enabled by the following lower layer modules, which are technology-specific:

- A **technology dispatcher**, which is responsible to re-direct the OpenFlow++ primitives to appropriate technology-specific module.
- Technology-specific **mapping modules**, which convert the OpenFlow primitives into the primitives of the corresponding technology and viceversa; among others, this includes the setting of new configurations and update the network vision.

Another key feature is that the architecture can be extended to support new technologies in a non-disruptive way: to enable the use of a new technology, we only need to design the specific mapping module and a minor extension to the Dispatcher to identify when a command is from/to that specific technology. We note that, because of this centralised management of all technologies in the network, the controller might suffer from scalability issues, a general concern in SDN scenarios. To ease this burden, one possible solution would be the definition of hierarchical areas [86, 87].

We next describe the technology-agnostic operation of the upper modules and how the OpenFlow++ interface is used and extended to support this vision (we summarise in Table 5.1 the required extensions to OpenFlow), while the technology-specific operation is described in the next section.

Primitive/parameter	Use by OFTEN
OFPT_HELLO	Announce new nodes and links. Link unavailability is signalled with TCP FIN.
OFPT_FEATURES_REPLY	Announcement of non-OF features (e.g. de-activation) via the reserved field
OFPT_FLOW_MOD	Issue mobility commands, i.e., change point of attachment
OFPT_EXPERIMENTER	Switching on and off of resources
OFPT_METER_MOD	Add/remove meter configuration
curr_speed, max_speed	Capacity announcement
counters, meter_bands	Measure throughput and trigger optimization module

Table 5.1: Extensions to OpenFlow introduced by OFTEN.

5.1.1. Technology-agnostic operation

One of the key challenges of our architecture is to achieve a technology-agnostic vision of the network. This allows the optimiser and controller to operate on abstract “nodes” and “links” instead of, e.g., 802.3az links or eNBs. More specifically, the challenge is to map the actual physical network, like e.g. the one illustrated in Fig. 5.2 (top), into an abstract vision (bottom). While the physical network is composed of wired links in the backhaul, relatively stable links from mobile terminals to cell towers, higher capacity but more dynamic connections to 802.11 Access Points, potential device-to-device links, etc., in the abstract vision there are nodes and links, each with a different capacity, that can be reconfigured and switched on and off as required by the optimiser.

The proposed architecture is flexible to accommodate different TE mechanisms for the optimiser, depending on the computational resources availability, complexity of the network, and periodicity/timeliness of the operation of the optimization (see Section IV.C for a description of the mechanism used in our implementation). Furthermore, the optimization of the network does

not have to be changed whenever a new technology is introduced, and only requires mapping the features of the new technology to our abstractions.¹

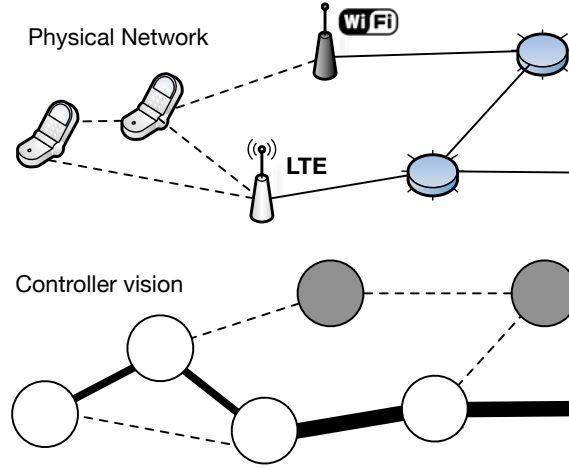


Figure 5.2: Physical network (top) and abstract vision (bottom). Grey circles denote nodes in power-saving mode, dotted lines represent available links, and solid lines represent used links.

The mapping between the actual network and the abstract one is made in the technology dispatcher module, which performs the association between the mapping modules and the OpenFlow++ interface. This interface is the central element of the architecture that supports the following functionality with technology-independent primitives: *(i)* the maintenance of the information in the database, including changes to link capacities, reachability, etc., *(ii)* the control of the forwarding tables, and *(iii)* the (de)activation of resources. We next describe the OpenFlow commands upon which we rely, as well as the extensions required, to support these features.

5.1.2. Maintaining the network vision

The first challenge is to maintain the list of nodes that compose the network, which in our case appear and disappear more frequently than in “traditional” (wired) networks, due to users’ mobility and wireless propagation. For the case of nodes connected to the network, i.e., APs or eNBs, they just have to establish a connection to the default OpenFlow transport protocol 6653 via TLS or TCP, and then perform the usual `OFPT_HELLO` exchange. For those nodes that can be (de)activated, we need to extend the database to announce this feature, which we do via the `reserved` field in the `OFPT_FEATURES_REPLY` message.

For the case of nodes that are one or more hops away from the wired infrastructure, we do not require them to implement any (major) modification and, in particular, to support OpenFlow. However, as they have to appear as nodes in our vision, we require that the PoAs act on the behalf of the terminals, and register them with the controller (following the standard procedure)

¹Of course, to achieve this it is critical that our abstraction is general enough to cover the functionality provided by the new technology, as otherwise the optimiser would need to be adapted to the novel functionality provided.

whenever they detect there is a new node or a new candidate link. When no link towards a terminal is available, the connection is terminated via e.g. a `TCP FIN` message. Thus, in order to keep the list of nodes available in the network, our architecture does not need to introduce changes to the OpenFlow specification, but only to ensure that the database of (de)registered nodes is updated when required.

Similarly, in order to announce the capacity of the links that connect two or more nodes, we can rely on the structures already defined by the OpenFlow specification, i.e., the `curr_speed` and `max_speed` parameters, which define the current and maximum bitrates, respectively, of a port. Whenever the capacity of a given wireless link changes (because of e.g. WLAN interference, a node moves away from the eNB), the node responsible for a link needs to modify the value of the corresponding parameters.

Finally, the usage of the links carrying data can be easily tracked with per-flow counters. By periodically polling these counters with read operations, the database can identify which links are becoming congested and which ones are being under-occupied and could support more traffic. This can then trigger the corresponding optimization. We note that an alternative implementation could be based on the use of per-flow meters; indeed, by specifying `meter bands`, we can trigger an action when certain thresholds are passed.

5.1.3. Installing a new configuration

For wired links, the setting of forwarding paths does not require to modify the default OpenFlow operation. With our abstraction, a wireless terminal can be considered as a node with a number of ports but only one active forwarding entry at a time, which corresponds to the selected point of attachment. In this way, changing the point of attachment only requires modifying a flow entry via the `OFPT_FLOW_MOD` message, e.g. to change of the Access Point the node is associated with, or the use of the cellular link. As we describe next, the Technology Dispatcher decides the technology specific module that handles the commands and triggers the protocol-specific operations to implement the change. As in the previous case, there is not need to specify new OpenFlow primitives.

5.1.4. Switching on/off resources

Energy-efficient operation of a network requires the ability to power on and off resources as required [88]. Accordingly, our architecture has been designed to provide such support. For the corresponding set of operations, we cannot rely on or extend the default OpenFlow primitives, and therefore we need to specify new primitives. To do this, we rely on the “experimenter” symmetric messages (`OFPT_EXPERIMENTER`), which are used for those nodes that upon registration announced that they supported deactivation, in addition to the time it takes to switch between states (so the the controller can preemptively activate resources) and the corresponding power consumption (to duly optimase energy consumption).

Following the above, we extend the OpenFlow set of primitives with a pair of commands, `switch_on` and `switch_off`, to power on and off (respectively) a given node. As we will describe in our testbed, these primitives can be used even when nodes do not support a sleep state, but are connected to e.g. a switched rack power distribution units (PDU), which can be remotely controlled via a network connection.

5.2. Mapping to technologies

The architecture enables an integrated traffic management of a heterogeneous mobile network, through the use of the OpenFlow++ primitives. We next describe how these primitives are mapped back-and-forth into technology-specific functionality thanks to the technology dispatcher, which acts as a relay of the messages between the OpenFlow++ API and the modules doing the mapping. *In nuce*, we require the ability to:

- Detect when new nodes and links are available, including the potential points of attachment to the network.
- Estimate the available capacity of the wireless links.
- Change the point of attachment of a terminal without disrupting the traffic served.

In what follows, we describe how the above is supported by the dispatcher and the current wireless technologies, by just introducing minor extensions to their operation.

5.2.1. Technology dispatcher

New nodes or available links are announced in a technology-specific manner to this module (as detailed next), which then performs the translation to the OpenFlow++ API. With this module, a mobile terminal detected by a set of access points and an eNB (i.e., multiple announcements) is registered only once as a node, but with a set of candidate links towards existing nodes. Given that wired nodes may support different deactivation techniques (e.g., wake-on-LAN, switched power distribution units), this module needs to be aware of the specific technique supported in order to issue the corresponding commands when needed. The capacity of the wireless links is computed by each technology as described next, and then passed via the OpenFlow++ API using the parameters described above.

Changing the default forwarding table of a terminal corresponds to performing a handover, which can be intra- or inter-technology. The former is handled within each technology using its own mechanisms, while for the latter we rely on the 3GPP support for multiple radio access networks (RANs), including those that are non-cellular. The mobile 3GPP architecture centralises the support of inter-RAN handovers on the cellular network (more specifically, on a set of network entities that may act as control and data plane anchors for the different handover scenarios). This is based on the assumption of full cellular coverage.

5.2.2. 802.11 mapping

In Wi-Fi networks there are at least three mechanisms to detect when a link towards a mobile terminal is available: *(i)* the `probe request` messages the mobile periodically sends on different channels to detect known or new Access Points (APs), which can also trigger the presence of a new node that is duly reported to the technology dispatcher; *(ii)* passive scanning mechanisms; and *(iii)* the `Neighbour Report` message exchange from the recent 802.11k amendment, already supported by new devices such as e.g. iPhones, which is used by mobile terminals to learn about APs in the surroundings, and by APs to gather measurements about the quality of the channel towards other APs, which are then reported to the network.

The estimation of the capacity of a link is supported by these measurements, building on e.g. the usual mappings of signal quality to maximum throughput, which are reported to the network vision database. In case a group of nodes contend in a WLAN, the AP can derive the achievable capacity thanks to the use of the *linearised capacity* model proposed in [89], which abstracts the specifics (i.e., contention) of the channel access into a simple linear-based model.²

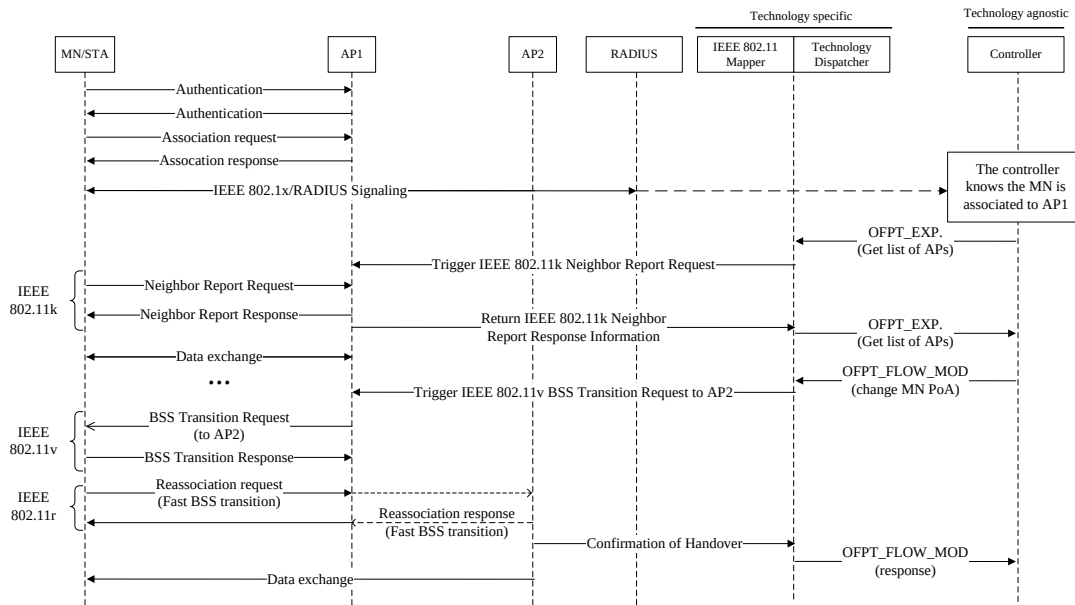


Figure 5.3: Simplified handover operation.

Changing the point of attachment of a Wi-Fi node while providing a seamless experience results is very challenging, given that in 802.11 *(i)* the mobile terminal typically selects its “best” point of attachment, and *(ii)* the signalling for carrier-grade operation is relatively complex. Still, thanks to the recent 802.11v and 802.11r amendments, this can be achieved with minor disruption of the service (this is validated by our prototype of Section 5.3). Fig. 5.3 illustrates a simplified

²Note that the proposed linearised capacity model can also be used to abstract the capacity of any technology, including OFDM, CDMA, etc.

version of the signalling occurring on the access network. Next, we describe how these interactions support the changing of the point of attachment when triggered by the OpenFlow command.

Following Fig. 5.3, when a terminal powers on, it authenticates and associates with the best AP (i.e., AP1), and performs the `Neighbour Report` exchange to learn about the APs in the vicinity belonging to the same network. During these operations, the APs that overhear the node activity report on the different links available to the mobile terminal, and update the database accordingly (among these, AP2 in the Figure). Assuming at some point that the optimiser decides that it is better if the mobile node (*MN*) associates with AP2, the controller issues a `OFPT_FLOW_MOD` primitive to change the (only) default forwarding entry of *MN*, from node *AP1* to node *AP2*. This primitive is processed by the 802.11 module with the controller, which issues a command to AP1 to trigger the 802.11v `BSS Transition Request` message, so the *MN* re-associates with AP2. Thanks to the use of 802.11r, this re-association is noticeably shorter than the original one. As we will see in the next section, the controller can duly issue other OpenFlow primitives to minimise the impact on performance.

5.2.3. Cellular mapping

We next describe the guidelines to implement a technology-specific module similar to the one described above for the case of cellular networks. While 3GPP-based networks involve more complex procedures, they also tend to favor a centralised control and estimation of resources, and therefore we expect that the design of this module is simpler than in the Wi-Fi case. However, given that cellular networks are designed for large deployments, one key difference with Wi-Fi is on maintaining user location.

If we focus on packet switched communications in a cellular network the list of active users within a cell is available at the Mobility Management Entity (MME), while the inactive users are less accurately located.

The above challenges the implementation of infrastructure-on-demand schemes, as a group of inactive users could suddenly change to be active in a certain geographical area, and an aggressive energy-saving policy might have powered off too many Base Stations to promptly react to the demand. Except for this, cellular networks readily provide the means to support the requirements of the extended OpenFlow interface: active UEs periodically report the quality of the channel towards other Base Stations, and network-initiated handovers are supported.

5.2.4. Device to device mapping

Our architecture also supports device-to-device communications, where mobile nodes opportunistically share wireless links for a better performance. However, in contrast to the previous cases, this paradigm requires introducing modifications to the mobile terminals. To that aim, we have adapted the SOLOR framework [90], which builds on Wi-Fi Direct to opportunistically create D2D groups, to communicate with the technology dispatcher to announce link availability and

support the set-up and release of this links. Given that SOLOR is a decentralised mechanism, the adaptation consists basically on centralising the control of the modules.

5.3. Proof of concept

To validate the feasibility of our approach, we have prototyped our architecture in a small testbed that includes the technology specific modules for Wi-Fi. The testbed, deployed in an office setting, is depicted in Fig. 5.4 and consists of one controller, an OpenFlow switch, two APs, two MNs, a correspondent node (CN) to support traffic generation, and a switched PDU to support the (de)activation of APs via HTTP requests.

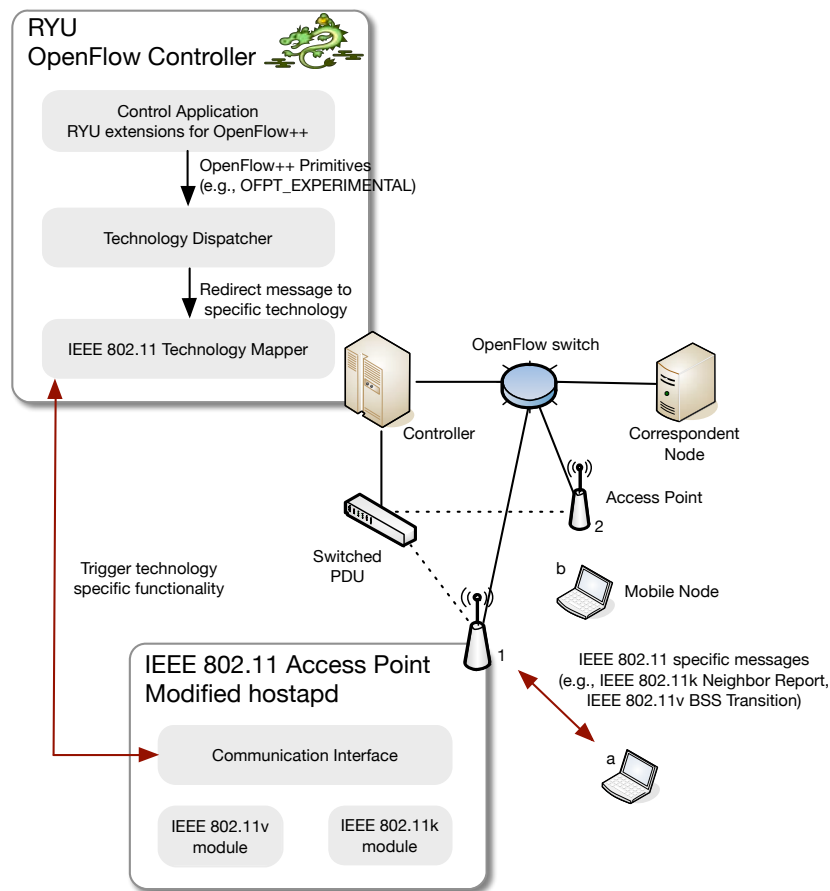


Figure 5.4: Deployed testbed and implemented modules and interfaces.

5.3.1. Implementing seamless NIHO

One of the benefits of an OpenFlow-based architecture is that it supports the implementation of a variety of mobility protocols. For simplicity, we decided to implement the following sequence of commands whenever the controller decides to change the point of attachment of a

MN: (1) activate at the switch *bicasting* of traffic between the CN port and the ports the APs are connected to; (2) issue the 802.11v BSS Transition Request; (3) wait until the handover is completed, and inform the controller accordingly; and (4) stop the bicasting of traffic at the switch and re-configure the forwarding tables accordingly.

5.3.2. Software setup

The controller is a desktop machine running Linux and the Ryu SDN framework,³ which we extend to support the proposed OpenFlow++ API, and two Python libraries: one to map the `switch_on/off` commands to the switched PDU, and another one for the 802.11-specific mechanisms. The controller also runs a MySQL database⁴ to keep the network status. Our controller uses a variation of the TE algorithm that we designed in [89], which is based on an integer programming formulation and minimises the number of nodes required to support a set of flows at a reduced computational cost. Other TE schemes could be supported, based on, e.g., flow management policies or traffic measurement requirements (see [91] for a recent survey).

The APs are small PCs running Linux and extended to support the required functionality as follows. We have modified the widely used `hostapd` demon⁵ to set up a connection with the 802.11 module at the controller, so that the AP can report changes in the network conditions (which are updated to the database) and can set up new configurations. When the controller issues a change on the default forwarding path of a node, the 802.11 module translates this primitive to an 802.11k request to perform channel measurements (so that the mobile updates the list of available APs) and an 802.11v command to change the point of attachment. An overview of the developed software modules and interfaces is depicted in Fig. 5.4.

The CN is a regular desktop machine, while the MNs are small PCs, like the switch, which runs OpenVSwitch. Finally, we set up an additional desktop machine, not shown in the picture, running the FreeRADIUS server⁶ to enable the use of WPA2-enterprise as required by Wi-Fi Passpoint.

5.3.3. Validation and support of infrastructure on demand

To validate that the controller activates resources as traffic requirements vary, we perform the following experiment. We first generate a TCP flow between the CN and node *a*, which is associated with AP 1, while AP 2 is inactive. At $t = 20$ s, we generate another TCP flow between the CN and node *b*, which saturates the capacity of the link. Thanks to periodic measurements, the controller decides 3 s later that more resources are required, and switches on AP 2.⁷ Once

³<http://osrg.github.io/ryu/>

⁴<http://www.mysql.com>

⁵<http://w1.fi/hostapd/>

⁶<http://freeradius.org>

⁷We note that the focus of our validation is on the framework and not on the specific mechanism to provide infrastructure on demand, which we acknowledge could be improved.

this AP is available, it notifies the controller, which then immediately issues a handover trigger to node b that associates with the new AP at $t = 45$ s.

The results of this experiment are depicted in Fig. 5.5, where we plot the per-flow throughput (bottom) and the total throughput (top) as measured by the Wireshark tool. According to the figure, the first TCP flow achieves at first about 20 Mbps, but when the second flow appears, its throughput is reduced to 5 Mbps, while the former gets about 15 Mbps (with some variations due to contention). Once this has been moved to AP 2, the flow from node a gets again about 20 Mbps, while the flow from node b gets about 20 Mbps as well.

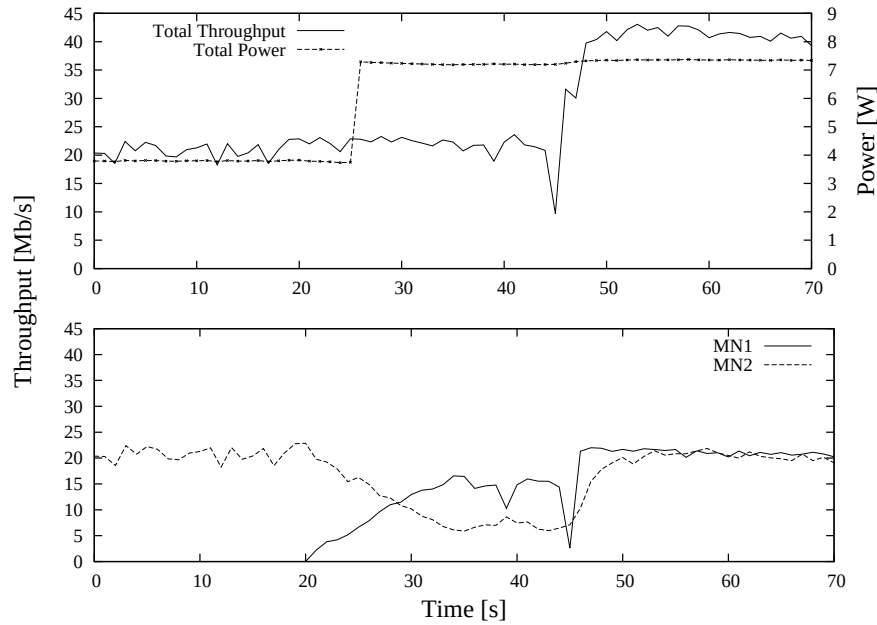


Figure 5.5: Validation of the proof of concept.

We also represent in Fig. 5.5 the estimated energy consumption of the infrastructure, following the model of [67]. As compared with the case of both APs on, our solution reduces energy consumption by 18%, an improvement that comes at the cost of an increased delay, mainly because the time it takes to detect a new flow and power-up the AP. Following the results reported in [88], we estimate that our framework could reduce energy consumption by 30–40% in a campus-wide deployment (note that our experiment corresponds to an “upper bound” in terms of performance reduction, as there is always wireless activity).

5.4. Conclusions

In this chapter, we have proposed a novel SDN architecture to support traffic engineering in mobile networks and RoD schemes previously presented in Chapter 4. Our OpenFlow-based architecture facilitates the support for heterogeneous technologies by making use of abstractions, and enables the use of infrastructure-on-demand schemes via novel primitives to switch on/off

devices as required. The validity of our approach has been demonstrated for the case of 802.11 through a simple prototype.

As extension of this work, we are working on the following challenges: *(i)* we are currently integrating within the prototype, LTE-like environments by adapting open source solutions such as srsLTE or OpenAirInterface (both described and analyzed in Chapter 2), and *(ii)* we are also deploying our framework in a campus-size testbed, to evaluate the performance improvements that can be achieved,⁸ analysing the trade-off between energy consumption and performance when following the optimal configurations obtained in Chapter 4.

⁸We have made our source code available so other researchers and practitioners can perform similar measurements:
<https://oruga.it.uc3m.es/redmine/projects/often>

Chapter 6

Summary and Conclusions

Throughout this thesis, we have carried out a detailed experimental analysis and characterization of the existing softwarized mobile networking platforms, understanding *i*) the capabilities and performance, *ii*) energy footprint in short time scale and, *iii*) the power consumption in large time scale. We have first presented a methodology to evaluate Software Defined Radio (SDR) solutions and second, a characterization of the performance for OpenAirInterface (OAI) and srsLTE, both open source projects. Next, we have presented an energy measurement platform for short temporal scale. To validate our platform, we have analyzed the Rate Adaptation (RA) algorithms in IEEE 802.11 through analytic model and we have compared it with experimental measurements. Furthermore we have extended our analysis to a prototype for C-RAN scenarios with multiple configurations for function splits. Our findings prove the energy management of the device is important and there is still room for enhancements of energy efficiency when considering multiple parameters (MCS, transmission power, bandwidth, etc.). Then, we have elevated the temporal scale of our analysis, introducing the Resource-on-Demand (RoD) schemes. Through numerical analysis, we have discovered that ultra-dense deployments can considerably reduce the energy consumption while maintaining the network performance. Additionally, we have provided the optimal configurations for these trade-offs. Considering these schemes and the lack of real world solutions for RoD, we have designed and implemented an Software Defined Networking (SDN) framework with energy-awareness and traffic engineering, validating through experimentation the concept of RoD with an experimental prototype.

As overview of the contributions of this thesis, we can summarize them as follows:

- In Chapter 2, we have first analyzed existing LTE platforms for experimentation with SDR *i*) a full software implementation of the stack for general purpose CPUs and a Field Programmable Gate Array (FPGA) based prototype with hardware acceleration. We have chosen the two most known solutions, srsLTE and OAI frameworks, and we have characterized them in terms of software extensibility, CPU usage and network performance. Our findings have proved the “word of mouth” knowledge among practitioners, showing the software design of srsLTE makes easier to add functionalities on top of the framework;

in the other hand, OAI has shown a better and more robust performance in stability, CPU resource consumption, networking and, standard compliance. Additionally, we have also provided some tips to configure both platforms, especially for TX/RX gain settings. We have concluded the chapter, presenting an alternative solution based-on a FPGA prototype with energy-awareness design for Cloud RAN (C-RAN) deployments. We have described the flexibility and reconfigurability of the prototype, being capable of splitting the functionality of the Evolved Node B (eNB) stack to run across multiple locations (Cloud, MEC-like server and the Remote Radio Head (RRH)) and reconfigure such splits dynamically in run-time.

- In Chapter 3, we have presented a framework for energy characterization of devices in short time scale, providing a granularity of μs in our measurements. This platform has allowed us to validate an analytic model for energy consumption of RA algorithms in IEEE 802.11 given a certain value of the transmission power. Our results prove the current algorithms are not efficient when considering the energy consumption. Therefore, these algorithms can be extended to consider the energy efficiency in order to keep the network performance while reducing the spent energy in the wireless card, lengthen in that way the battery life in mobile devices. In the second part of this chapter, we have carried out an exhaustive characterization of the energy consumption for the different subsystems that compose the FPGA prototype introduced in Chapter 2. This analysis lies on the characterization of the CPU, Ethernet card and Radio Frequency Integrated Circuit (RFIC), providing a complete picture of how the energy is spent across the entire prototype. Our results show that depending on the selected configuration of the deployment has indeed an impact on the energy efficiency of the system.

- In Chapter 4, we have raised the time scale of our energy analysis, introducing the concept of the RoD schemes for mobile networks. We have proposed an analytic model for realistic environments that considers the start-up time of the Access Point (AP). This model has been presented in two flavors: an exact model with high computational complexity and simplified model that shorten the computational time approximately 50 times while providing accurate results. We have studied the performance evaluation of the proposed model when applied to a campus-like deployment. Our results show these schemes are needed for dense networks to adapt to varying load of the system while improving the energy efficiency. To conclude the chapter, we have also presented the optimal configurations for our system considering different service rates and bounding the service time.

- Finally, in Chapter 5, we have presented the OFTEN framework (Open Flow framework for Traffic Engineering in mobile Networks with energy awareness) an SDN platform that enables traffic engineering and RoD schemes in real deployments. The goal of this framework is to offer an API (Application Programming Interface) to facilitate the implementation of a SDN application to support centralized decision policies and the ability to

control switch on/off the elements that form the network. We have proposed a set of extensions for OpenFlow protocol that allows the controller to perform energy management of the switches. As part of this work, we have implemented the OFTEN framework employing Ryu controller in a real world testbed. Our experimental validation of the system prove *i)* the feasibility of the RoD schemes for real deployment and *ii)* the reduction of the energy consumption while providing QoS in this kind of scenarios.

As summary, this thesis has addressed several aspects of experimentation applied to research, providing analysis and methodologies to follow. We have first presented and characterized different solutions for experimentation with LTE systems employing SDR boards. Next, we have analyzed the energy efficiency at multiple time scales, first at μs scale, characterizing and dissecting the energy impact when considering different wireless communication parameters for IEEE 802.11 and LTE; and second, raising the time scale up to seconds, analyzing the RoD schemes applied to mobile networking deployments. To conclude this work, we have presented the OFTEN framework, an SDN solution that enables RoD schemes in real world scenarios. The presented work proves that the softwarized platforms can effectively enhance the energy efficiency at different scales whether we consider the device deployment or the entire network.

Appendices

Appendix A

FPGA energy aware design and system implementation

This appendix contains more details of the FPGA platform implementation presented in the Chapter 2. This is a fully distributed architecture designed to facilitate the flexible partition of the communication functions. On top of that, the implementation relies on standard networking technologies (e.g., it complies with the most relevant features described in the LTE standard and targets commercial off-the-shelf HW elements). In the following the design and implementation fundamentals of the dynamic hotspot prototype are discussed, setting the focus on the HeNB.

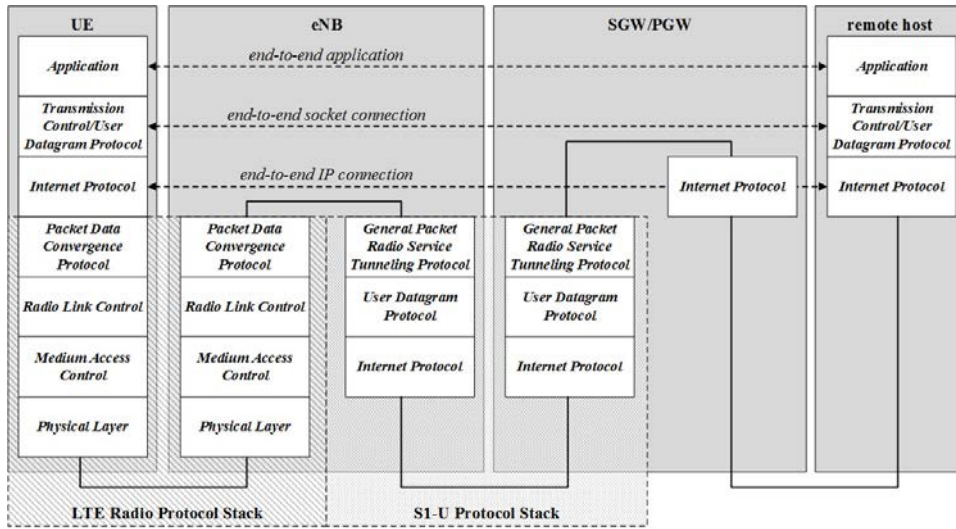


Figure A.1: LTE-EPC data plane protocol stack.

A.1. L2 and upper-layers

All L2 and above layer functionalities have been implemented in the SW domain by extending the existing features of the LTE-EPC network simulator (LENA) [92]. LENA is an open-

source implementation of the LTE and EPC standards, originally born to enable the simulation of realistic scenarios relying on the widespread network simulator ns-3 [93]. It possesses two main characteristics that result crucial to the development of the dynamic reconfigurable hotspot. First, given that ns-3 is a full stack simulator, all upper-layer functionalities are accurately implemented (which is not the case for link and system level simulators). Second, the core design of LENA is based on the Small Cell Forum MAC scheduler application programming interface (API). This enables its fast integration to realistic developments (i.e., by considering commercial product requirements), as well as modelling their constraints in the scheduler design. On top of that, the LENA module also implements the Evolved Packet Core (EPC) network elements and its protocol stacks, including the Serving Gateway (S-GW), the PDN Gateway (P-GW) and the Mobility Management Entity (MME). A full internet protocol stack is also incorporated. It provides cornerstone networking functions, including the Transmission Control Protocol (TCP), the User Datagram Protocol (UDP) and the Internet Protocol (IP). With all those combined features, LENA allows to accurately simulate the performance of end-to-end services in LTE-based network configurations, considering both the fronthaul and backhaul, as well as all communication stack protocols (Figure A.1).

The original purpose of LENA was to be used as an advanced SW simulator. Consequently, a single process included all network elements of a given simulation scenario. Similarly, all interactions between those network elements were constrained to the ns-3 simulation-space (i.e., no interaction to third party SW or external network elements was enabled; e.g., video applications). Moreover, neither real-time computing nor HW interaction support was meant to be originally provided. All these limitations have been addressed in the current work by adequately extending the original LENA code. To start with, the advanced capabilities offered by ns-3 with the objective to ease its integration into testbeds have been exploited to facilitate its time-constrained interaction with external SW and HW elements (e.g., FPGA-based PHY-layer). Moreover, the LTE and EPC processes have been splitted to enable the distributed execution of the underlying communication functions of the hotspot prototype. A new L2-L1 interface has also been implemented to facilitate the dynamic reconfiguration of the HWA L1. Finally, specific HW requirements have been integrated onto the scheduler.

Those features aimed at enabling the real-time interaction of LENA with external HW elements have been implemented based on the emulation functions provided by ns-3. Specifically, it has been used the *RealTime* scheduler to synchronize the events generated by the different network models (i.e., ns-3 is a discrete-event network simulator, where events are scheduled to be executed at specified simulation instants) with the wall clock of the local host (i.e., instead of the virtual simulated clock that would be normally used). As a result, all packet traffic is generated and processed in real-time (i.e., according to the timing requirements of the HWA L1 implementation, as well as those related to the currently adopted NETCFG). Moreover, the original LTE interfaces have been updated to work with real IP packets, that can be sent through the network and reinjected to the simulator. In more detail, the *File Descriptor NetDevice* functionality offered

by ns-3 has been used to read/write traffic from/to a physical device in the host machine, permitting likewise the implemented LTE protocol stack to send/receive packets/frames to/from external HW components, third party SW and/or other ns-3 processes by means of standard UDP-IP packets. This extension considers the L2-L1, S1-U and S1-MME interfaces. All these modifications allow to realistically emulate 5G scenarios considering different SDN/NFV configurations [94], and facilitate MEC-based deployments (e.g., in the line of *NETCFG2*).

Several modifications have also been introduced in LENA to enable distributing its underlying functions among different 5G nodes. To start with, the EPC protocols have been extended to facilitate the emulation of its functions on different network elements. Furthermore, all processes implementing the RAN have also been separated. Specifically, the functionalities corresponding to the (H)eNB and UE are now completely splitted from one another, as showed in Figure 2.1. As a result, all underlying SW communication functions can be executed in a completely flexible and distributed manner, covering a wide range of function-split cases (i.e., from being hosted in a single server, up to a fully cloud offload as in the case of *NETCFG3*).

The specifications defined by the Small Cell Forum for the scheduler API have been exploited in order to facilitate a virtualized small cell architecture [60]. Toward that end, the RAN functions of LENA are now fully virtualizable, enhancing accordingly the splitting capabilities of the whole LTE protocol stack. On top of that, a custom header has been added encapsulating the standard UDP-IP packets to facilitate the interchange of protocol-related information among distributed nodes. On this matter, following the principles of the MAC-split defined in the virtual small cells paradigm, the API has been conveniently extended to generate time-stamped header frames. These frames contain the relevant primitives that need to be exchanged with the HWA L1 (i.e., as a serialized bit sequence). As for the EPC they also enabling a number of primitives defined by the LTE standard, including the procedure to attach new users, create new connections and transmit data and control packets over them. This design relaxes the requirements in terms of both latency and BW in C-RAN and PHY splits as defined in [60], such as the one used by *NETCFG1*. Similarly, the MAC-split helps attaining an increased flexibility when moving the L2 and EPC network elements.

Concerning the scheduler functionalities, the baseline round robin implementation has been updated to include specific constraints originated at the HWA implementation of the downlink (DL) L1. In more detail, natively LENA does not consider any limitation regarding the number of UEs that can be simultaneously allocated in the DL at any given moment, whereas in reality the physical resources that can be dedicated to convey the DL control information (DCI) is limited and depends on the adopted WCPs (i.e., each DL BW configuration uses a different number of REs). In the same way, the size of the allocated TBs takes into account the specifications of the channel encoder implementation provided by the FPGA-based PHY-layer.

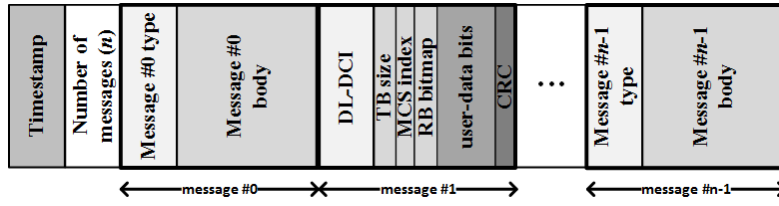


Figure A.2: Simplified view of the L2-L1 interfacing frame (with an example DL-DCI message).

A.2. L2-L1 interface

The L2-L1 interface constitutes a key element to facilitate the dynamic function split by enabling the agile communication of the HWA and SW blocks. In that regard, the main objective of this SW piece is to provide a reliable communication means with strict latency requirements. Moreover, its inner procedures are transparent to the HWA and SW blocks, simplifying likewise their design. This means that the communication of the partitioned functions is abstracted from the DSP design and only presents a minimal impact to the internal structure of each block (i.e., by conforming with essential interfacing specifications). This allows optimizing each stage independently and promotes the modularity of the overall system design, which is essential to the distributed nature of the dynamically reconfigurable hotspot that we are presenting.

The real-time L2-L1 interfacing SW is executed in the PS of the target SoC, which also hosts the PHY-layer implementation. On top of that, a customized distribution of the Linux operating system has been utilized, which emphasizes real-time tasks. More particularly, a fully preemptive kernel is used to satisfy the stringent latency requirements of real-time applications (i.e., real-time priority can be assigned to specific tasks, ensuring a predictable and minimized scheduling delay). By this way, application-level and kernel-level jitters are reduced enabling likewise a reliable communication between the partitioned L2 process and the HWA L1 with a deterministic behaviour. A series of SW techniques were employed toward that end, which among others include real-time scheduling policies, assignation of real-time priority to critical tasks, memory locking mechanisms and pre-faulting stacks [95].

From an architectural point of view, the L2-L1 interface is connected to the 5G node hosting the L2 process of LENA on the one end, and to the HWA implementation of the L1 residing in the same chip on the other end.

The interaction with L2 and above layers is based on the exchange of messages on a sub-frame basis. Given the strict latency requirements of the real-time L2-L1 communication, the transfer of information between the SW processes relies on a UDP connection. In more detail, the messages are assembled onto L2-L1 interfacing frames, which use a custom and flexible format that efficiently adapts its contents according to the current system configuration (e.g., the use of different WCPs greatly affects the amount of information to be exchanged between L2 and L1). As it can be observed in Figure A.2, the interfacing frames are comprising a number of control and data messages that need to be passed from LENA to the HWA L1. The DCI is of

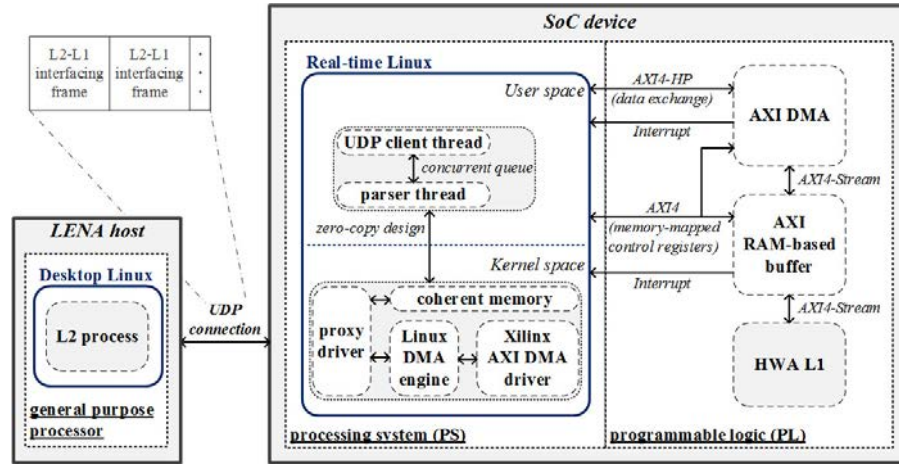


Figure A.3: General overview of the implemented L2-L1 interface.

key importance, since it defines the contents each radio frame (i.e., a DCI is generated for each subframe, according to the Small Cell Forum API). For each transport block (TB) directed to a given UE, the following will be specified: its size (i.e., amount of user-data bits), MCS index and the bitmap of the allocated RBs (i.e., bit-mask indicating which set of physical DL resource elements (REs) are dedicated to transmit the TB). Other control plane messages that are supported include the system information block (SIB) or the master information block (MIB). Evidently, the implemented interface is also enabling the time-constrained exchange of the data plane. Given that the hybrid automatic repeat request (HARQ) mechanism has not been implemented, cyclic redundancy check (CRC) codes are used as a basic error detection mechanism in order to avoid passing corrupted packets to the upper layers. The first task of the L2-L1 interfacing SW is thus to parse the received L2-L1 interfacing frames in order to recover the control information (i.e., MCS index, TB size and RB allocation bitmap) and user data (i.e., TB) required by L1. As a result of the parsing a number of 32-bit words are generated and are forwarded to the PL.

As for the interaction with the HWA L1, the PS and PL communicate through an embedded dedicated high-speed interface which is based on a proprietary bus specification known as advanced extensible interface (AXI). In more detail, the communication of the L2-L1 interface and the HWA L1 is controlled by an AXI-based direct memory access (DMA) core. The DMA block is connected to a high-performance and high-bandwidth port, AXI4-HP, on the PS side. On the PL end, the DMA is connected to an AXI4-Stream interfacing block, which provides a simplified low-latency master-slave communication mechanism. By using this specific interfacing architecture, the PS-PL communication supports an exchange of data up to 6400 Mbits/s¹.

Figure A.3 presents an overview of the custom SW architecture implementing (in C and C++) the L2-L1 interface. As it can be observed, the application is divided in two main components: one is executed in the user-space, while the other runs in kernel-space.

¹This data-rate calculation accounts for the 32-bit AXI-4 bus and the specific operating frequencies of the embedded PS (i.e., 666.7 MHz) and AXI logic within the PL (i.e., 200 MHz).

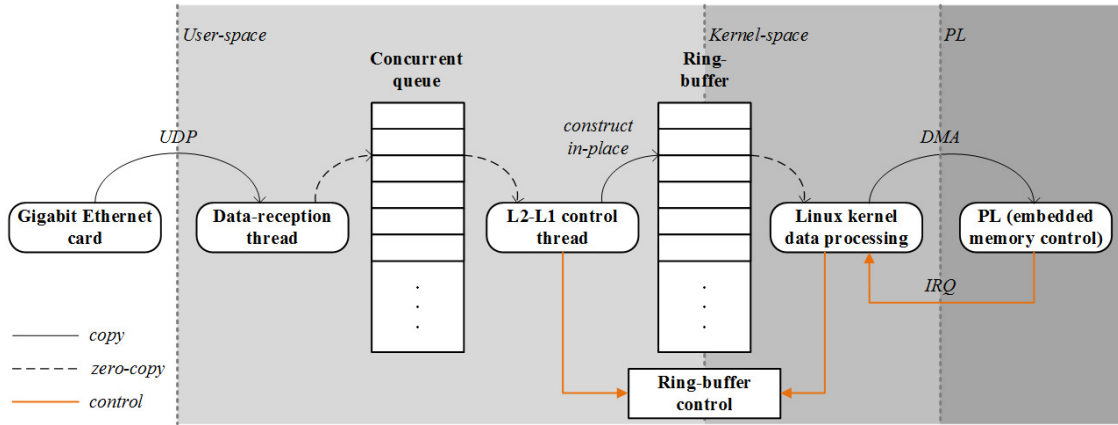


Figure A.4: PS to PL communication and data processing flow.

The user-space process comprises two threads that share a concurrent queue. The first thread is responsible for establishing the network connection to the L2 host and receiving the UDP packets generated in the communication. The second thread implements the parsing functions, in order to interpret the received information and produce the 32-bit words that will be forwarded to the HWA L1. Consequently, this second thread also manages the required intermediate buffering. An effective implementation of the concurrent queue, combined with the use of modern C++ programming features (e.g., move semantics), enables to minimize the time required by the PS to complete this task and guarantees that no incoming data will be queued for prolonged periods, even if the current system configuration results in an elevated data-rate.

A custom driver executed in the kernel-space efficiently manages the interaction with the different HW elements. Namely, it is in charge of parsing the device-tree information and initializing those HWA blocks that play a relevant role in the dynamic reconfiguration of the system by programming a set of memory-mapped registers (i.e., DMA core and custom L1 implementation). Moreover, the driver is also responsible for setting up the related interrupt service routines and allocating the necessary DMA-able memory resources. Regarding the interrupts, each 1 ms (i.e., every new subframe) the HWA L1 will generate a request to receive a new L2-L1 interfacing frame. Toward that end, a fraction of the L2-L1 interface functionalities is implemented in the PL side. More concretely, a random access memory (RAM)-based buffer that acts as a jitter-absorbing first input first output (FIFO) memory has been implemented (i.e., a small number of L2-L1 interfacing frames will be initially stored internally at the FPGA, accounting for the network delays originated from the distribution of functions among different 5G nodes). The memory controller associated to this embedded buffer implements a simple control mechanism which is in charge of generating interrupts to the PS when necessary (i.e., a counter and a register controlling the number of stored frames) and forwarding the stored L2-L1 interfacing frames to the HWA PHY-layer (i.e., basic control of the underlying memory read and write operations). Similarly, the kernel driver uses a ring-buffer memory element mapped onto the user-space, where the associated thread pushes the 32-bit words generated by the parser. Upon the reception of a

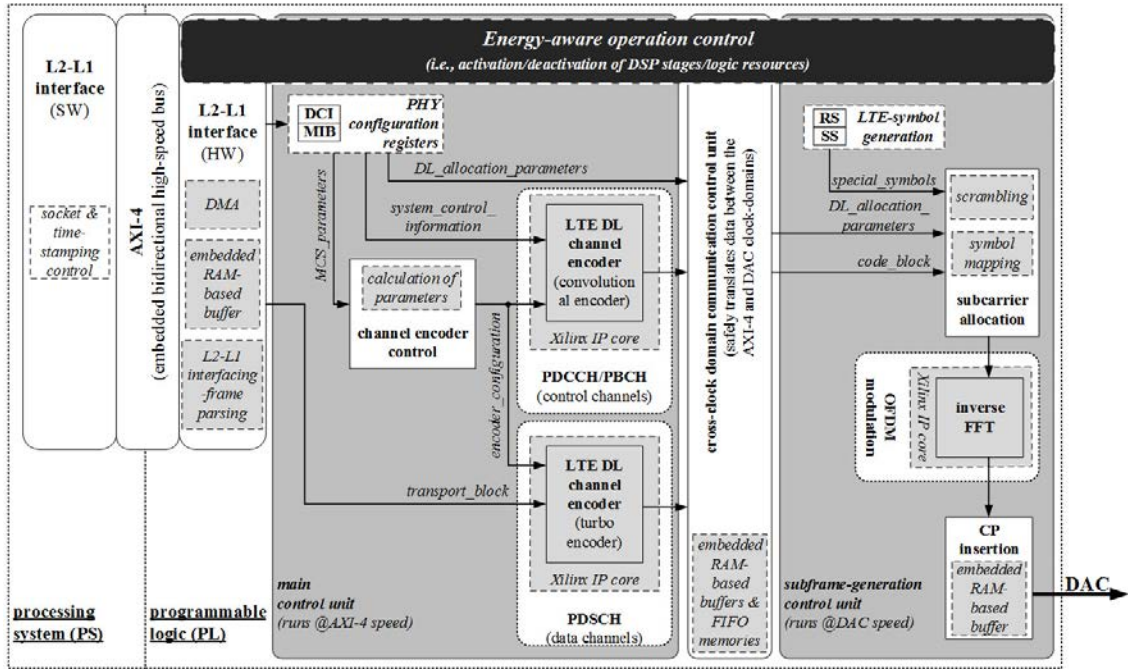


Figure A.5: RTL architecture of the HWA L1.

new interrupt from the PL, the kernel driver recovers a L2-L1 interfacing frame from the buffer and pass it to the FPGA. A zero-copy design has been implemented to avoid degrading the performance of the PS during the interactions between the user-space and kernel-space code. Like this, unnecessary data copying and context switching operations are avoided. Additionally the interrupt service routine is also used to configure the DMA and start a new transaction when required, by transferring the data that was last fetched from the ring-buffer descriptor to the PL buffer. Finally, when the transaction is completed the driver updates the ring-buffer pointers accordingly. A detailed diagram of the communication and data processing flow taking place inside the L2-L1 interface is provided in Figure A.4.

A.2.1. Energy-aware HWA L1

All required DL L1 features have been implemented as real-time FPGA-based HWA functions by using advanced digital design techniques. A low-level optimized register transfer level (RTL) architecture was designed, focusing on two major goals: i) to minimize the utilized logic resources and its related energy-consumption, and ii) to enable a flexible reconfiguration of its operation at run-time. This was achieved by combining a highly resource-efficient RTL design, which reused Xilinx intellectual property cores (IP-cores; i.e., predefined synthesizable blocks) to implement the most complex DSP functions (e.g., inverse FFT, channel coding), and dedicated control units to ensure the energy-efficient yet very flexible operation of the logic. In this respect, the operation of the HWA L1 can be adapted according to the requirements provided by the SW L2 or due to the adoption of a new NETCFG (including DL BW adaptations), in a subframe basis. A

general overview of the energy-aware RTL architecture implementing the HWA L1 of the HeNB is depicted in Figure A.5.

From a functional point of view, the L2 dictates the configuration to be utilized by the PHY-layer according to the instantaneous operative requirements of the HeNB. Among others, this includes the specific allocation of frequency resources to each UE attached to the HeNB (i.e., RBGs) and its related QoS constraints (i.e., channel coding parameters). All the necessary control information to reconfigure the HWA L1 is provided from the SW domain (PS) through L2-L1 interfacing-frames (recall Figure A.2). The main control unit of the PHY-layer parses this information in order to determine whether there is data allocated onto the available RBGs or not, and generates the corresponding user-data (i.e., DL shared channel, DL SCH; management of the turbo encoding stage) and control channels contents (i.e., physical DL control channel, PDCCH/physical broadcast channel, PBCH; management of the convolutional encoding stage). Additionally, this complex state machine also manages the errors in the L2-L1 communication: in case of missing or wrongly decoded control information, the HeNB will interrupt its transmission until valid data is received from L2 to generate the following frame (i.e., starting from subframe 0). This situation will be signalled to the embedded memory buffer within the HWA part of the L2-L1 interface which, in turn, will generate an interrupt to alert the PS. This ensures that the exception will be correctly handled at all communication levels. A second control unit is in charge of allocating the required contents to each RE in the DL signal (i.e., generation of the frequency-subframe), driving the operation of the inverse FFT and CP insertion stages, and, finally, providing further management of L2-L1 communication errors. In more detail, in case the required DL SCH control is not available when needed, a request will be made to the principal state machine to interrupt the HeNB transmission.

In order to deal with the bit-intensive computation resulting from the adaptive real-time operation of the HWA L1, the RTL architecture is articulated around two differentiated clock domains (see Figure 8). Apparently, the first one is directly related to the sampling frequency of the digital-to-analog conversion (DAC) circuitry (i.e., derived from the principal LTE sampling frequency, $\frac{1}{N} \cdot 30.72$ MHz, with $N = [1..16]$). Regarding the second clock domain, it needs to be aligned with the PS from which data is received through the dedicated high-speed AXI-4 streaming interface. Considering the specifications of the channel coding stage, the most convenient is to use a clock with a frequency, M , at least six times the one derived from the DAC (i.e., $M \geq 6 \cdot \frac{1}{N} \cdot 30.72$ MHz). That is accounting for the worst possible case, when the MCS index ≥ 17 , which requires to work with groups of six bits (i.e., 64-QAM) for each allocated RE. In that case, since we do consider a 20 MHz DL BW, six bits will then need to be sequentially entered to the channel encoder IP-core during each 30.72 MHz clock cycle (i.e., $N = 1$ and $M \geq 184.32$ MHz). Using a fixed 200 MHz² clock, thus, allows processing one RE per slow-clock cycle independently of the current system configuration (i.e., DL BW and MCS). Furthermore, it accelerates the computation

²This specific frequency value has been selected by considering the specifications of the PS clock subsystem of the target SoC device.

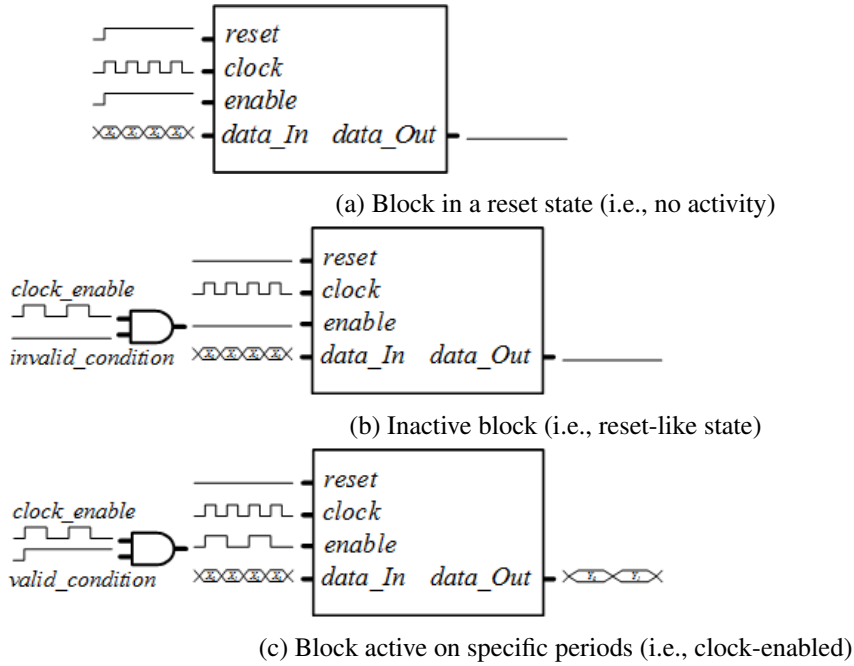


Figure A.6: Control the activity of the DSP stages through clock-gating techniques in order to save energy.

of the channel encoding operations resolving likewise all the associated latency issues.

Besides reconfiguring and driving the operation of the different stages comprising the PHY-layer, the two control units are also in charge of controlling all synchronous communications among them. This demands to efficiently exploit the limited time budget while accounting for the characteristics of cross-clock domain data sharing, for which complex memory structures have been built. More importantly, the flexible control of the logic provided by the control units forms the basis of the energy-aware design, when intelligently combined with traditional clock-gating techniques. That is, each (synchronous) component/process of the digital circuit will be active on the rising edge of the clock directing its operation, only when the related clock-enable signal is high. On top of that, the clock-enable signal is combined with other control signals which account for the current configuration of the HeNB (e.g., selected DL BW, allocated RBGs or MCS values), in order to minimize the amount of logic which is actively utilized at each moment. This design-time decision contributes to the overall reduction of the energy budget of the HeNB: minimizing the switching activity of the implemented gates (e.g., flip-flops), actively reduces the energy consumed by the digital circuit (experimental studies show dynamic power savings up to 30% for the complete design, and up to 90% at a block/IP-core level [96, 97]). This is especially relevant for those logic elements residing in the faster clock domain, considering that the dynamic power consumption of any FPGA design has a linear dependency on the clock frequency [98]. Our energy-aware RTL design exploits this trait by minimizing the amount of active logic at any instant, keeping the rest in a reset-like state, as shown in Figure A.6. At a system level,

all those DSP components related to the turbo encoding stage will remain completely inactive during the generation of those subframes where no user data is being transmitted. That is, only those stages related to the parsing and encoding of the PDCCH or PBCH, as well as the ones generating the synchronization or reference signals, will present activity during the generation of each subframe comprising the transmitted signal (i.e., there are certain REs that will be always occupied even when no DL user data is allocated at all). Similarly, the energy-saving efforts are propagated across the entire RTL hierarchy, through low-level optimizations within each designed DSP block. As an illustrative example, whereas the internal storage is dimensioned to support the highest user data capacity requirements defined by the LTE standard (i.e., all RBGs allocated to a single user, maximum MCS value, 20 MHz DL BW), only the minimum required number of registers/memory elements are actively used; the others will remain in an idle/reset state. A similar approach has been used for the FIFOs, and related control-logic, that are utilized to pass the encoded user data, generated in the AXI-4 clock domain, to the symbol mapping stage residing in the DAC clock domain. In that case, the logic with the faster operating frequency is only active each clock cycles, where M depends on the MCS value (being $M = 6$ the worst case, for $MCS \geq 17$).

An added yet substantial benefit of using clock-gating techniques, in energetic terms, is that the use of the frequency synthesizer primitives found in the clock management (CM) tiles of the FPGA devices is avoided. CM conveniently allows deriving the required clock signals from those received by the FPGA (i.e., from an external source), providing highly precise phase and jitter specifications, but also incurring in a nonnegligible increase in the consumed energy. According to estimations that we obtained by using the *Xilinx power analyzer* (XPA) SW tool, the increase in the consumed energy derived from the use of CMs can be as high as 33.44%. Specifically, we compared a simplified version of the clock-enabled HeNB design presented above (i.e., not interfaced to L2, but using a fixed DL allocation) with a second implementation that was modified to exploit the features provided by the CM tiles (i.e., different clock signals were generated internally instead of using clock-enable signals). Precise energy consumption estimates for each implementation were finally obtained by loading the placed and routed (PAR) designs to the XPA, jointly with its related post-PAR signal-toggle activity files (i.e., realistic modelling of the timing delays of the implemented circuit).

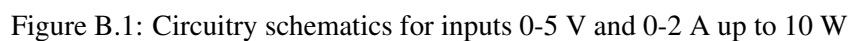
Appendix B

Measurement Circuitry Schematics

The university's Technical Office in Electronics designed and implemented two different prototypes with the requirements of DAQ card in mind, to be attached as *i*) input for measurements, *ii*) as output to power the device as explained in Chapter 3.

- Fig B.1 shows the schematics for 0-5V and 0-2A
- Fig B.2 shows the schematics for 12.5-20V and 0-5A

Following the schematics, **CLEMA6** is attached as input to the DAQ to acquire the measurements in terms of current (*MEDIDA I +/-*) and voltage (*MEDIDA V +/-*), **CLEMA4** is designated to be attached to the device to be characterized and finally, **CLEMA8** is attached to a power source to feed the circuitry itself and the device.



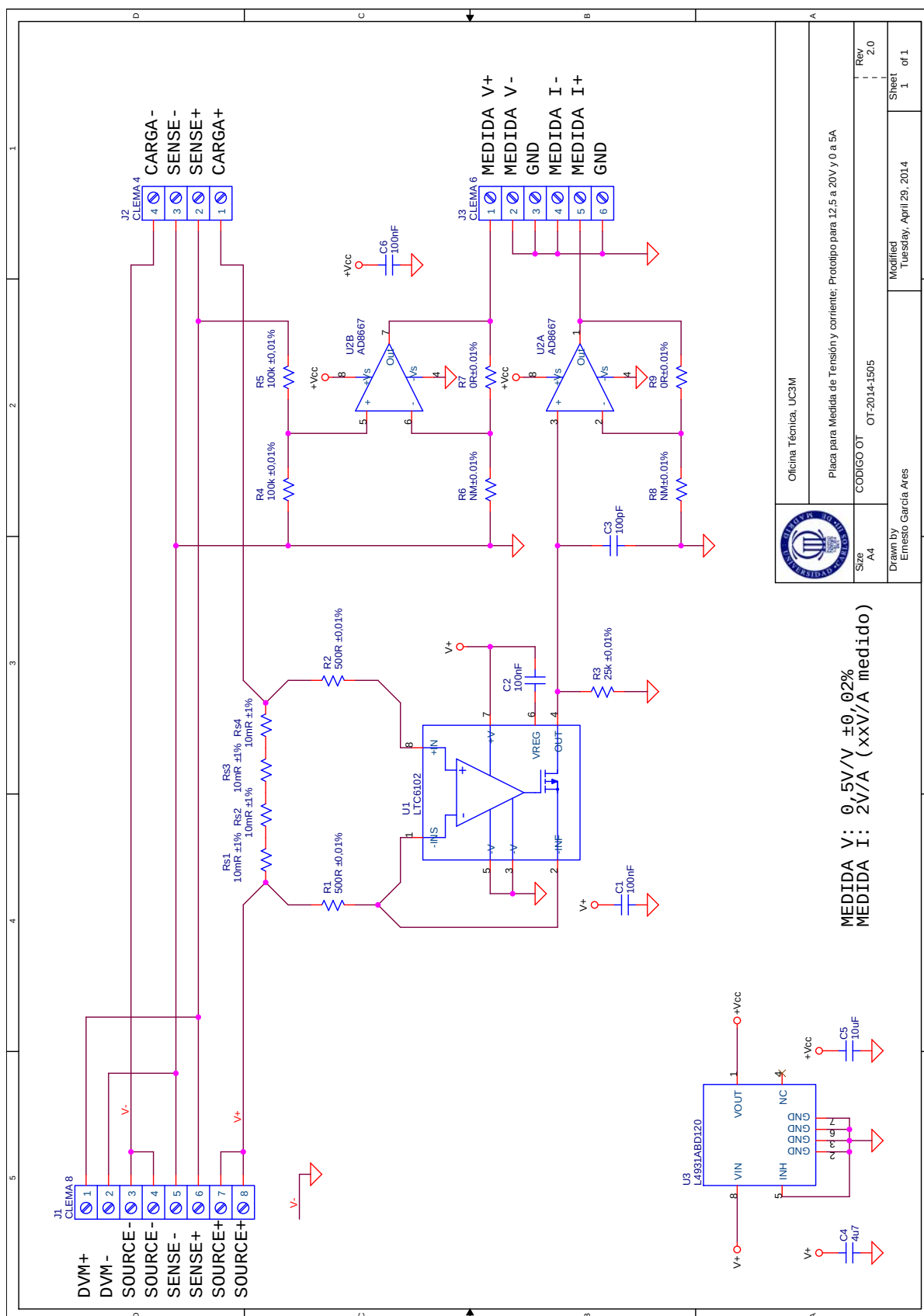


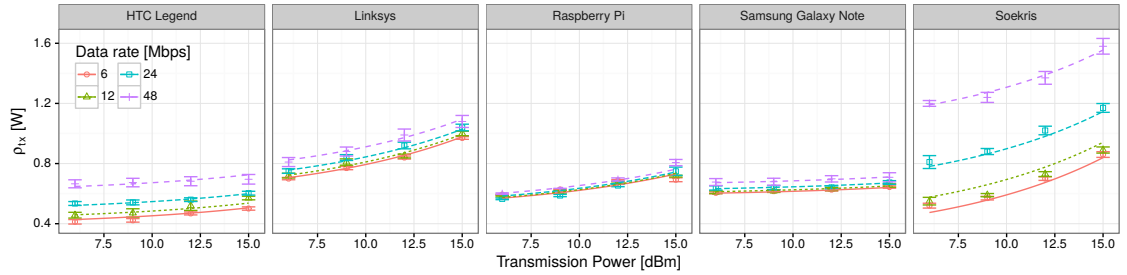
Figure B.2: Circuitry schematics for inputs 12.5-20 V and 0-5 A up to 100 W

Appendix C

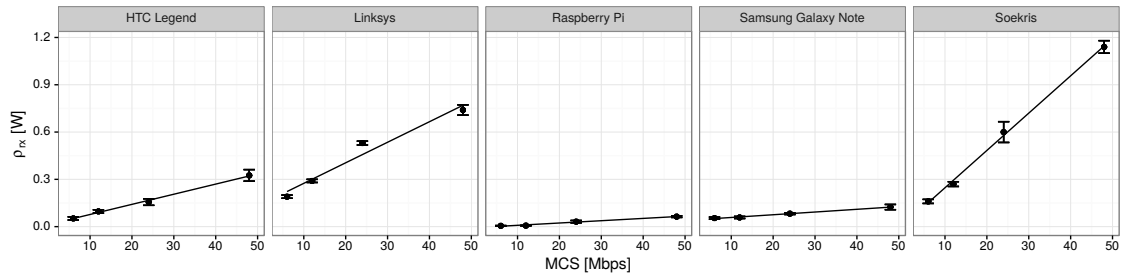
Extension of Energy Model for Rate Adaptation

Fitting the model

The models presented in Section 3.2.2.2 are fed with the data reported in [67], and the resulting fitting is illustrated in Figs. C.1a and C.1b, while Table C.1 collects the model estimates for each device (with errors between parentheses), as well as the adjusted r-squared. Since these linear models show a very good fit, they support the generation of synthetic data for the different MCS and TXP required.



(a) ρ_{tx} fit as a function of MCS and TXP.

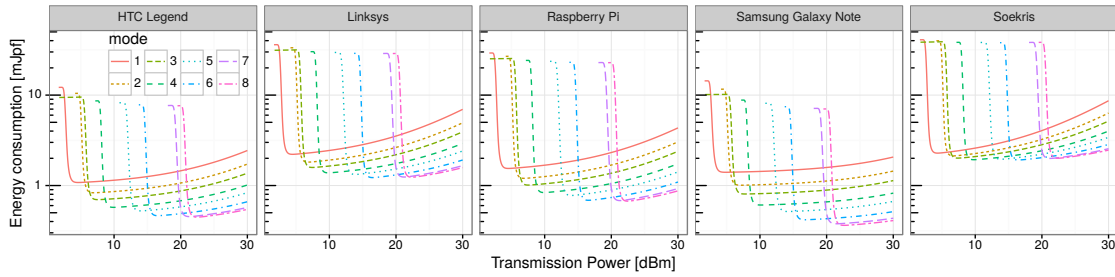


(b) ρ_{tx} fit as a function of MCS.

Figure C.1: Linear regressions.

Table C.1: Linear Regressions

Device	ρ_{tx} model estimates (α_i)				ρ_{rx} model estimates (β_i)		
	(Intercept) [W]	MCS [Mbps]	TXP [mW]	adj. r^2	(Intercept) [W]	MCS [Mbps]	adj. r^2
HTC Legend	0.354(14)	0.0052(3)	0.021(3)	0.97	0.013(3)	0.00643(11)	>0.99
Linksys WRT54G	0.540(12)	0.0028(2)	0.075(3)	0.98	0.14(2)	0.0130(7)	0.96
Raspberry Pi	0.478(19)	0.0008(4)	0.044(5)	0.88	-0.0062(14)	0.00146(5)	0.98
Galaxy Note 10.1	0.572(4)	0.0017(1)	0.0105(9)	0.98	0.0409(10)	0.00173(4)	0.99
Soekris net4826-48	0.17(3)	0.0170(6)	0.101(7)	0.99	0.010(8)	0.0237(3)	>0.99

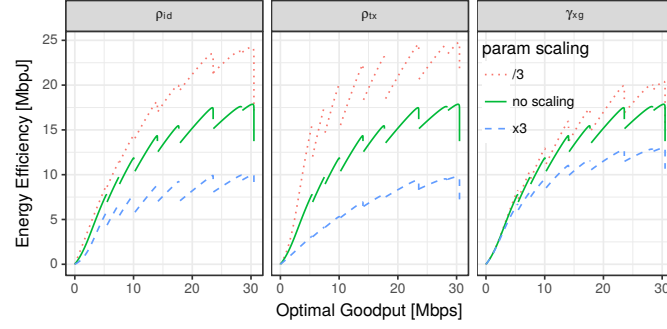
Figure C.2: Expected energy consumption per frame in *millijoules per frame* (mJpf) under fixed channel conditions.

Sensitivity to Energy Parameter Scaling

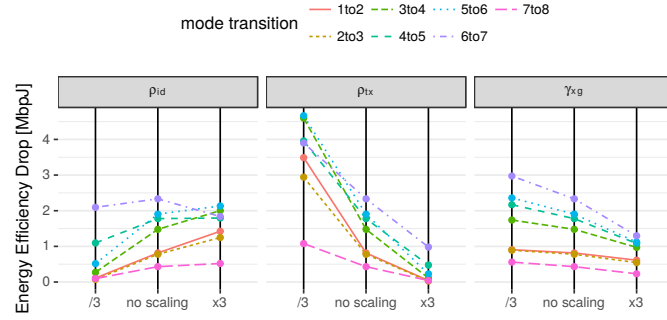
We explore how the different energy parameters affect the energy efficiency vs. optimal goodput relationship. For this purpose, we selected the Raspberry Pi curve from Fig. 3.3 (results are analogous with the other devices) and we scale up and down, and one at a time, the four energy parameters ρ_{id} , ρ_{tx} , ρ_{rx} , and γ_{xg} . The scaling up and down is done by multiplying and dividing by 3, respectively, the numerical value of the considered parameter. One of the first results is that the impact of ρ_{rx} is negligible, a result somehow expected as the cost of receiving the ACK is practically zero. From this point on, we do not consider further this parameter.

We show in Fig. C.3a the overall effect of this parameter scaling. The solid line represents the base case with no scaling (same curve as in Fig. 3.3), and in dashed and dotted lines the corresponding parameter was multiplied or divided by a factor of 3, respectively. As expected, larger parameters contribute to lower the overall energy efficiency. However, the impact on the energy efficiency drops between mode transitions is far from being obvious, as in some cases transitions are more subtle while in others they become more abrupt.

To delve into these transitions, we illustrate in Fig. C.3b the “drop” in energy efficiency when changing between modes. As it can be seen, the cross-factor is the less sensitive parameter of the three, because its overall effect is limited and, more importantly, it scales all the leaps between mode transitions homogeneously. This means that a higher or lower cross-factor, which resides almost entirely in the device and not in the wireless card, does not alter the energy efficiency vs. optimal goodput relationship (note that this parameter results in a constant term in (3.7)). Thus,



(a) Overall effect.



(b) Impact on mode transitions.

Figure C.3: Impact of energy parameter scaling on the energy efficiency.

the cross-factor is not relevant from the RA point of view, and energy-aware RA algorithms can be implemented by leveraging energy parameters local to the wireless card.

On the other hand, ρ_{id} and ρ_{tx} have a larger overall effect, plus an inhomogeneous and, in general, opposite impact on mode transitions. While a larger ρ_{id} contributes to larger leaps, for the case of ρ_{tx} , the larger energy efficiency drops occur with smaller values of that parameter. Still, the reason behind this behavior is the same for both cases: the wireless card spends more time in idle (and less time transmitting) when a transition to the next mode occurs, which has a higher data rate.

This effect is also evident if we compare the Samsung Galaxy Note and the HTC Legend curves in Fig. 3.3. Both devices have ρ_{id} and ρ_{tx} in the same order of magnitude, but the HTC Legend has a larger ρ_{id} and a smaller ρ_{tx} . The combined outcome is a more dramatic sub-linear behaviour and an increased energy efficiency drop between mode transitions.

References

- [1] “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021,” Tech. Rep., March 2017.
- [2] 5G-PPP, “5g ppp progress monitoring report,” 5G-PPP, Tech. Rep., 2017. [Online]. Available: <https://5g-ppp.eu/wp-content/uploads/2018/10/5G-PPP-Progress-Monitoring-Report-2017.pdf>
- [3] H. Kim and N. Feamster, “Improving network management with software defined networking,” *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, 2013.
- [4] M. Condoluci and T. Mahmoodi, “Softwarization and virtualization in 5g mobile networks: Benefits, trends and challenges,” *Computer Networks*, vol. 146, pp. 65–84, 2018.
- [5] 3GPP, “TS 23.501 – System Architecture for the 5G System; Stage 2,” Dec 2017.
- [6] ITU, “Press Release: ITU agrees on key 5G performance requirements for IMT-2020,” Feb 2017.
- [7] P. Serrano, P. Salvador, V. Mancuso, and Y. Grunenberger, “Experimenting with commodity 802.11 hardware: Overview and future directions,” *IEEE Comms. Surveys Tutorials*, vol. 17, no. 2, pp. 671–699, 2015.
- [8] M. Duarte, A. Sabharwal, V. Aggarwal, R. Jana, K. K. Ramakrishnan, C. W. Rice, and N. K. Shankaranarayanan, “Design and characterization of a full-duplex multiantenna system for wifi networks,” *IEEE Trans. Vehicular Tech.*, vol. 63, no. 3, pp. 1160–1177, Mar 2014.
- [9] P. D. Sutton, K. E. Nolan, and L. E. Doyle, “Cyclostationary signatures in practical cognitive radio applications,” *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 1, pp. 13–24, Jan 2008.
- [10] P. Bahl, R. Chandra, T. Moscibroda, R. Murty, and M. Welsh, “White Space Networking with Wi-fi Like Connectivity,” in *Pro. ACM SIGCOMM*, 2009, pp. 27–38.
- [11] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, “OpenAirInterface: A Flexible Platform for 5G Research,” *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 33–38, Oct 2014.

- [12] I. Gomez-Miguel, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, and D. J. Leith, “srsLTE: An open-source platform for LTE evolution and experimentation,” in *Proc. ACM WiNTECH*, 2016.
- [13] C. Capretti, F. Gringoli, N. Facchi, and P. Patras, “LTE/Wi-Fi Co-existence under Scrutiny: An Empirical Study,” in *Proc. ACM WiNTECH*, Oct 2016.
- [14] P. Rost, C. J. Bernardos, A. D. Domenico, M. D. Girolamo, M. Lalam, A. Maeder, D. Sabella, and D. Wabben, “Cloud technologies for flexible 5G radio access networks,” *IEEE Communications Magazine*, vol. 52, no. 5, pp. 68–76, May 2014.
- [15] G. Mountaser, M. L. Rosas, T. Mahmoodi, and M. Dohler, “On the Feasibility of MAC and PHY Split in Cloud RAN,” in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, March 2017, pp. 1–6.
- [16] S. S. Kumar, R. Knopp, N. Nikaein, D. Mishra, B. R. Tamma, A. A. Franklin, K. Kuchi, and R. Gupta, “FLEXCRAN: Cloud radio access network prototype using OpenAirInterface,” in *2017 9th International Conference on Communication Systems and Networks (COMSNETS)*, Jan 2017, pp. 421–422.
- [17] J. Paulo, I. Freire, I. Sousa, C. Lu, M. Berg, I. Almeida, and A. Klautau, “FPGA-based testbed for synchronization on Ethernet fronthaul with phase noise measurements,” in *2016 1st International Symposium on Instrumentation Systems, Circuits and Transducers (IN-SCIT)*, Aug 2016, pp. 132–136.
- [18] K. Sundaresan, M. Y. Arslan, S. Singh, S. Rangarajan, and S. V. Krishnamurthy, “FluidNet: A Flexible Cloud-Based Radio Access Network for Small Cells,” *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 915–928, April 2016.
- [19] N. Saxena, A. Roy, and H. Kim, “Traffic-Aware Cloud RAN: A Key for Green 5G Networks,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 1010–1021, April 2016.
- [20] C. Bluemm, Y. Zhang, P. Alvarez, M. Ruffini, and L. A. DaSilva, “Dynamic energy savings in Cloud-RAN: An experimental assessment and implementation,” in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2017, pp. 791–796.
- [21] A. Sadek, H. Mostafa, A. Nassar, and Y. Ismail, “Towards the implementation of multi-band multi-standard software-defined radio using dynamic partial reconfiguration,” *International Journal of Communication Systems*, vol. 2017, pp. 1099–1131, June 2017.
- [22] M. Braun, J. Pendlum, and M. Ettus, “RFNoC: RF Network-on-Chip,” in *GNU Radio Conference 1 (1)*, 2016. [Online]. Available: <https://pubs.gnuradio.org/index.php/grcon/article/view/3>

- [23] A. F. Beldachi and J. L. Nunez-Yanez, "Accurate power control and monitoring in ZYNQ boards," in *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, Sept 2014, pp. 1–4.
- [24] M. Shafique, L. Bauer, and J. Henkel, "Adaptive Energy Management for Dynamically Reconfigurable Processors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 1, pp. 50–63, Jan 2014.
- [25] C. Ravishankar, J. H. Anderson, and A. Kennings, "FPGA Power Reduction by Guarded Evaluation Considering Logic Architecture," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 9, pp. 1305–1318, Sept 2012.
- [26] R. Marlow, C. Dobson, and P. Athanas, "An enhanced and embedded GNU radio flow," in *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, Sept 2014, pp. 1–4.
- [27] J. van de Belt, P. D. Sutton, and L. E. Doyle, "Accelerating software radio: Iris on the Zynq SoC," in *2013 IFIP/IEEE 21st International Conference on Very Large Scale Integration (VLSI-SoC)*, Oct 2013, pp. 294–295.
- [28] H. Zeng, X. Liu, S. Megeed, N. Chand, and F. Effenberger, "Real-Time Demonstration of CPRI-Compatible Efficient Mobile Fronthaul Using FPGA," *Journal of Lightwave Technology*, vol. 35, no. 6, pp. 1241–1247, March 2017.
- [29] D. Riscado, J. Santos, D. Dinis, G. Anjos, D. Belo, N. B. Carvalho, and A. S. R. Oliveira, "A Flexible Research Testbed for C-RAN," in *2015 Euromicro Conference on Digital System Design*, Aug 2015, pp. 131–138.
- [30] B. Guan, X. Huang, G. Wu, C. Chan, M. Udayan, and C. Neelam, "A Pooling Prototype for the LTE MAC Layer Based on a GPP Platform," in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–7.
- [31] W. Sun, O. Lee, Y. Shin, S. Kim, C. Yang, H. Kim, and S. Choi, "Wi-fi could be much more," *IEEE Communications Magazine*, vol. 52, no. 11, pp. 22–28, November 2014.
- [32] R. L. G. Cavalcante, S. Stanczak, M. Schubert, A. Eisenblaetter, and U. Tuerke, "Toward Energy-Efficient 5G Wireless Communications Technologies: Tools for decoupling the scaling of networks from the growth of operating power," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 24–34, Nov 2014.
- [33] O. Bulakci, Z. Ren, C. Zhou, J. Eichinger, P. Fertl, D. Gozalvez-Serrano, and S. Stanczak, "Towards flexible network deployment in 5G: Nomadic node enhancement to heterogeneous networks," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, June 2015, pp. 2572–2577.

- [34] E. Ternon, P. Agyapong, L. Hu, and A. Dekorsy, "Energy savings in heterogeneous networks with clustered small cell deployments," in *2014 11th International Symposium on Wireless Communication Systems (ISWCS)*, August 2014, pp. 126–130.
- [35] E. Lähtekangas, K. Pajukoski, J. Vihriälä, G. Berardinelli, M. Lauridsen, E. Tirola, and P. Mogensen, "Achieving low latency and energy consumption by 5G TDD mode optimization," in *2014 IEEE International Conference on Communications Workshops (ICC)*, June 2014, pp. 1–6.
- [36] A. Fehske, G. Fettweis, J. Malmudin, and G. Biczok, "The global footprint of mobile communications: The ecological and economic perspective," *IEEE Commun. Mag.*, vol. 49, no. 8, pp. 55–62, August 2011.
- [37] Y. Chen, S. Zhang, S. Xu, and G. Li, "Fundamental trade-offs on green wireless networks," *IEEE Commun. Mag.*, vol. 49, no. 6, pp. 30–37, June 2011.
- [38] C. Han *et al.*, "Green radio: Radio techniques to enable energy-efficient wireless networks," *IEEE Commun. Mag.*, vol. 49, no. 6, pp. 46–54, June 2011.
- [39] G. Lim, C. Xiong, L. Cimini, and G. Li, "Energy-efficient resource allocation for OFDMA-based multi-RAT networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 5, pp. 2696–2705, May 2014.
- [40] L. Budzisz *et al.*, "Dynamic resource provisioning for energy efficiency in wireless access networks: A survey and an outlook," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 2259–2285, 4th Quart. 2014.
- [41] P. Serrano, A. de la Oliva, P. Patras, V. Mancuso, and A. Banchs, "Greening wireless communications: Status and future directions," *Computer Communications*, vol. 35, no. 14, pp. 1651–1661, 2012.
- [42] Y. S. Soh, T. Quek, M. Kountouris, and H. Shin, "Energy efficient heterogeneous cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 5, pp. 840–850, May 2013.
- [43] J. Wu, Y. Zhang, M. Zukerman, and E.-N. Yung, "Energy-efficient base-stations sleep-mode techniques in green cellular networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 803–826, 2nd Quart. 2015.
- [44] W. Sun, H. Li, and J. Wu, "Study on real energy consumption of large-scale campus wireless network," in *Proc. 2013 International Conference on Computing, Networking and Communications (ICNC)*, Jan 2013, pp. 605–609.
- [45] F. Ganji, L. Budzisz, and A. Wolisz, "2013 IEEE 24th annual international symposium on personal, indoor, and mobile radio communications (pimrc)," Sept 2013, pp. 2835–2840.

- [46] F. Ganji, L. Budzisz, F. G. Debele, N. Li, M. Meo, M. Ricca, Y. Zhang, and A. Wolisz, "Greening campus WLANs: Energy-relevant usage and mobility patterns," *Computer Networks*, vol. 78, pp. 164–181, 2015.
- [47] F. Debele, M. Meo, D. Renga, M. Ricca, and Y. Zhang, "Designing resource-on-demand strategies for dense WLANs," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2494–2509, Dec 2015.
- [48] M. A. Marsan, L. Chiaraviglio, D. Ciullo, and M. Meo, "A simple analytical model for the energy-efficient activation of access points in dense WLANs," in *Proc. ACM International Conference on Future Energy Systems (ACM e-Energy)*, May 2010, pp. 159–168.
- [49] A. P. C. da Silva, M. Meo, and M. A. Marsan, "Energy-performance trade-off in dense WLANs: A queuing study," *Computer Networks*, vol. 56, no. 10, pp. 2522 – 2537, 2012.
- [50] R. G. Garroppo, G. Nencioni, G. Procissi, and L. Tavanti, "The impact of the access point power model on the energy-efficient management of infrastructured wireless LANs," *Computer Networks*, vol. 94, pp. 99–111, Jan. 2016.
- [51] C.-Y. Li, C. Peng, S. Lu, and X. Wang, "Energy-based Rate Adaptation for 802.11n," in *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, ser. Mobicom '12. New York, NY, USA: ACM, 2012, pp. 341–352.
- [52] M. O. Khan, V. Dave, Y.-C. Chen, O. Jensen, L. Qiu, A. Bhartia, and S. Rallapalli, "Model-driven energy-aware rate adaptation," in *Proceedings of the Fourteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '13. New York, NY, USA: ACM, 2013, pp. 217–226.
- [53] S. Eryigit, G. Gur, S. Bayhan, and T. Tugcu, "Energy efficiency is a subtle concept: fundamental trade-offs for cognitive radio networks," *Communications Magazine, IEEE*, vol. 52, no. 7, pp. 30–36, July 2014.
- [54] A. Garcia-Saavedra, P. Serrano, A. Banchs, and M. Hollick, "Balancing energy efficiency and throughput fairness in IEEE 802.11 WLANs," *Pervasive and Mobile Computing*, vol. 8, no. 5, pp. 631 – 645, 2012.
- [55] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: Ran slicing for a flexible and cost-effective multi-service mobile network architecture," in *Proc. ACM MobiCom*, 2017, pp. 127–140.
- [56] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, and K. Kontovasilis, "Flexran: A flexible and programmable platform for software-defined radio access networks," in *Proc. ACM CoNEXT*, 2016, pp. 427–441.

- [57] A. Viridis, N. Iardella, G. Stea, and D. Sabella, "Performance analysis of openairinterface system emulation," in *2015 3rd International Conference on Future Internet of Things and Cloud*. IEEE, 2015, pp. 662–669.
- [58] N. Nikaein, "Processing radio access network functions in the cloud: Critical issues and modeling," in *Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services*. ACM, 2015, pp. 36–43.
- [59] S. Bhaumik, S. P. Chandrabose, M. K. Jataprolu, G. Kumar, A. Muralidhar, P. Polakos, V. Srinivasan, and T. Woo, "Cloudiq: A framework for processing base stations in a data center," in *Proceedings of the 18th annual international conference on Mobile computing and networking*. ACM, 2012, pp. 125–136.
- [60] "Small Cell Virtualization Functional Splits and Use Cases, Small Cell Forum Release 5.1 (159.05.1.01)," Tech. Rep., June 2015.
- [61] 3GPP, "Technical Specification Group Radio Access Network, Study on New Radio Access Technology, Radio Access Architecture and Interfaces (Release 14), 3GPP TR 38.801 V1.0.0," Tech. Rep., December 2016.
- [62] "Virtualization for small cells: Overview, Small Cell Forum White Paper (106.06.01)," Tech. Rep., June 2015.
- [63] EXTREME Testbed. [Online]. Available: http://networks.cttc.cat/mobile-networks/extreme_testbed/
- [64] I. Ucar, "Energy efficiency in wireless communications for mobile user devices," Ph.D. dissertation, Jul 2018. [Online]. Available: <http://hdl.handle.net/10016/27448>
- [65] K. Huang, K. Duffy, and D. Malone, "H-RCA: 802.11 Collision-Aware Rate Control," *Networking, IEEE/ACM Transactions on*, vol. 21, no. 4, pp. 1021–1034, Aug 2013.
- [66] D. Qiao, S. Choi, and K. Shin, "Goodput analysis and link adaptation for IEEE 802.11a wireless LANs," *Mobile Computing, IEEE Transactions on*, vol. 1, no. 4, pp. 278–292, Oct 2002.
- [67] P. Serrano, A. Garcia-Saavedra, G. Bianchi, A. Banchs, and A. Azcorra, "Per-Frame Energy Consumption in 802.11 Devices and Its Implication on Modeling and Design," *IEEE/ACM Transactions on Networking*, vol. PP, no. 99, pp. 1–1, 2014.
- [68] I. Ucar and A. Azcorra, "Deseeding Energy Consumption of Network Stacks," in *Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI), 2015 IEEE 1st International Forum on*, Sept 2015, pp. 7–16.
- [69] ITU-R, International Telecommunication Union, Jul., Recommendation P.1238, 2015.

- [70] The R Project. [Online]. Available: <https://www.r-project.org/>
- [71] J. A. Aroca, A. Chatzipapas, A. F. Anta, and V. Mancuso, "A Measurement-Based Characterization of the Energy Consumption in Data Center Servers," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 12, pp. 2863–2877, Dec 2015.
- [72] A. Chatzipapas and V. Mancuso, "An M/G/1 Model for Gigabit Energy Efficient Ethernet Links With Coalescing and Real-Trace-Based Evaluation," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2663–2675, October 2016.
- [73] J. Phung, Y. C. Lee, and A. Y. Zomaya, "Application-Agnostic Power Monitoring in Virtualized Environments," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, May 2017, pp. 335–344.
- [74] S. Leibson, "Reduce SOC Energy Consumption through Processor ISA Extension," in *2007 International Symposium on System-on-Chip (SoC)*, Nov 20107, pp. 1–4.
- [75] A. Jardosh, K. Papagiannaki, E. Belding, K. Almeroth, G. Iannaccone, and B. Vinnakota, "Green WLANs: On-demand WLAN infrastructures," *Mobile Networks and Applications*, vol. 14, no. 6, pp. 798–814, 2009.
- [76] J. Florwick, J. Whiteaker, A. C. Amrod, and J. Woodhams, "Wireless LAN design guide for high density client environments in higher education," Cisco, Tech. Rep. [Online]. Available: http://www.cisco.com/c/dam/en_us/solutions/industries/docs/education/cisco_wlan_design_guide.pdf
- [77] M. Papadopouli, H. Shen, and M. Spanakis, "Modeling client arrivals at access points in wireless campus-wide networks," in *Proc. 14th IEEE Workshop on Local and Metropolitan Area Networks (LANMAN)*, April 2005, pp. 6–12.
- [78] V. Paxson and S. Floyd, "Wide area traffic: the failure of Poisson modeling," *IEEE/ACM Trans. Netw.*, vol. 3, no. 3, pp. 226–244, Jun 1995.
- [79] A. Feldmann, A. C. Gilbert, W. Willinger, and T. G. Kurtz, "The changing nature of network traffic: Scaling phenomena," *SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 2, pp. 5–29, Apr. 1998.
- [80] G. R. Hiertz, D. Denteneer, L. Stibor, Y. Zang, X. P. Costa, and B. Walke, "The IEEE 802.11 Universe," *IEEE Comm. Mag.*, vol. 48, no. 1, pp. 62–70, Jan. 2010.
- [81] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains*. Wiley-Interscience, 2006.
- [82] O. C. Ibe, *Markov Processes for Stochastic Modeling*. Academic Press, 2009.

- [83] D. A. Johnson, David B. and Maltz, *Mobile Computing*. Boston, MA: Springer US, 1996, ch. Dynamic Source Routing in Ad Hoc Wireless Networks, pp. 153–181.
- [84] “High density wi-fi deployment guide,” Cisco, Tech. Rep. [Online]. Available: https://documentation.meraki.com/MR/WiFi_Basics_and_Best_Practices/High_Density_Wi-Fi_Deployment_Guide
- [85] “High-density wireless networks for auditoriums, version 3.3,” Aruba Networks, Tech. Rep. [Online]. Available: <http://community.arubanetworks.com/aruba/attachments/aruba/Aruba-VRDs/21/1/High-Density%20Wireless%20Networks%20for%20Auditoriums.pdf>
- [86] X. Li, P. Djukic, and H. Zhang, “Zoning for hierarchical network optimization in software defined networks,” *IEEE Network Operations and Management Symposium (NOMS)*, vol. 2014, pp. 1–8, May 2014.
- [87] X. Li and H. Zhang, “Creating logical zones for hierarchical traffic engineering optimization in sdn-empowered 5g,” in *Proceedings of the International Conference on Computing, Networking and Communication (ICNC)*, 2015, 2015.
- [88] F. Ganji, L. Budzisz, F. Debele, N. Li, M. Meo, M. Ricca, Y. Zhang, and A. Wolisz, “Greening campus wlans: energy-relevant usage and mobility patterns,” 2014. [Online]. Available: http://www.tkn.tu-berlin.de/fileadmin/fg112/Papers/2014/Ganji14greening_campus_wlans.pdf
- [89] A. D. L. Oliva, A. Banchs, and P. Serrano, “Throughput and energy-aware routing for 802.11 based mesh networks,” *Computer Communications*, vol. 35, no. 12, pp. 1433–1446, July 2012.
- [90] A. Garcia-Saavedra, B. Rengarajan, P. Serrano, D. Camps-Mur, and X. Costa-Perez, “Solor: Self-optimizing wlans with legacy-compatible opportunistic relays,” *IEEE/ACM Transactions on Networking*, vol. 99, 2014.
- [91] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, “A roadmap for traffic engineering in sdn-openflow networks,” *Computer Networks*, vol. 71, pp. 1–30, 2014.
- [92] N. Baldo, M. Miozzo, M. Requena-Esteso, and J. Nin-Guerrero, “An Open Source Product-oriented LTE Network Simulator Based on Ns-3,” in *14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, Nov 2011, pp. 293–298.
- [93] The network simulator - ns-3. [Online]. Available: <http://www.nsnam.org>
- [94] R. Martínez, A. Mayoral, M. Requena-Esteso, N. Baldo, R. Vilalta, R. Casellas, M. Miozzo, and J. M. R. Muñoz, “Application of SDN-based orchestration for the automated deploy-

- ment of fixed and mobile convergent services in future 5G networks,” in *1st International Workshop on Elastic Networks Design and Optimisation (ELASTICNETS)*, May 2016.
- [95] The Linux Foundation. The Real Time Linux collaborative project. [Online]. Available: <https://wiki.linuxfoundation.org/realtime/start>
- [96] “Reducing Switching Power with Intelligent Clock Gating, Xilinx White Paper (WP370),” Tech. Rep., August 2013.
- [97] H. Blasinski, F. Amiel, and T. Ea, “Impact of different power reduction techniques at architectural level on modern FPGAs,” in *1st IEEE Latin American Symposium on Circuits and Systems (LASCAS)*, Feb 2010.
- [98] P. Jamieson, W. Luk, S. J. E. Wilton, and G. A. Constantinides, “A Flexible Research Testbed for C-RAN,” in *2009 19th International Conference on Field Programmable Logic and Applications (FPL)*, Dec 2009, pp. 324–327.

