



HAL
open science

Performance Modeling of IEEE 802.11 WLANs

Marija Stojanova

► **To cite this version:**

Marija Stojanova. Performance Modeling of IEEE 802.11 WLANs. Networking and Internet Architecture [cs.NI]. Lyon1, 2019. English. NNT: . tel-02456055

HAL Id: tel-02456055

<https://hal.science/tel-02456055v1>

Submitted on 27 Jan 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N°d'ordre NNT :
2019LYSE1338



THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
l'Université Claude Bernard Lyon 1

Ecole Doctorale 512
InfoMaths

Spécialité de doctorat : Informatique

Soutenue publiquement le 16.12.2019, par :
Marija STOJANOVA

Performance Modeling of IEEE 802.11
WLANs

Modélisation des performances des réseaux locaux sans fil

Devant le jury composé de :

André-Luc Beylot	Professeur des universités, ENSEEIHT Toulouse	Rapporteur
Andrzej Duda	Professeur des universités, INP-Ensimag Grenoble	Rapporteur
Hind Castel-Taleb	Professeure des universités, Telecom SudParis-Evry	Examinatrice
Florent Dupont	Professeur des universités, Université Lyon 1	Examinateur
Nathalie Mitton	Directrice de recherche, INRIA Villeneuve d'Ascq	Examinatrice
Thomas Begin	Maître de conférences, Université Lyon 1	Directeur de thèse

To my parents and sisters,
the four best humans I know

Thank you...

The experiences of the last three years have been marked by an abundance of extraordinary people whom I had the immense luck of meeting for which I will always be grateful. Those that I have unwillingly omitted here will, I hope, write off my mistake to the frenzy of finishing a thesis and will forgive me over a beer.

To begin with the end, I will like to wholeheartedly thank the members of my jury, as having such distinguished scientists and kind people in this last stage was a joy on its own. Thank you André-Luc Beylot and Andrzej Duda for accepting to be the referees of this manuscript and for its in-depth analysis. My gratitude goes to Hind Castel-Taleb and Nathalie Mitton for showing interest in my work and accepting to be a part of its defense as members of the jury. I am thrilled that Florent Dupont, whom I have had the honor and privilege to call my professor and colleague, has also accepted the role of jury member and thus provide his invaluable contribution to this thesis.

My deepest gratitude is to my advisor, Thomas Begin. There are no words to say how thankful I am and how lucky I feel to have been able to be advised by someone I consider to be not only my favorite professor, a scientist of impeccable integrity and incurable curiosity, but most of all a kind and considerate person. Through every step of this thesis, Thomas has shown great compassion, has pushed me in the right direction when needed, and has given me the time and space to explore my own directions. Thomas always leads by example and I will forever aspire to becoming that same type of remarkable teacher and kind person.

My unofficial advisor Anthony Busson has been a constant source of scientific inspiration with a dose of daily humor. Through almost four years of collaboration, Anthony has never been less than an advisor and has always known how to put things in the right perspective. I will forever be grateful to Isabelle Guérin Lassous for being a devoted professor, a wonderful colleague, and an inspiration to so many future scientists. Most of all I thank Isabelle for helping in two crucial moments of my stay in France by giving me my first lecture in French and by suggesting that I contact Thomas for my Master's internship. The world of GSP would have been unknown to me had it not been for the patient and detailed explanations of Paulo Gonçalves. I thank Paulo for his scientific contribution and for also taking me out of my Markovian comfort zone. I am happy to have met Maxime De Freitas, the best intern we could have asked for, and I am grateful for his contribution to our GSP work. My gratitude goes to the first professor to show me the world of computer networks, Toni Janevski, whose devoted work and amazing lectures in the field will inspire many generations to come. Thank you Saïda, H el ene, Laetitia, Marie, Chiraz, and Sophie for the warm welcome, for making my life much happier

and French administration almost bearable.

Anyone who knows my family, or has spent a few minutes talking to me, knows that they are the pillar of my life and everything I am is because of them. My extraordinary parents and sisters are a never-ending source of support, motivation, inspiration, and laughter. My mom and dad gave me my love of learning and computers, they showed my sisters and me everything and let us choose anything, they believed all our choices were possible and did everything in their power (and beyond) to make it so. I consider myself immensely lucky to be the child of such an exceptional pair of humans. My sisters gave me more than I could ever say, thank you Bojana for France and Menka for Lyon. Thank you both for the security of knowing I will always have my two best friends and favorite people, and for showing me that there is indeed a life after the PhD.

My grandparents, as annoyingly proud as they are, have always given us all the love and support they could. Jana and Anika are as close to sisters as a person can have and my life would have been very different if our rendition of the Spice Girls was incomplete. I thank my sisters for bringing Philippe, Simon, Loïc, and Demian to our family, they are all perfect additions! Without naming all my family, thank you everyone!

To my family in the lab, you have made PhD life not only bearable but enjoyable. Jazmin is our grounding force, a combination of a best friend and a big sister that everyone should have, I cannot imagine my thesis and my life without her. This PhD has been forever shaped by the presence of Sam and Jacob, the two people that have made me laugh so much and that have made every day of the thesis better, I consider myself extremely lucky to call you both my friends. Do not go to L'heure de manger. I wish Sarra and Mohammed were here until the end so that we could have our office laughs, you are truly the most wonderful officemates. Though Renata and Miro were a late addition to the party, I am extremely grateful for all the coffee breaks and late nights that helped to keep my sanity. No coffee break is complete without our resident Italian, Lorenza, the living proof that good things come in small packages. I am grateful to our wannabe Italian Misha who has taught us that we can love someone and find them exceptionally annoying at the same time. Had it not been for Rodrigo in that first year, probably none of us would be the friends we are today, thank you Rodrigo for being our social glue and one of my greatest friends. Thank you Rémy for all the laughs and your valuable insight on WiFi, and thank you for bringing Margaux in our group. Thanks to Eugenia, my eastern ally, Pascal, the organic guy, Natalia, our Russian spy, Sebastien, our source of positive energy, Esteban and Gaetan, the cultured part of the lab.

My friends from Macedonia (and those that moved everywhere), thank you for the support and for making every vacation home memorable. I feel lucky to have shared well over a decade with you all and hope to share many more! My list of almost-sisters would not be complete without Katerina, arguably the strongest and wisest person I know, Marija, the best roommate and friend one can have, and Biljana, a late addition (I should have listened to my mom) that I feel I have know forever.

The only thing I regret about the end of this PhD is having to continue working without the daily support of my advisor and colleagues. Wherever I go next, you have all set the bar very high.

Contents

1	Introduction	1
1.1	Challenges	2
1.2	Performance Analysis	3
1.2.1	Performance metrics	3
1.2.2	Performance methodologies	4
1.3	Thesis Statement	7
1.4	Manuscript Organization	9
2	IEEE 802.11 Networks	11
2.1	IEEE 802.11 Standard Amendments	13
2.1.1	Frequency bands and channels	13
2.1.2	Data rates and modulation and coding schemes	14
2.1.3	Frame aggregation	15
2.2	Distributed Coordination Function	16
3	State of the Art	19
3.1	Legacy IEEE 802.11 including a/b/g	19
3.2	IEEE 802.11n/ac	22
3.2.1	Frame aggregation and MIMO	22
3.2.2	Comparing 802.11n and 802.11ac	23
3.2.3	Channel bonding	23
3.3	Machine Learning in WLANs	25
3.4	Summary and Open Problems	26
4	Opening Remarks	27
4.1	High-level description of a WLAN	27
4.2	Three Generations of Markovian Models	29
5	Model A: Arbitrary Topologies and On/Off Traffic	33
5.1	Introduction	33
5.2	Modeling Approach and Algorithm	34
5.2.1	Modeling approach	34
5.2.2	Algorithm	41
5.3	Numerical Results	42
5.3.1	Simulation setup and performance metrics	42
5.3.2	Validation	43
5.3.3	Applications	45

5.4	Discussion	47
5.4.1	Choosing the network simulator	47
5.4.2	Testing the impact of uplink traffic	49
5.4.3	Contributions and limitations	50
6	Model B: Larger and Heterogeneous WLANs	51
6.1	Introduction	51
6.2	Modeling Approach and Algorithm	52
6.2.1	The backoff factor α	53
6.2.2	Modeling Approach	55
6.2.3	Decomposing into subnetworks	55
6.2.4	Solving each subnetwork as one or more Markov chain(s)	56
6.2.5	Combining subnetwork solutions	61
6.2.6	Algorithm	63
6.3	Numerical Results	63
6.3.1	Model validation	64
6.3.2	Applications	70
6.4	Discussion	73
6.4.1	Modeling complexity	73
6.4.2	Contributions and limitations	75
7	Model C: Channel Bonding in IEEE 802.11ac	77
7.1	Introduction	77
7.2	Closed-form backoff factor α	78
7.3	Modeling Approach and Algorithm	79
7.3.1	Network decomposition	79
7.3.2	Representing a subnetwork as a Markov chain	80
7.3.3	Calculating output rates and achieved throughputs	82
7.3.4	Algorithm	83
7.4	Model Validation	83
7.5	Channel Bonding in IEEE 802.11ac	87
7.5.1	Impact of MCS and frame aggregation on channel bonding	87
7.5.2	Saturated networks with arbitrary topologies	89
7.5.3	Larger unsaturated network	92
7.6	Discussion	93
7.6.1	Contributions and limitations	93
7.6.2	Possible improvements	94
8	Graph Signal Processing for 802.11 WLANs	95
8.1	Introduction	95
8.1.1	Related Works	96
8.2	GSP for WLANs	96
8.2.1	Basic definitions: Graph and signal	97
8.2.2	Moving average filters	97
8.2.3	Adapting GSP to WLANs	99
8.3	Model Validation	102
8.4	Conclusions	105
8.4.1	Application	105
8.4.2	Contributions and limitations	106

9	Conclusions	109
9.1	Contributions	109
9.2	Limitations and Possible Extensions	110
9.3	Performance Evaluation for WLANs: Concluding Remarks	112

List of Figures

1.1	WLAN with two APs and their associated stations.	8
1.2	WLAN with three APs and their associated stations.	9
2.1	Comparison of infrastructure (left) and ad-hoc (right) mode in WLANs.	12
2.2	WLAN with four APs and eight STAs.	12
2.3	Available channels in the 2.4GHz frequency band.	14
2.4	An extract of the available channels in the 5GHz frequency band. . .	14
2.5	A-MSDU aggregation of three frames.	15
2.6	A-MPDU aggregation of three frames.	15
2.7	The DCF procedure for medium access.	16
4.1	A four-node network and its corresponding conflict graph.	28
4.2	Evolution of our modeling approaches.	30
5.1	Conflict graph of a four-node network.	35
5.2	Markov chain of the four-node network in Fig. 5.1. Edges with no arrows represent transitions in both directions.	40
5.3	Four-node network of Fig. 5.1: varying the input rate of node 2. . .	44
5.4	Conflict graph of a nine-node network.	44
5.5	Nine-node network of Fig. 5.4: varying the input rate of node 6. . .	45
5.6	Jain's fairness index and output rates for the network of Fig. 5.4 as a function of node 9's input rate.	46
5.7	Conflict graph of a ten-node network.	47
5.8	Jain's fairness index and network utilization for the network in Fig. 5.7.	47
5.9	Conflict graph of a three-node chain network.	48
5.10	Comparing ns2 and ns3's results for the network in Fig. 5.9.	49
6.1	Schematic representation of the proposed solution.	52
6.2	Conflict graph of a three-node FIM network.	53
6.3	Influence of the backoff factor α on node 2's output rate in the FIM network of Fig. 6.2.	54
6.4	Conflict graph of a four-node network.	55
6.5	Possible sending states and corresponding existing transitions asso- ciated to the subnetwork $b_{16} = [ON ON ON ON]$	57
6.6	Probabilities of entering each sending state, i.e., $\sigma_i(k)$, and the cor- responding weighting factors, i.e., ω_i^m for the subnetwork $b_{16} =$ $[ON ON ON ON]$	60

6.7	Four-node network of Fig. 6.4: varying the input rate of node 2. . .	65
6.8	Conflict graph of a six-node network.	66
6.9	Six-node network of Fig. 6.8: varying the input rate of node 6. . . .	67
6.10	Conflict graph of a ten-node network.	67
6.11	Ten-node network of Fig. 6.10: varying the input rate of node 4. . .	68
6.12	Frame aggregation for the network in Fig. 6.8: varying the input rate of node 6.	69
6.13	Different channel allocations for a randomly-generated 12-node net- work.	72
6.14	Number of (sending) states per Markov chain (subnetwork) as a function of the network's size and density.	74
7.1	Conflict graph of a three-node FIM network.	78
7.2	Resource sharing for a small α	78
7.3	Resource sharing for a large α	78
7.4	The backoff function in simulation and in the two models.	80
7.5	Conflict graph of a nine-node network.	84
7.6	Conflict graph of a ten-node network.	84
7.7	CDF of the relative error.	85
7.8	Nine-node network of Fig. 7.5: varying the input rate of node 1. . .	86
7.9	Ten-node network of Fig. 7.6: varying the input rate of node 10. . .	87
7.10	The maximum network throughput as a function of MCS with vary- ing channel size and frame aggregation rate.	88
7.11	Enumeration of available channels.	90
7.12	Initial conflict graphs of the four-node networks.	91
7.13	Enumeration of available channels.	92
7.14	Channel allocation solutions for maximizing different performance metrics.	93
8.1	Classic signal processing (left) and GSP (right).	98
8.2	Conflict graphs of five-node networks.	100
8.3	Conflict graph of a two-node network.	100
8.4	Nuisance value experienced by node 2 in Fig. 8.3 when node 1 is varying its input rate and MCS index.	101
8.5	WLANs used for the model validation.	103
8.6	Results for the 42-node network of Fig. 8.5a.	104
8.7	Results for the 41-node network of Fig. 8.5b.	104
8.8	Mean squared error as a function of the number of samples in the training set.	105

List of Tables

2.1	Data rates in Mbps, using short guard interval.	15
2.2	The DCF parameters for different IEEE 802.11 standard amendments.	17
4.1	System notation.	29
5.1	Modeling notation.	34
5.2	Distribution of the absolute errors for the output rates, y_n	45
5.3	Comparison of downlink and uplink traffic in real-life WLANs.	49
6.1	Modeling notation.	53
6.2	Simulation parameters.	64
6.3	Default input rates.	65
6.4	Distribution of the relative error for the throughput, t_n	65
6.5	Data rates of the nodes in the six-node network in Fig. 6.8.	68
6.6	Heterogeneous data rates: distribution of the relative error for the throughput, t_n	68
6.7	Frame aggregation: distribution of the relative error for the throughput, t_n	69
6.8	Input rates of the 12-node network in Fig. 6.13a.	71
6.9	Evaluation of the proposed channel allocations.	71
6.10	Evaluating the gain in upgrading to IEEE 802.11n.	73
7.1	Traffic parameters for the nine-node network.	84
7.2	Traffic parameters for the ten-node network.	84
7.3	Input parameters, demanded and obtained throughputs for the ten-node network.	93

Abstract

It is virtually impossible to name all the spheres of society that have been profoundly changed by the widespread of the Internet or to measure its impact inside each sphere. However, a consensus opinion of network experts is that this influence will only grow in the coming years. In the networking community, we are expecting an ever-increasing amount of traffic that will more than ever depend on wireless technologies, specifically Wireless Local Area Networks (WLANs).

The increase of traffic volume means that standardization organisms, vendors, and network architects need to come up with solutions for more coverage and capacity for WLANs. Given the distributed nature of resource sharing in 802.11-based WLANs, these solutions can become inefficient when they amount to simply deploying a larger number of resources. Thus, proper configuration and deployment of the networks is crucial to their performance.

In this thesis, we propose stochastic performance modeling approaches for WLANs. All our models are designed for unsaturated WLANs with arbitrary topologies. The first three models are based on Markov chains and model the network at a high level of abstraction. Each new model refines its predecessor by being conceptually simpler and at the same time closer to the real system. Our last Markovian model is tailor-made for IEEE 802.11ac WLANs and incorporates channel bonding, MCS indexes, and frame aggregation.

The increasing system fidelity of the models and their precision have allowed us to propose several different applications regarding the performance evaluation and configuration of centrally-managed WLANs. In particular, we are interested in issues of fairness and throughput maximization and propose several approaches that can help a network administrator to properly configure a network given a certain performance metric.

Our last modeling approach is profoundly different and incorporates a Graph Signal Processing (GSP) method for the performance modeling of WLANs. The need for such modeling arises mostly from scalability issues, as even though our Markovian models' accuracy makes them suitable for many applications, their lack of scalability can sometimes be seen as restrictive. We show that this black box approach can be successfully used for modeling the network and that incorporating WLAN-specific knowledge can help increase the accuracy of the model.

The final chapter of this manuscript details the contributions and limitations of each modeling approach we proposed, including a discussion on potential future works and on general practices in the performance evaluation of WLANs.

Chapter 1

Introduction

The Internet is the first thing that humanity has built that humanity doesn't understand, the largest experiment in anarchy that we have ever had.

Eric Schmidt

It is hard to fully conceptualize today that a communication invention has unprecedentedly revolutionized not only how we share information, but how we generate and think of information. The Internet's revolution is hardly reserved to technophiles, as we see its massive impact in communication, entertainment, economy, education, health care, and even governance structures and environmental issues. As a result, the stakes today are much higher than simply providing an efficient Internet access for entertainment and communication.

Global connectivity has allowed for the development of novel trends in education and health care. Massive Open Online Courses (MOOCs) revolutionized the transmission of knowledge. According to the Class Central platform, in 2018 a total of 101 million students attended 11 400 different classes from over 900 universities [1]. Quantifying the full impact of e-learning is an impossible task, however its effects on knowledge sharing are undeniable and worldwide. The contributions to health care, on the other hand, are usually more supervised and better defined. The Internet of Things (IoT) for health care [2] has provided us with applications such as glucose level sensing, electrocardiogram, blood pressure, and body temperature monitoring, and medication management.

In the early years of the Internet widespread, many authors argued for its positive impact on environmental issues. The Internet provided an efficient platform for dematerialization resulting in a smaller power consumption and less greenhouse gas emissions, and office space and travel savings from telework [3]. Unfortunately, today's data shows that we may have guessed wrong. According to Climate Care [4], the environmental impact of using the web can be summarized in two parts: the manufacturing and shipping of electronic devices, and the powering and cooling of connected devices. The Information and Communication Technologies (ICT) industry is responsible for 2% of global carbon emissions, over a fifth of which are

due to data centers. With the predicted constant increase in generated data and deployment of data centers, the Internet will represent an even larger part of global environmental issues. As a result, current (and future) works will largely focus on making the Internet, and our use of it, more environmentally friendly.

The economic impact of the Internet is also far from negligible. The GSM Association's annual Mobile Economy [5] reports that by 2023, an estimated 60% of the world's population will have subscribed to mobile Internet services. In the same period, mobile services will account for 4.8% of GDP.

A large contributor to the massive spread of the Internet is the popularization of wireless technologies over the last two decades, including mobile technologies, e.g., 3G and 4G, and Wireless Local Area Networks (WLANs), commonly known as Wi-Fi. Smartphones have, for better or for worse, long surpassed their ancestor's role and today's mobile traffic represents more than half of global Internet traffic. However, mobile users often do not know that a large part of the Internet access on smartphones is provided by WLANs, as operators choose to offload the traffic to WLANs. In fact, Cisco's Visual Networking Index [6] (VNI) predicts an increase from 54% in 2017 to 59% in 2022 for offloaded traffic, resulting in an over eight-fold growth in the volume of offloaded traffic.

The ever-increasing demands for capacity and coverage have resulted in several generations of IEEE 802.11 standards for WLANs and the deployment of even more Access Points (APs). Both the development of new (and more complex) standards and the densification of WLANs result in a series of performance issues. Given the only increasing popularity of wireless communications, the performance evaluation of WLANs will remain an important research topic for years to come.

Challenges

The popularity of WLANs has led to a myriad of technological advances and standard improvements. Nevertheless, as it is often the case in communication systems, a common approach to increase the available capacity and coverage is to simply deploy more resources. In WLANs, the deployment predominantly concerns APs, whose numbers will have increased over four-fold from 2017 to 2022, going from 124 to 549 million devices worldwide [6]. While this increase effectively solves many of the coverage problems, if not properly executed, it potentially results in a marginal capacity increase. Since APs in close proximity have to share the available resources, WLANs suffer from their own *paradox of plenty*, as more is not always better.

A common network densification problem arises when APs select their communication channel [7]. Consider a simple WLAN with two APs, denoted by A and B , operating on the only available non-overlapping channels, a and b , respectively. Let us suppose that a third interfering AP, denoted by C , from a different WLAN is deployed close to AP B , but out of reach of AP A and that C is configured to use channel b . In this scenario, APs B and C need to compete for the same resource, channel b . From a local view though, both AP A and B have an optimal solution, as A experiences no competition and B would be competing with one other AP on either channel. Globally, however, an optimal configuration would be for APs A to switch to channel b and for AP B to use channel a . This global optimum is most

often achieved by implementing centralized control for the WLAN of APs A and B .

Centralized control is a popular topic in WLAN management. Several vendors, including Aruba [8], Cisco Mobility [9], Meraki [10], and Meru [11], have proposed such solutions that are currently available for public use. Their technologies rely on the software-defined networking (SDN) paradigm in which a centralized controller manages all the WLAN's APs. The software used by the controller is regularly updated and can be defined to suit a particular user's demands. With the increasing presence of WLANs and their densification, and especially taking into account the predicted increase in WLAN-native and offloaded traffic, it is reasonable to assume that SDN will constitute an important part of the future of WLANs.

As appealing as centralized control is to anyone that has spend time working on WLAN performance issues, it should be underlined that today's solutions still have years of improvements to come. If the reader is familiarized with the Medium Access Control (MAC) layer enhancements of the recent 802.11 standards, they understand that properly configuring a WLAN is far from a simple task. If not, we have the pleasure to introduce the reader to the intertwined existence of standard amendments, channel bonding, frame aggregation, and MCS indexes in Chapter 2. Still, an avid connoisseur of WLANs knows that the challenges of their performance evaluation are numerous and can often be divided into two groups: those relating to choosing the appropriate performance metric and those more interested in the performance methodology. In the next section we briefly review some of the possible choices available and discuss the difficulty of properly designing and validating WLAN performance evaluation tools.

Performance Analysis

Due to the complexity of communication networks, and in particular the ever-growing complexity of WLANs, crucial trade-offs lie in the initial evaluation choices. As a result, the early stages of the development of performance evaluation tools for WLANs are always challenging and become easily overwhelming. Researchers have significantly differing, and often equally valid, opinions on the topic and this section summarizes only the author's personal views. In an effort to provide a concise summary and discussion on the topic, we focus on two questions:

1. What is the network behavior of interest?
2. How do we evaluate the current behavior?

Performance metrics

In communication networks, performance metrics allow us to measure certain quantities of interest to the network provider and/or its users. In WLANs, common metrics include delay, jitter, error rate, throughput, and, fairness. Choosing the right performance metric, or their combination, will then depend on the desired network behavior.

The average delay, jitter, and throughput are mainly significant to streaming applications, i.e., applications such as video or game streaming and voice that

generate data at (mostly) regular intervals and need the data to be delivered with the same regularity. However, most of the time these applications allow larger error rates, because losing a single frame of a video usually does not diminish the overall viewing experience. A completely opposite example would be email services that are fully elastic, as even delays of several minutes do not disturb the application, however they are intolerant to losses. Between these two extremes, a multitude of applications exist that have different performance requirements. While these metrics mostly focus on the experience of a single network device or application, we can also measure quantities such as the fairness of resource sharing that relate to the network as a whole.

Choosing the right metric for a given performance evaluation approach is a crucial decision. Not only does the choice affect the type of result and evaluation we can expect from the approach, but it often defines the complexity of the approach. An illustrative example is the modeling of delay and of throughput in WLANs. While it is possible to obtain accurate estimations of a device's achieved throughput using only high-level descriptions of the network, modeling delay often requires representing the transmissions of single packets and detailed buffer descriptions. Hence, choosing the proper metric of interest for any performance evaluation tool can be a make or break factor, as it is virtually impossible to represent all the WLAN's performance metrics in a tractable manner.

Performance methodologies

The three main approaches in WLAN performance evaluation are modeling, simulation, and experimentation. When choosing the appropriate approach for a given problem, it has to be clear that there are intrinsic trade-offs that are fundamental to every one of them. Most of the time, the development of a performance evaluation tool is goal-driven and has well defined objectives that need to be met. Thus, instead of reviewing the approaches separately, we discuss the five trade-offs of every approach that seem essential to the author. After careful consideration of these trade-offs, it should be relatively clear which evaluation approach is suitable for which application.

Fidelity

Analytical models have always been the battlefield of practitioners and theorists. Practitioners argue that analytical models are often too far from reality, while theorists often forget that communication networks do exist in reality. More particularly, in WLANs many critiques are concerned with the definitions of propagation and error models, the perfectly shaped communication zones, the disappearance of obstacles and interference, and the distributions of packet sizes and arrivals. However, analytical models are, by definition, only modeling the existing system and as such they are expected to have a given number of simplifying hypotheses. A discussion on the validity of those hypotheses is always welcomed, as finding the answer to "how simple is too simple?" is rarely an effortless exercise. My personal opinion is that a model's fidelity should be application-driven, i.e., fidelity is as important as the model's intended usage needs it to be.

Probably the most often encountered criticism of simulation, and thus advocacy for experimentation, is the fidelity of the approach to the original system. Simu-

lation can often be seen as a highly detailed and fully automatized model. This view puts simulation somewhere between models and experiments, as they are still simply modeling parts of the real functioning of the network, however their level of detail brings them closer to experimentation. Given its later widespread as a performance evaluation technique, as compared to theoretical and experimental work, the simulation of communication networks has experienced several crises of credibility. In the early 2000s, Pawlikowski et al. [12] reviewed over 2200 simulation-based articles and summarized three main points of criticism: *i*) use of proper random number generators, *ii*) detailed and valid statistical analysis of the obtained outputs, and *iii*) complete descriptions of the simulation results allowing for reproducibility. Since then, methods of random number generation have significantly improved and if properly used they are rarely an important point of disagreement. The two other points, however, are not linked to simulation as a performance evaluation tool, but to the publishing practices. It is the experience of the author that a large portion of these practices are, regrettably, still prevalent both in journal publications and conference proceedings. Modern criticism of simulation in WLANs usually refers to the bridging the gap between simulation and experimentation in the Physical (PHY) and Medium Access Control (MAC) layers. It should be noted that substantial advancements have been made in WLAN simulation. A series of works have focused solely on improving simulators by implementing realistic code for Software-Defined Wireless Networks (SDWNs) [13], environmental noise [14], or introducing experiment-driven simulator improvements [15, 16].

Finally, the holy grail of system fidelity has always been experimentation. In the last several years, 802.11 devices have become both widespread and reasonably inexpensive, making place for a new wave of experimental WLAN performance evaluation. The fidelity of experiments is less frequently questioned, as it is logical to think that nothing could be closer to the real system than that system itself. Today, testbeds are becoming more and more common and are being deployed in either anechoic chambers or public spaces. In both scenarios, usually devices are uniform and the traffic we monitor is generated solely for the performance evaluation purposes. However, several works show that there is a substantial gap between the 802.11 standards and the actual mechanisms implemented in Network Interface Cards (NICs) [15, 17, 18]. Furthermore, anechoic chambers completely isolate the WLAN from external influence, forcing us to rethink the realism of data obtained in fully controlled environments with fully controlled external factors. Still, it is important to note that the fidelity of experimentation to the system relates almost exclusively to the reproducibility of the results and the sufficient generality of the experimental setup.

Reproducibility

The reproducibility of experimental work, or lack there of, has been a recurring topic in many fields. In WLAN performance evaluation, many early experimental works only considered small topologies of a handful of nodes in highly homogeneous networks and traffic conditions. Recent works, to the satisfaction of the community, have incorporated larger WLANs with several dozens of nodes in many different configurations. However, the lack of uniformity of the network devices from different vendors, the lack of transparency in the experimental procedure, and the

volatility of WLANs to a changing environment often aggravate the reproducibility issues.

In simulations and analytical modeling, the main problems of reproducibility are the same as identified by Pawlikowski et al [12] in 2002. Published works experience a profound scarcity of detailed simulation setups and code availability in the case of modeling. Fortunately, many existing initiatives are successfully encouraging transparent publications, and their impact is far from negligible. When the all necessary code is made available, reproducing simulation and analytical results is usually fairly simple.

Cost

The cost of a performance evaluation approach usually relates to time and money. Developing an analytical model from scratch requires a significant amount of time and expertise, and should be avoided when immediate results are needed. However, once the model is fully developed, obtaining output is usually notably faster than in simulations or experiments. Additionally, analytical models can be developed in fully open source environments and thus require almost no additional funds.

Simulation tools usually require a reasonable short time to master and many of the available WLAN simulation environments are open source. Equipment costs are usually similar for analytical models and simulation setups, with the benefit of the simulation being that the code that reproduces the network's behavior is already written. However, depending on the complexity of the simulated network, simulations can have an unreasonably long execution time when no parallelization is enabled.

Experimentation is historically by far the costliest methodology, as not only is the equipment potentially very expensive, but requiring the knowledge to properly setup and run experiments takes a notable longer period of time than in simulation or modeling. Today, many of the cost problems of experimentation have been solved, or drastically reduced, by the deployment of open testbeds. The testbeds have an already completed and fully detailed experimental setup that is usually accompanied by a reasonably user-friendly access. Once the experimental setup is mastered, executing scenarios is faster than in simulation.

Scalability and flexibility

When it comes to scalability and large deployments, simulation and analytical models have an upper hand on experimental work. Increasing the network from 10 to 100 APs in simulation mostly requires very little code changes. Deploying 90 APs in reality takes a lot more than writing several lines of code. Still, it should be noted that large-scale simulation and modeling do take longer to execute and can require a significant amount of computational resources.

We refer to the process of easily modifying the network's parameters as flexibility. Flexibility is especially important when different network configurations are being tested in order to find an optimal set of parameters. In analytical models and in simulation, flexibility is straightforward: we simply modify the value of the parameter we wish to evaluate, provided that parameter is taken into account in the code. In experimental works, the flexibility of the network depends, once again, on the chosen network devices. Another advantage of today's testbeds is that they

are often configured for remote use and are almost fully adaptable to the demanded scenarios.

Finally, it is the author's conviction that each performance evaluation approach has its merits and appropriate applications. Fast and not costly results most often call for the simulation and modeling of networks. When it is important to keep the output closer to that of a real WLAN, then experimentation is most often the right solution.

The three approaches have quite different learning curves. Simulation, most often, can be used by anyone capable of running the corresponding simulation script. Setting up experiments can be almost as simple as running a simulation on existing testbeds, or it can require a significant effort when deploying a testbed from scratch. Developing analytical models requires an expertise in the underlying system. However, all three approaches have something in common: they provide us serendipitous discoveries. When analyzing the simulation and experimental results, we often find that parameters we have overlooked have a larger impact than previously thought. The development of models requires nothing short of a dissection of the network, as constructing an abstraction of a mechanism demands having a deep knowledge of that mechanism. This process often provides us with insight that is harder to achieve by simply looking at the network as a collection of inputs and outputs obtained in simulation or experiment. Ultimately, there is still a long road to full transparency in publishing results in all three approaches. As long as network configurations, simulation and experimentation setups, and modeling codes are not made available along with the published work, there will always be doubt to the validity of the results and the possible undisclosed hypotheses they might contain. However, it is the author's strong opinion that the lack of transparency is involuntary and represents a simple oversight in most publications, as shown by the increase of available research data over the last years.

Thesis Statement

Since the beginning of our work on 802.11 networks, the question we try to answer is how can we develop an efficient, accurate, and scalable performance evaluation tool for resource sharing in WLANs? We begin by stating the broad context and the motivation behind this thesis using several toy examples that we believe show the complexity of the task.

Let us consider a fairly simple WLAN consisting of two APs and their associated stations, pictured in Fig. 1.1. The two APs are in each others detection zones, depicted with the dashed circles, and thus are obligated to share the medium. To make matters simpler, we assume that the two APs are completely identical: fully saturated, generating packets of the same size, using links with the same transmission capacity. In such a WLAN, the fully distributed mechanisms of the IEEE 802.11 standard will enforce a round-robin type of resource sharing, i.e., on average the two APs take turns sending packets. As a result, both APs will attain the same throughput (roughly equal to half their transmission capacity).

Let us now assume that the green AP suffers from too many errors and decides to switch to a more robust modulation resulting in a link with a much smaller transmission capacity. Since the packet size remains the same, the green AP needs

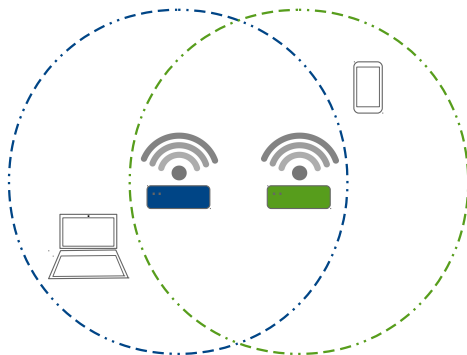


Figure 1.1: WLAN with two APs and their associated stations.

a longer time to send its packets over the now slower link. The network continues to function in a round-robin manner and the blue AP is forced to wait for the green AP to finish its lengthy transmission before every packet. Heusse et al. [19] referred to the phenomenon of the decline in throughput of the blue AP as 802.11's performance anomaly. Since then, the performance anomaly has been often revisited, as its practical implications are of high importance to all WLANs. In a household WLAN, the effect is often encountered when one of the connected stations moves far away from the AP and thus decreases the throughput of all other stations. Similarly, a neighboring AP can cause the exact effect if the two APs are within reach, as is often the case in residential buildings.

Now let us go back to the original scenario, where the APs are equal in capacity. A third AP is deployed close to the green one, with the same average capacity and packet size, shown in Fig. 1.2. In the best case scenario, the middle AP can hope for a round-sharing of the resources and thus obtain roughly a third of the available transmission capacity. Unfortunately for the middle AP, the three-way round-robin only is an exceptional phenomenon and in most cases, the middle node will spend most of its time waiting for the medium to be idle, as it is occupied by one or both edge APs. This particular network topology is known as a Flow In the Middle (FIM) network, studied in depth by Chaudet et al. [20], and widely known for its unfairness towards the middle AP that experiences throughput starvation.

The two networks we have considered are quite simple, their nodes are fairly homogeneous as they have the same packet sizes, they are always saturated, and only sometimes their capacities are differing. If these issues happen with networks of two or three APs, how do we manage larger networks? Even more so, how do we manage unsaturated networks, with highly heterogeneous nodes, and differing traffic demands? How do we centrally evaluate and control a network that is distributed by nature? Given the trade-offs of the different performance evaluation methodologies, we opted for stochastic models of WLANs that allow us to have a highly flexible and potentially scalable tool at a low cost with easily reproducible results. The system fidelity of an analytical model is likely to be its largest drawback, as many simplifying hypothesis have to be made in order to keep the model tractable.

Given the complexity of even small WLANs we seldom find models that are

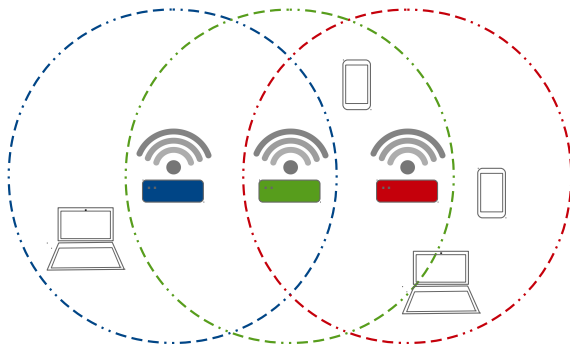


Figure 1.2: WLAN with three APs and their associated stations.

widely applicable without having highly limiting constraints on the network type. These limitations often come from the complex nature of the WLANs, but also from the fact that commonly the network is modeled at a very fine level of abstraction. The works presented in this thesis have a more global view of the network. Hence, instead of modeling the processes happening inside each node, we choose to model the whole network as a set of nodes. The high-level modeling approach makes it possible for all the models we propose to be applied to any WLAN with an arbitrary topology and unsaturated nodes with different traffic demands. Our models, however, provide only an estimation of the achieved throughput of every node, and do not model metrics such as buffer occupation or delay. Given a WLAN topology composed of APs and the knowledge about their traffic demands, our models estimate the resource sharing, i.e., the attained throughputs of all nodes. Such models allow for a fine tuning of already existing centrally managed WLANs or be used as guidance tools for deploying new networks.

Manuscript Organization

The organization of the manuscript is as follows: Chapter 2 describes the details of the internal mechanisms of IEEE 802.11 WLANs. Chapter 3 then reviews some of the existing approaches in performance evaluation of WLANs and discusses several open problems. In the interest of clarity, we include in Chapter 4 several opening remarks that detail the exact system we consider in our modeling approaches and a short summary of our Markovian models that should help the reader to better understand the reasoning behind their presentation in this manuscript and their development during the thesis. We advise that Chapter 4 be read before any modeling chapter. Chapters 5, 6, 7 present our three generations of Markovian models. Each of the three chapters includes a detailed description of the corresponding modeling approach, an algorithmic version of the model, its numerical validation and potential real-world applications. Our last model is presented in Chapter 8 and represents a work that is fully independent of our Markovian models and can also be read independently. Chapter 8 still respects the system defined in Chapter 4, however it also includes a review of the state of the art works in the field of Graph Signal Processing. We conclude with Chapter 9 that summarizes the

author's views on the contributions of this thesis and discusses potential improvements and drawbacks of the our modeling approaches and the field of performance evaluation of WLANs in general.

The modeling approaches presented in Chapters 5, 6, 7 resulted in conference and journal publications of the author, their supervisor Thomas Begin, and two collaborators. The Markovian models are joint work with Anthony Busson that was presented in the WiOpt conference in 2017 [21] and published in Performance Evaluation in 2019 [22]. Our initial Graph Signal Processing model is a collaboration with Paulo Gonçalves published in the Gretsri conference in 2019 [23] (in French). This collaboration later led to the work presented in Chapter 8, a joint effort with Paulo Gonçalves and Maxime De Freitas.

Chapter 2

IEEE 802.11 Networks

In the end of the 20th century, the Internet's popularity was experiencing constant growth in the industrial, commercial, and domestic spheres. Users were connecting more and more devices and it quickly became evident there is a need for mobility and wireless connectivity. Wireless Local Area Networks (WLANs) became the answer to these needs. Although initially developed for industry-specific applications, WLANs quickly showed they have their place in the office and home environments. In 1997, the first IEEE 802.11 standard was released [24], defining the Physical (PHY) and Medium Access Control (MAC) layer protocols to be used in WLANs. Since then, the IEEE 802.11 standard has become the norm for WLANs and in this chapter we summarize its basic mechanisms as well as the improvements implemented over the years. At the end of this chapter, we detail the specifics of the system we consider throughout this thesis.

In WLANs, we distinguish two types of network devices: Access Points (APs) and user stations (STAs). A STA associates to an AP in order to establish a communication. A WLAN is simply a collection of interconnected devices that communicate in one of two possible modes: infrastructure and ad-hoc mode. Figure 2.1 shows an example of the two modes. In the infrastructure mode, all STAs are associated with an AP, meaning that all communication is centralized. Even when two STAs are fairly close to each other, all communication between them must pass through the AP. In the ad-hoc mode there is not a centralized AP and STAs communicate directly with one another. While ad-hoc modes are a popular choice for setting up a connection between a few devices, e.g., in LAN gaming, most of the WLANs in use today, and the WLANs of interest in this thesis, use the infrastructure mode.

In a wired network, two devices are directly connected by a physical link and the speed at which they can communicate depends mostly on the characteristics of that link. We refer to the physical speed of communication as the device's *data rate*. In wireless networks however, the interconnectivity and the data rate depend on the distance between the devices as well as other environmental factors. When two devices are in close proximity, they can communicate or detect each others' transmissions with only a few errors that can be detected and often corrected. Such connections allow the devices to use dense modulations and achieve high data rates. As the distance between the devices increases, it becomes harder

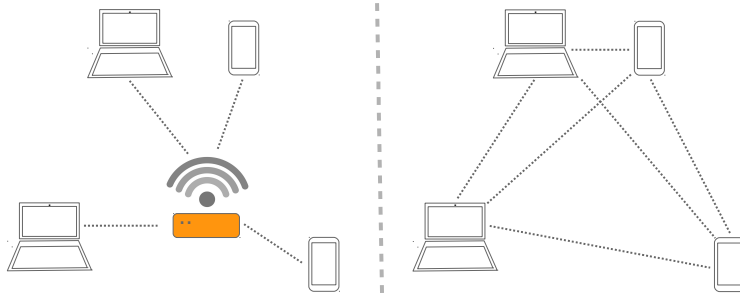


Figure 2.1: Comparison of infrastructure (left) and ad-hoc (right) mode in WLANs.

to correctly decode the information being exchanged as the signals received can significantly differ from those being sent. As a result, the devices are forced to use more robust modulations to ensure (most of) the messages are correctly received, i.e., use a large portion of the communication for encoding mechanisms facilitation error detection. If the distance increases even further, the devices completely lose all means of communication as they no longer detect each others' transmissions. These notions help us define the Carrier Sensing (CS) range. The CS range is the area in which a device detects the transmissions of all other devices, denoted as its *neighbors*. Because neighbors share the same resources, a collision occurs whenever two neighboring devices transmit at the same time, resulting in the potential loss of one or both frames. For example, Fig. 2.2 depicts an infrastructure WLAN with four APs connected to the backbone, and eight stations associated to the APs. The CS range of each AP is illustrated with a dashed circle around that AP. In this network, the blue and red APs could simultaneously transmit to their associated stations without collision. The yellow and red APs, on the other hand, have to take turns when transmitting to their stations.

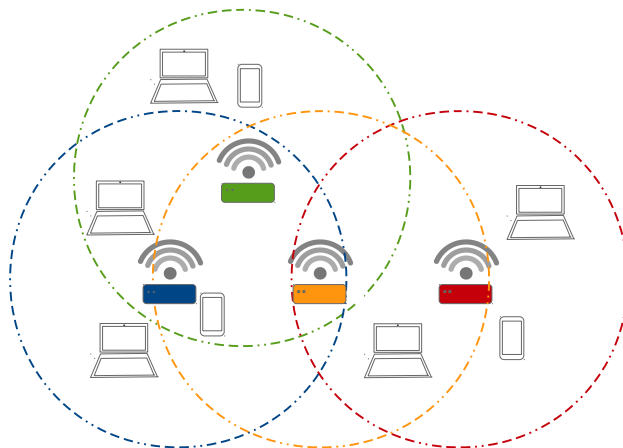


Figure 2.2: WLAN with four APs and eight STAs.

As neighbors have to share the available resources, there is strong incentive to have as few neighbors as possible. Intelligent planning of the positioning of APs as well as transmission power adjustments can significantly help in reducing the number of neighbors per AP. However, the increasing need for capacity and coverage dictates that more and more APs need to be deployed. This leads to network densification and generally low spatial reutilization, i.e., having several APs transmit at the same time because they are sufficiently far apart. As a solution, all IEEE 802.11 standard amendments allow APs to choose one of several radio channel that use different frequencies. Some of the channels are non-overlapping and can be used for simultaneous transmissions even when the devices are neighbors. The details of the available channels, data rates, and improvements in every new generation of IEEE 802.11 standard amendments are presented in the next section.

IEEE 802.11 Standard Amendments

When the original IEEE 802.11-1997 standard was released [24], the maximum data rate was 2Mbps. Today, the 2Mbps connections have been long surpassed by developing a series of techniques implemented in different 802.11 standard amendments. An 802.11 standard amendment is a collection of improvements and/or specific techniques to be implemented on top of an existing 802.11 standard that allow for the use of higher data rates, incorporate specific security requirements, or adapt to particular environments. For example, the 802.11a amendment introduced a new frequency band and higher data rates, the 802.11i enhanced the security of the transmissions, and the 802.11p adapted WLANs to the vehicular environment. When several amendments become available and are used in practice, the original 802.11-1997 standard is revised. This revision is referred to as a roll-up and is performed only once every several years, as modifying the entire 802.11 standard is a delicate task. So far, the standard has been revised in 2007, 2012, and 2016, with the 802.11-2016 being the most recent roll-up available. Below, we review several of the main improvements implemented in the IEEE 802.11 standard over the years.

Frequency bands and channels

WLANs operate mostly in two frequency bands: the 2.4GHz and the 5GHz band. Initially, only the 2.4GHz band was available in the 802.11-1997 standard. In the 2.4GHz band, there is a maximum of 14 radio channels 20MHz wide that APs can choose from, depicted in Fig. 2.3. As shown in the figure, neighboring channels are overlapping and there can be at most three channels used simultaneously with no overlap. Usually, the APs of a WLAN are configured to use channels 1, 6, or 11 in order to minimize the number of neighbors per AP and thus maximize the spatial reutilization of the network. It should be noted that depending on a country's legislation, not all channels are available for private use.

Not long after the first publication of the 802.11 standard, the 5GHz band was introduced in the 802.11a-1999 amendment. The 5GHz band offers a larger choice of channels, with 25 non-overlapping 20MHz channels available in five sub-bands: the UNII-1, UNII-2, UNII-2 Extended, UNII-3, ISM. In practice, most WLANs use UNII-1 and UNII-2 bands, depicted in Fig. 2.4. While the 5GHz band offers more

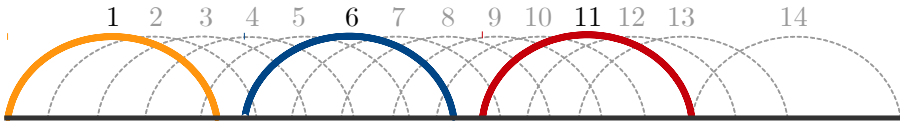


Figure 2.3: Available channels in the 2.4GHz frequency band.

channels and thus less interference, the shorter waves result in a smaller range than in the 2.4GHz band.

In older IEEE 802.11 standard amendments (802.11-1997,a,b,g), all frames are sent over 20MHz channels. As more channels became available, the idea of using larger channels emerged and led to the development of *channel bonding* in IEEE 802.11n. Channel bonding allows APs to use two contiguous channels as a single 40MHz channel. The result is over two times larger data rates, as the guard interval that was separating the channels can now also be used for transmissions. A novelty introduced in IEEE 802.11ac is the channel bonding of up to eight channels resulting in 160MHz channels and over eight times larger data rates, as shown in Fig. 2.4.

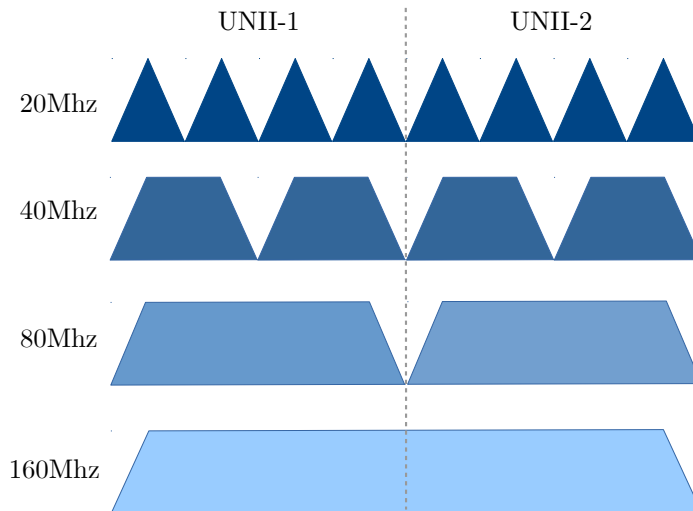


Figure 2.4: An extract of the available channels in the 5GHz frequency band.

Data rates and modulation and coding schemes

Each standard amendment defines a series of available data rates, the 802.11-1997 standard has only two data rates: 1Mbps and 2Mbps. The more recent 802.11ac [25] standard amendment (part of the 802.11-2016 roll-up) can reach data rates ranging from 6.5Mbps to over 6.5Gbps. These high data rates are achieved by introducing more advanced Modulation and Coding Schemes (MCS). Every connection in an 802.11ac WLAN has a given MCS index. This index is a combination of a modulation type (such as BPSK, QPSK, 16-QAM) and a coding rate (1/3, 2/3, 3/4...)

that result in a fixed data rate. The IEEE 802.11ac standard amendment offers ten different MCSs, denoted MCS0 to MCS9. The higher the MCS index, the higher the data rate of a node. When choosing an MCS, the network interface card's goal is to maximize the data rate, while keeping a relatively low packet error rate. Table 2.1 lists the data rates for different MCS values in 802.11ac and for different channel sizes. We notice that even for the same channel size, moving from MCS0 to MCS9 results in an over ten-fold increase in data rate. Overall, larger channels, along with high MCS values, result in much higher data rates.

MCS	0	1	2	3	4	5	6	7	8	9
20MHz	7.2	14.4	21.7	28.9	43.3	57.8	65	72.2	86.7	NA
40MHz	15	30	45	60	93	120	135	150	180	200
80MHz	32.5	65	97.5	130	195	260	292.5	325	390	433.3
160MHz	65	130	195	260	390	520	585	650	780	866.7

Table 2.1: Data rates in Mbps, using short guard interval.

Frame aggregation

In IEEE 802.11 protocols, one of the largest overheads a node experiences is waiting to acquire medium access. Legacy IEEE 802.11 standards send frames one at a time, meaning that the node spends a significant amount of time trying to access the medium, instead of actively transmitting data. A simple solution to this problem is to send several frames as a single aggregate frame. IEEE 802.11n proposes two types of frame aggregation: Aggregate MAC Service Data Unit (A-MSDU) and Aggregate MAC Protocol Data Unit (A-MPDU). An A-MSDU is a collection of several MSDUs that share a common MAC header and DCF overhead (see Fig. 2.5), resulting in a very efficient network utilization. A-MPDU aggregation is slightly less efficient, as several MPDUs share the DCF overhead however each MPDU has its own MAC header (see Fig. 2.6). However, in environments that are prone to error, A-MPDU allows the receiver to acknowledge MPDUs individually and effectively results in higher throughput than when using A-MSDU [26].



Figure 2.5: A-MSDU aggregation of three frames.



Figure 2.6: A-MPDU aggregation of three frames.

By default, all frames in the IEEE 802.11ac standard amendment are sent using A-MPDU (even if they contain only a single MPDU). The 802.11-2016 and

standard imposes new regulation on the maximum frame length when using aggregation. More specifically, when using A-MPDU aggregation the maximum size of the aggregated frame is limited by three conditions [25]:

1. at most 64 MPDUs can be aggregated in a single frame;
2. the duration of the Physical Protocol Data Unit (PPDU) must not exceed 5,484 μs ;
3. the maximum length of the aggregate frame cannot be more than 1,048,575 bytes.

Distributed Coordination Function

Up to now, we have mostly focused on the parameters that can increase the data rate in a WLAN. However, these mechanisms do not define how several devices share a common medium. In order to ensure an efficient sharing of the wireless medium between neighbors, WLANs implement the Distributed Coordination Function (DCF). DCF defines the procedure taking place before every frame transmission in order to avoid collisions. Before starting a transmission, the device first listens to the medium to make sure it is not occupied by one of its neighbors. Should the medium be occupied, the device defers its transmission until it senses an idle medium. Figure 2.7 schematically represent the DCF procedure for unicast frames and Table 2.2 summarizes the corresponding DCF parameters for several 802.11 standard amendments.

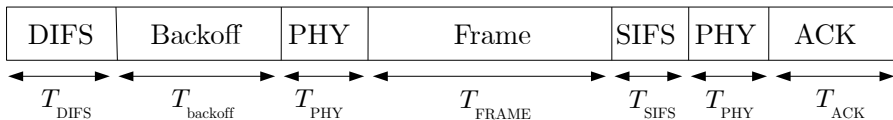


Figure 2.7: The DCF procedure for medium access.

Before a device starts a frame transmission, it first needs to make sure that the medium is continuously sensed idle for the duration of one DCF Interframe Space (DIFS) period, so as to avoid a collision with an ongoing transmission. Next, the device starts the *backoff* period whose goal is to desynchronize the beginnings of transmissions of neighboring nodes. The backoff's random duration makes it unlikely that neighboring nodes start transmitting at the exact same time once they sense that a common neighbor has stopped transmitting. Unlike the DIFS period, the backoff can be frozen if the medium is sensed busy and then resumed when the medium becomes idle again. The duration of each backoff period is calculated as the product of an integer value randomly generated in the interval $[0, CW]$ and the slot time T_{slot} , where CW is the current contention window size. Initially, the contention window is set to $CW = CW_{\text{min}}$. For every retransmission of the same frame, the CW is doubled until it reaches CW_{max} . Thus, for frames without retransmissions, the mean duration of the backoff period is:

$$T_{\text{backoff}} = \frac{CW_{\text{min}} \times T_{\text{slot}}}{2} . \quad (2.1)$$

Once the backoff countdown has finished, the physical header is sent and then the frame transmission (payload and headers) begins. We use frame here to designate a simple non-aggregated frame, as well as A-MPDU or A-MSDU frame. The duration of the frame transmission, T_{FRAME} is calculated as:

$$T_{\text{FRAME}} = \frac{(L + H) \times 8}{R}, \quad (2.2)$$

where L is the payload size in bytes, H is the sum of all headers from the MAC and upper layers in bytes, and R is the data rate in Mbps. The PHY header of the frame is sent at the lowest possible data rate, so that all devices can decode any PHY header they receive.

The last mechanism is the Acknowledgement (ACK) frame. The destination acknowledges every successfully received frame by sending an ACK to the source. If the source does not receive an ACK for a unicast frame, it supposed that the frame was lost and begins retransmission. As all other frames, the ACK frame is preceded by a physical header (see Fig. 2.7). It should also be noted that the ACK is sent using a data rate smaller or equal to the frame data rate and belonging to the basic data rate set. The basic data rate set is simply a set of rates defined in an 802.11 standard amendment that must be provided in every implementation of that amendment. In practice it means that all ACKs are sent using a data rate supported by all network devices.

Finally, the total transmission time, T , can be calculated:

$$T = T_{\text{backoff}} + T_{\text{DIFS}} + T_{\text{PHY}} + T_{\text{FRAME}} + T_{\text{SIFS}} + T_{\text{PHY}} + T_{\text{ACK}}, \quad (2.3)$$

and we can obtain the maximum achievable throughput for a give L and R as:

$$t_{\text{max}} = \frac{L}{T}. \quad (2.4)$$

There are two ways of increasing the maximum achievable throughput: increasing the payload size or the data rate. If we suppose that no frame aggregation is used and that the maximum frame size is 1500B, even with an infinite data rate, we would be limited to a maximum throughput of around 80Mbps in 802.11n/ac. This limitation comes from the SIFS, DIFS, backoff, and physical headers and shows the impact of DCF on the WLAN's capacity. However, if we increase the frame size by aggregating (either A-MSDU or A-MPDU), we can achieve the several Gigabit throughputs available in recent 802.11 standard amendments.

Parameter	802.11g/n@2.4GHz	802.11a/n@5GHz/ac
CW_{min}	15	15
CW_{max}	1023	1023
$T_{\text{slot}} (\mu\text{s})$	9 or 20	9
$T_{\text{backoff}} (\mu\text{s})$	67.5	67.5
$T_{\text{DIFS}} (\mu\text{s})$	28 or 50	34
$T_{\text{SIFS}} (\mu\text{s})$	10	16

Table 2.2: The DCF parameters for different IEEE 802.11 standard amendments.

Having described the basic mechanisms of 802.11 WLANs, we now proceed with a review of the performance evaluation works developed for them over the years.

Chapter 3

State of the Art

Ever since their first introduction in 1997, IEEE 802.11 networks have been a constant and important part of performance evaluation research in communication networks. As the standard evolved over the years, so did the works that modeled and analyzed them. In this chapter, we review a selection of those works and present their continuous development by dividing them in a section for legacy 802.11 including a/b/g, and a section for the more recent 802.11ac/n. Being of special interest to Chapters 7 and 8, we also separately and briefly review works that focus on channel bonding and Machine Learning (ML) in WLANs.

Legacy IEEE 802.11 including a/b/g

The introduction of WLANs in the late 1990s quickly prompted the development of a series of works dedicated to WLAN performance evaluation. Given the specific nature and the novelty of the underlying system, most of these works are analytical, constructive models of WLANs. Any review of WLAN models would not be complete without the mention of the seminal works of Cali et al. [27] and Bianchi [28]. Both works present analytical models of the network at a very fine level of abstraction by taking into account every single frame transmission. In [27] the authors analyze the ratio of the average frame size and its average transmission time in order to study the utilization of the network's capacity. Bianchi's work [28] introduces a model based on a two dimensional Markov chain. The Markov chain models the backoff process that takes place before every frame transmission in a fully connected network, i.e., all nodes are neighbors. A property shared by both models is that the networks they consider are saturated, i.e., all nodes constantly have backlogged frames waiting to be sent.

The saturation assumption can be deemed too restrictive in some cases, thus many subsequent works are centered on relaxing it. Kosek-Szott [29] as well as Gupta and Rai [30] circumvent this barrier by adding one more state to the Markov chain proposed by Bianchi [28]. This new state represents a node that has no frames to be sent. Another solution is proposed by Felemban and Ekici [31], who remove the saturation condition by introducing the probability that a node has a frame waiting to be sent. They do so by creating a second Markov chain, embedded into Bianchi's original Markov chain. The embedded chain describes the current state

of the channel, which can be either idle, in collision, or in successful transmission. The solution to their model is found by successively iterating between the two chains. Upon convergence, the found solution delivers the steady state transmission probability for each node, which can then be used to evaluate the network's performance. However, like Bianchi's original model [28], the focus of these works is restricted to fully-connected networks.

Aside from performance evaluation, a fine-level modeling can also help refine networking protocols in order to improve the overall network performance. To achieve this goal, Lee et al. [32] introduce the so-called Optimal DCF (O-DCF). O-DCF adapts a node's MAC parameters, such as the backoff period and transmission length, in order to improve the network's utilization or fairness. The adaptation depends on the current length of a node's MAC queue and tends to favor nodes that have more queue buildup. Even though the adaptation is calculated in a distributed manner for every node, an estimation of global performance metrics is required. This requirement is removed in [33] where Fitzgerald et al. introduce the Throughput Optimal DCF (TO-DCF) that needs only local measurements. TO-DCF also favors nodes with larger queue buildup, but it uses node weights to express the different priorities. Nodes with higher weights decrement their backoffs faster, which gives them a higher transmission priority. While O-DCF can be directly implemented in an existing 802.11 chipset as a driver update, TO-DCF modifies the DCF procedure and represents a new DCF-based protocol.

To overcome the inherent complexity tied to a fine level of abstraction when the network grows in size, other works have developed modeling approaches that incorporate both a fine-level and a high-level of abstraction. Two such models are the works of Shi et al. [34] and Begin et al. [35]. Both models analyze non-saturated multi-hop networks. In a multi-hop network, a packet from node A travels across relay nodes before arriving at its destination node B (as opposed to single-hop networks, where A and B directly exchange packets). Both papers present two-level modeling approaches of unsaturated multi-hop wireless networks, in which the low-level model is a version of Bianchi's original Markov chain, while the high-level model aims at capturing the inter-node dependencies in the network. The solution to the overall model is found using a fixed-point iteration between the high and low level. In [35], the high-level model consists of a set of $M/M/1/K$ queues, where each queue represents a given node of the network. Although their modeling framework was designed to handle any number of nodes, examples shown in their paper involve multi-hop wireless paths with at most 4 nodes. In [34], the high-level model is a separate Markov chain describing the channel's behavior depending on the current states of neighboring nodes, with nodes being either idle, transmitting, or in backoff. Because of the three possible states for each node and the added complexity brought by multi-hop networks, the analytical model of [34] leads to a large state space as the number of nodes increases, making it intractable for networks with more than 7 or 8 nodes, for which a decomposition into smaller networks is necessary.

Finally, at the other extreme of the spectrum, there are the modeling approaches that analyze the network from a high level of abstraction. These models do not take into account the behavior of every frame transmission, and instead, deal with the behavior of the entire network as a whole. In [36–38], Markov chains are used to model a network based on its topology. The states of the chain describe

the set of nodes that are transmitting in the current network state. Nardeli and Knightly [36] rely on their proposed Markov chain to derive a model that takes into account the errors due to collisions and hidden terminals for a single-hop network. The authors derive a closed-form multi-parameter expression of throughput, which is subsequently used for evaluating the performance of the considered network. Although the model accurately captures the behavior of CSMA/CA networks, it only deals with saturated networks and introduces some complexity due to the calculation of successful transmission probability. Durvy et al. [37] use a similar Markov chain to evaluate the fairness and spatial reuse in multi-hop, saturated networks with different carrier sensing and reception zones. More particularly, the authors study the spatial reuse in line-networks to show that CSMA/CA achieves maximal spatial reutilization as traffic intensity increases, at the cost of creating starvation in certain links. Boorstyn et al. [39] and Wang and Kar [40] model CSMA/CA networks as continuous time Markov chains and the model is then used to study the fairness of the network. Jiang and Walrand [38] extend the usage of this model by proposing an adaptive solution that changes the nodes' backoff periods in the goal of maximizing the network's throughput and utilization.

A significant number of models have been developed for specific network topologies, such as chain networks. Chaudet et al. [20] study the behavior of the three-node chain network known as the Flow In the Middle (FIM) topology. FIM networks are well-known for exhibiting high levels of inequality because, when placed in saturation, the edge nodes experience a high throughput while the middle node is in starvation. The authors model the network as a Markov chain in which every state contains, among other values, the idle time experienced by the middle node. They show that as this idle time increases, the inequality of channel access decreases, i.e., the middle node gets more channel access. They conclude that shorter transmission times favor equality at the expense of utilization. Recently, Ducourthial et al. [41] developed a model that is also based on the idle time experienced by a node. However, their model can be used on chain networks of arbitrary lengths. They use a set of equations where the variables describe the transmission probability of every node in the network. They show that chains with an even number of nodes manifest more equality, and that for very large chains the inequality of channel access vanishes around the 15th node. The authors prove analytically and in simulation that modifying either the data rate or the frame length of the edge nodes can drastically increase the fairness in the network.

A novel approach in the modeling of non-saturated networks is introduced in [42] and [43], where the authors have chosen to map the idle time of a node to a longer backoff period. This approach keeps the simplicity of a saturated network model by not explicitly representing idle states, and yet allows the study of unsaturated nodes. Kai and Zhang [42] propose a model that calculates the throughput of non-saturated CSMA/CA networks with arbitrary topologies. Laufer and Kleinrock [43] use a similar model to estimate the throughput of a node in a fully-connected CSMA/CA network using the ratio between the transmitting and the backoff periods of that node, its probability of successful transmission, and the channel capacity. The result is then used in the analysis of a network's capacity region, based on the nodes' throughputs under stability conditions. Bonald and Feuillet [44] also characterize both the capacity region and the stability of a wireless network. However, their work focuses on multi-channel networks in either ad-hoc

or infrastructure mode, and they propose a refinement to CSMA to achieve a more efficient and fair access to the channel in the infrastructure mode.

IEEE 802.11n/ac

The introduction of 802.11n and 802.11ac significantly changed the landscape of modern WLANs. These standards allow for the use of MIMO transmissions, frame aggregations, as well as bonding two 20MHz channels into a single 40MHz channel in 802.11n reaching data rates of over 600Mbps, or bonding up to 8 20MHz channels into a single 160MHz channel in 802.11ac to achieve Gigabit connections. We invite the reader to consult Perahia and Stacey [45] or the IEEE standards [46, 47] for a detail description of the PHY and MAC layer specifications of both standards.

Frame aggregation and MIMO

Given the high number of parameters available for configuration in 802.11n/ac, many works have focused on evaluating the impact of individual or combinations of parameters on the network's performance. Ginzburg and Kesselman [26] propose an analytical model to study the maximum throughput experienced in an 802.11n WLAN of a single Access Point (AP) and its station. They quantify the impact of different types of frame aggregation when using either UDP or TCP as transport layer protocols, and while having different levels of noise. They conclude that A-MPDU aggregation largely outperforms A-MSDU aggregation as the noise level increases. Lin and Wong [48] extend Bianchi's original model to study the performance of A-MSDU and A-MPDU aggregation in WLANs prone to errors. They compare the results delivered by their analytical model to those provided by the ns2 simulator and propose a frame aggregation adaption algorithm that chooses the optimal size of an A-MSDU for a transmitter with a given bit error rate. Ong et al. [49] perform an in-depth theoretical analysis of 802.11ac and 802.11n in a saturated, error-free network and show how the enhancements of 802.11ac (channel width and MIMO) can result in an 84% increase from 802.11n's throughput. The authors also show that a combination of A-MSDU and A-MPDU aggregation can result in a higher throughput than using either aggregation alone. Cha et al. [50] analytically compare Multi-User MIMO (MU-MIMO) to Single and Multi-User Frame Aggregation (SU-FA and MU-FA) in a network consisting of a two-antenna AP transmitting to two single-antenna stations. They conclude that when both stations have similar frame lengths MU-MIMO renders higher throughputs as it better utilizes the channel. When frame lengths are heterogeneous or the channel is fast-varying it is better to use MU-FA. Both MU-MIMO and MU-FA always outperform SU-FA due to the fact that SU-FA experiences a much higher MAC overhead. Kosek-Szott [51] introduces a new scheduling mechanism named DEMS based on the idea of decoupling Access Classes (AC) and Downlink MU-MIMO (DL-MU-MIMO). Simulation results on the DEMS mechanism show that it manages to increase the average throughput while decreasing delay in high priority ACs.

Comparing 802.11n and 802.11ac

As the 802.11n/ac equipment became widely available, many works focused on experimental performance evaluation of WLANs. Dianu et al. [52] study the performance of 802.11ac networks in an office environment. They test the maximum achieved throughput when the transmitter-receiver distance and the interference change and when the 802.11ac network cohabitates with an 802.11n network. They show that the network can easily achieve a throughput of 700Mbps when the transmitter-receiver distance is small. However, as any WLAN operating in the 5GHz frequency band, the network suffers from low throughput and bad connectivity when obstacles are present, such as load-bearing walls. Pelechrinis et al. [53] also show that high data rates in 802.11n are highly susceptible to noise and interference. The authors perform a study in an indoor testbed and emphasize the importance of properly combining packet size and MAC enhancements (e.g., channel bonding, MIMO) in order to improve network performance. Zeng et al. [54] test 802.11ac performance in an experimental indoor and outdoor setup. Their work on energy efficiency shows that there are several combinations of MCS indexes and number of spatial streams that result in the same throughput, but have different energy consumptions. They also measure the energy consumption on different channel widths to find that, contrarily to popular belief, larger channels, particularly the 80MHz channels, consume more in idle mode making them overall less efficient. Through a series of tests, the authors show that properly configuring the MAC parameters can result in a substantial energy gain while keeping the same throughput performance. Kriara et al. [55] also empirically study the impact of MIMO, channel bonding, MCS, and guard intervals on the performance of 802.11ac networks in a home and office environment. They perform tests in an environment with/without interference, using different link qualities and topologies. The authors use multiple linear regression on their experimental findings to show how the throughput and jitter are highly impacted not only by the choice made for a single parameter, but also by the mutual influence of several parameters.

Channel bonding

Since the introduction of channel bonding in 802.11n, many works have been focused specifically on modeling its impact on the performance of 802.11n/ac WLANs. The trade-off to be made in channel bonding is between interference and data rates, as wider channels mean higher data rates, but also larger neighborhoods and more interference. Bukhari et al. [56] provide a detailed discussion on channel bonding in wireless networks, as well as a survey of published works. We use this section to summarize some of the main approaches in analytical models, together with experimental and simulation studies.

Kim et al. [57] analytically study the achieved throughput of 802.11ac WLANs using a Markovian model similar to Bianchi's original work [28]. Like Kosek-Szott [29], they use additional states to represent an empty buffer. Their model takes into account channel bonding and it handles unsaturated networks with collisions, however the model is limited to fully-connected WLANs. Sree Vasthav et al. [58] model the coexistence of legacy and 802.11ac devices using Dynamic Channel Bonding (DCB). They use two 20MHz channels that can be bonded into a single channel for the 802.11ac users. The authors present a six-state discrete-time

Markov chain that explicitly models the 40MHz and 20MHz channels and from it they derive the saturation throughput for a fully-connected network operating in an error-free environment. Bellalta et al. [59] develop a Continuous Time Markov Network (CTMN) model for static channel bonding in WLANs. The model offers high accuracy on small and heterogeneous networks and the authors study both throughput and fairness to show that channel bonding increases the overall throughput even in dense networks, however this increase is often at the cost of creating localized starvation. As the WLANs increase in size the authors are forced to simplify the considered system (introduce a saturated and fully connected network with homogeneous nodes) in order to keep a tractable state space and a reasonable execution time. Barrachina-Muñoz et al. [60] extend [59] and introduce a similar analytical Markovian model for predicting throughput in saturated WLANs using different DCB policies in dense networks. They find that the DCB policy has a high impact on the network's performance and that inter-node dependencies are important even outside carrier sensing zones. They confirm the results of [59] and conclude that using the widest available channel can lead to maximizing the throughput of several nodes while creating starvation in others.

Deek et al. [61] study how the efficiency of channel bonding is impacted by the interference, RSSI, physical obstacles and channel leakage, as well as data rate and the transport protocol in 802.11n networks. They perform an extensive experimental study to conclude that both the network and environmental conditions play an important role when choosing an intelligent channel bonding solution and that a node's knowledge about its environment can greatly help in choosing the right channel width. In a testbed of 12 AP-station couples, Simić et al. [62] test the benefits of using larger channels in high-density, saturated WLANs. They find that the 80MHz channels are best suited for highly-dense networks, while lower-density networks can take better advantage of separate 40MHz or even 20MHz channels by providing independent simultaneous transmissions. Bhartia et al. [7] propose a measurement-based algorithm for channel assignment in centrally-managed Meraki networks implementing channel bonding. The measurements on the large Meraki dataset show that the number of 802.11ac-capable devices has grown from 18% to 47% from 2015 to 2017, emphasizing the importance of developing standard-specific methods for WLAN management. The authors caution the use of centralized decision-making algorithms, as not all network devices today support the Channel Switch Assignments (CSA) of 802.11h [63] and a channel switch causes an average 8 seconds delay on a mobile device. To mitigate this issue, they propose a penalty for channel switching when calculating the optimality of a new channel assignment.

Ravindranath et al. [64] evaluate and compare the performance of 802.11n and 802.11ac using the ns3 discrete-event simulator. They show that while the higher MCS indexes of 802.11ac offer a slight improvement, the actual benefit of upgrading the standard is the possibility to use wider channels of 80MHz and 160MHz. Malekmohammadi [65] also uses ns3 to study dense, saturated WLANs. The author concludes that using shared wider channels with lower transmission powers results in higher throughputs than using separate narrower channels. Moreover, they show how the starvation experienced by some nodes can be lessened by finely tuning the transmission powers and carrier sensing thresholds of those nodes.

Machine Learning in WLANs

Machine Learning (ML) methods are applied to different problems where large amounts of data are available. Communication networks, including WLANs, are a prime candidate for such methods. In WLANs, ML is used in vastly diverse topics: from traffic prediction and classification, to network security or QoS management. A detailed description of all ML methods in WLANs is out of the scope of this thesis, we however briefly summarize some of the proposed works and refer the reader to Boutaba [66] for an extensive review on the topic.

In WLANs, ML is often used for classifying different types of traffic or types of users that generate it. User classification and behavior prediction is often motivated by commercial goals. Manweiler et al. [67] experimentally show that there exist patterns of user behavior in public WLANs and that certain applications could leverage from using ML methods to detect those patterns. They propose a measurement-based Support Vector Machine (SVM) approach for predicting user length of stay at APs and suggest that such solutions can be used to prioritize certain user classes depending on their dwell time. Ruiz-Ruiz et al. [68] also develop a user classification method using large sets of WLAN traces and several ML methods. The authors show how such predictive methods can be useful in the case of facility planning.

Traffic classification is of great use in security applications, where it is highly important to properly distinguish malicious and regular data. Rasthofer et al. [69] propose a supervised learning method for classifying sources (i.e., code providing device-related data) and sinks (i.e., code collecting data) on Android devices. They show that a manual identification of all sources and sinks is virtually impossible, as certain sources can only pose a problem when combined together. The authors extract their own set of features, they test different ML approaches and show that their proposed application, *SuSi*, is often more accurate in detecting sources that are unnoticed by other similar applications. Koliass et al. [70] provide a detailed summary of common security issues in WLANs. The authors then use a public dataset to show how many different ML methods can be used to detect intrusions in WLANs with a high degree of accuracy.

Lastly, one of the most common applications of ML in WLANs is the development of positioning systems. WLAN positioning systems can be used complementary to GPS, as they can function both indoors and outdoors and the high presence of APs allows for a higher degree of precision. Pan et al. [71] use manifold co-regularization to build a multi-view adaptive representation of the environment based on Received Signal Strength (RSS) measurements at different time intervals. Online Sequential Extreme Learning Machines (OS-ELM) are used by Han et al. [72] to provide a fast and accurate positioning estimation. Their online recalculation approach allows for a dynamic adaptation of the estimation and thus a more accurate prediction than classic ELM methods. Figuera et al. [73] use SVMs with a priori knowledge about the network (such as previous RSS values) to localize indoor devices. The authors especially focus on the importance of including system-related knowledge when using ML methods for WLANs. A novel approach is introduced by Krieg et al. [74] where the entire ML operation is done before any estimation of the user's location. The authors develop a framework that recognizes elementary patterns of human locomotion (e.g., running or walking) based on

a real-life data set of measurements. The application is then capable of providing a meter-level precise indoor localization using a single measurement from the device's sensors.

Summary and Open Problems

The performance evaluation publications we reviewed can be broadly divided into analytical models [20, 27–44], and experimental (including ML) studies [7, 52–55, 61, 62, 67–74]. In the recent years, the affordability and availability of 802.11 devices have allowed researchers to publish more experimental works and rely less on simulation. While it is my opinion that experimental studies offer real-life insight that analytical models simply cannot provide, I believe the inverse to be true as well (as discussed in Chapter 1).

In this manuscript, we present four analytical models for WLANs. The three following chapters describe Markovian models for unsaturated WLANs with arbitrary topologies. While analytical models of WLANs are not a novelty, researchers still struggle to develop models that do not require the network to be saturated, fully connected, or have homogeneous nodes.

The reader will notice that our system evolves with our modeling frameworks, becoming more realistic over time and allowing us to represent channel bonding in 802.11ac in our last Markovian model. Given the high number of parameters that impact the efficiency of channel bonding, finding a single guideline for optimal configuration remains an open issue. We believe that the model we propose in Chapter 7 can help bridge the gap between the data we have from measurement-based studies and our analytical understand of the underlying behavior.

Our Markovian models are all constructive, i.e., they are based on our understanding of the system and they reproduce several indispensable WLAN mechanisms. The model in Chapter 8 describes, to the best of our knowledge, a novel, ML-based, descriptive model of WLANs. Using ML methods for WLAN performance evaluation, especially analytical ones, remains an issue to be tackled. Our last model is based on a black box approach in which we initially assume no a priori knowledge of the underlying system and simply use an input/output dataset to model the performance of the WLAN. Nevertheless, we quickly came to agree with Figuera et al. [73] and realized that implementing parts of our expertise knowledge into the black box can only be beneficial.

Before presenting the modeling approaches developed through the course of this thesis, we describe the exact system they model and discuss their general evolution and improvements over time.

Chapter 4

Opening Remarks to the Modeling Approaches

The three years of this thesis were essentially focused on answering the question: how can we develop an analytical model that helps make WLANs more efficient? In that simple question are hidden the notions of performance metrics and the choice on modeling tools we discussed in Chapter 1, as well as the description of a WLAN. As it is often the case when developing performance evaluation models, the first modeling decisions take place when the system description is being developed. Should the system be simple enough, as it is rarely the case, then its full description can be used as the ground truth for the model. WLANs fall into the category of complex systems, as they are made up of layers of functionalities, each offering a multitude of implementation choices developed over twenty years of evolution. It quickly became evident that an all-encompassing description of today's WLANs is a task that will almost inevitably fail, a reality that is clear to anyone who attempted to understand 802.11 networks by only reading the standard's documentation. Thus, our search was focused on a setup that is as simple as possible, yet allows us to draw conclusions applicable to real-life WLANs. Having already introduced 802.11 WLANs in Chapter 2 and a part of the existing modeling approaches in Chapter 3, we now provide the high-level description of the system considered throughout this thesis coupled with explanations of the reasoning behind our choices. We also include a short discussion on the development of our models, that should provide the reader with insight into the ordering and logic of this manuscript and the thesis itself.

High-level description of a WLAN

We consider a network of APs belonging to the same WLAN implementing an IEEE 802.11 standard amendment. We assume that not all APs belong to each others' CS ranges, i.e., the network is not fully connected and can have any arbitrary topology. We also assume that the CS ranges are symmetrical, if AP n detects AP m 's transmissions, then m also detects n 's transmissions. These assumptions make it possible to depict each WLAN as a conflict graph in which two nodes share

an edge if the corresponding APs are neighbors. Figure 4.1 presents a four-node network, on the left, and its corresponding conflict graph, on the right. We assign, in no particular order, an identifying number to every AP and we use N to denote the total number of APs in the network.

When looking at our conflict graphs, the first thing we notice is that the stations are not depicted. Our choice to describe WLANs using only their APs is motivated by the aim of providing a *simple* performance evaluation tool that potentially allows the discovery of a more efficient *configuration*. Avoiding the explicit representation of the stations largely simplifies the model, both conceptually and computationally. Reconfiguration, in the general case, can only be done on the network devices controlled by the network administrator/architect. These devices usually include only the APs, and it follows that we are more interested in how they share the available resources between each other. Moreover, it is worth keeping in mind that Internet traffic is highly asymmetrical [75, 76], i.e., user stations download much more data than they upload. According to the Cisco Visual Networking Index [6], this trend will not only continue, but also magnify in the future. In order to confirm the existence of this trend, we performed our own measurements that we present in detail at the end of Chapter 5 in Section 5.4.2. We also simulated networks where 10% of the network traffic is generated by the stations to find that it only impacts the output rates of the APs by 5.5%. While these results show that uplink traffic can be omitted at a reasonable loss in accuracy, we are fully aware that in some cases uplink traffic potentially has a significant impact on the network's behavior.

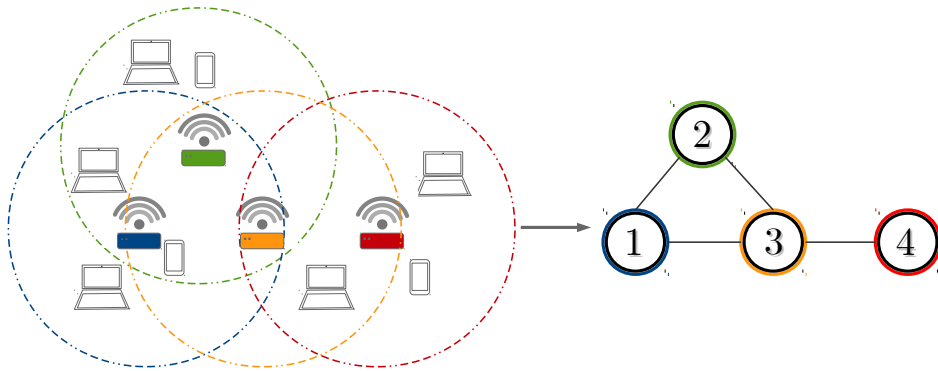


Figure 4.1: A four-node network and its corresponding conflict graph.

We presume that we have a certain knowledge regarding the average traffic parameters of every node, i.e., we know its frame size and aggregation rate (average number of MPDUs per transmission), data rate, and channel size. With all this information, we can calculate the maximum achievable throughput node n , denoted by $t_{n,\max}$, $n = [1, \dots, N]$ using Eq. (2.3) and (2.4). Node n also has a *demanded* and *achieved* throughput, denoted v_n and t_n , respectively. They characterize the rate the node wishes to obtain (in Mbps) and the rate the node does obtain. Using the demanded and the maximum achievable throughput, we derive node n 's *input rate*. A node's input rate, x_n , is simply the normalized version of its demanded throughput, so that $x_n \in [0, 1]$. Alternatively, x_n can be calculated as the proportion of time node n wishes to occupy the medium, be it in active

transmission or in DCF overhead:

$$x_n = \frac{v_n}{t_{n,\max}}, \quad (4.1)$$

where we assume that $v_n \leq t_{n,\max}$. Equivalently, we define node n 's output rate as the proportion of time node n occupies the medium, calculated as:

$$y_n = \frac{t_n}{t_{n,\max}}. \quad (4.2)$$

Depending on the node's position inside the network's topology and the throughput demands of its neighbors, a node may obtain all or a part of its demanded throughput. For example, in our four-node network of Fig. 4.1, it is clear that node 4 has the most advantageous position as it has only one neighbor with whom it shares the resources. Conversely, node 3 has to share the resources with all other nodes. However, should nodes 1 and 2 have $v_1 = v_2 = 0$, then nodes 3 and 4, in practical terms, have the same neighborhood. Nevertheless, the nodes' demanded throughputs and their traffic parameters still make it difficult to predict exactly how the two nodes share the available resources, even in an overly-simplified scenario. Over the years, many analytical models, simulations, and experimental solutions were proposed to solve the question of resource sharing in WLANs. In the following chapter we present a part of these works and we review the advantages and drawbacks of every approach.

Parameter	Description
N	total number of nodes
w_n	node n 's neighborhood, not including node n
R	data rate (in Mbps)
R_{ack}	data rate for the acknowledgement (in Mbps)
L	mean payload length (in bytes)
H	amount of headers brought by the MAC, Network, and Transport layers (in bytes)
v_n	demanded throughput of node n (in Mbps)
t_n	achieved throughput of node n (in Mbps)
$t_{n,\max}$	maximal throughput node n can achieve if all its neighbors are silent (in Mbps)
x_n	input rate of n -th node, $x_n \in [0, 1]$
y_n	output rate of n -th node, $y_n \in [0, 1]$

Table 4.1: System notation.

Three Generations of Markovian Models

The development of the Markovian models in this manuscript reflects to a great extent the advances made over the course of this thesis and even more so my understanding of the underlying system. Figure 4.2 captures an effect that will become evident to the reader after completing the three following chapters: over time our models became simpler and more accurate and our system more complex.

We begin with Model A, a framework that, at the time, seemed conceptually simple and accurate. When reading the manuscript however, the reader will notice that though the idea of the framework is easily comprehensible, its actual implementation (that includes a series of convoluted exceptions) does not offer a

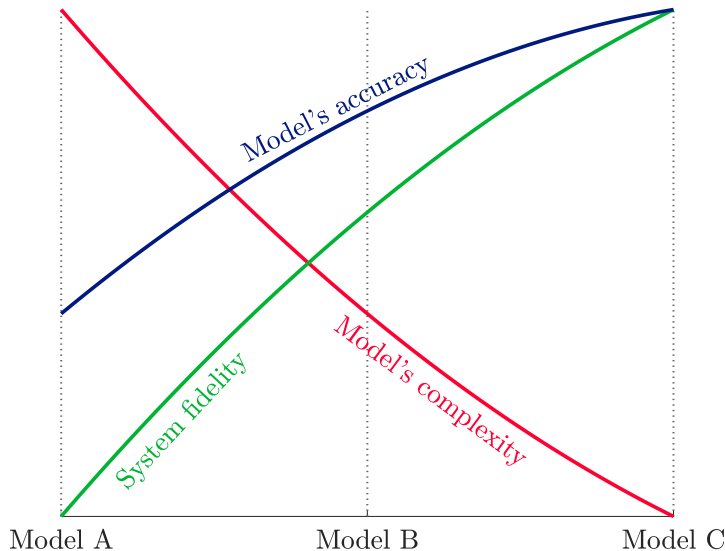


Figure 4.2: Evolution of our modeling approaches.

straightforward understanding. Had the considered system been closer to real-life WLANs, maybe such a complex model would have been well founded. The system we shall describe for Model A, however, is quite limiting and represents only a small fraction of the technologies used in today’s WLANs. Nevertheless, Model A provided us with solid learning ground and the development of all subsequent models and ideas would not have been possible without it.

The second model represents the largest part of the achievements of this thesis. The introduction of a Divide-and-Conquer approach allowed us to greatly simplify Model A: we no longer have two intertwined state spaces and we avoid most of the exceptional cases. By that time however, we were aware that Model A had limitations that were not linked to its complexity but to its applicability to WLANs. Being more familiarized with the system and the related works on the topic, we managed to introduce the idea of more heterogeneous nodes while still keeping Model B accurate.

Model B taught us about the importance of properly incorporating heterogeneity into the model. We learned that without introducing the backoff factor, all our modeling approaches would be ultimately limited in the accuracy they can achieve, as is the case for Model A. The greatest disadvantage of Model B comes from the fact that it was modified to accommodate heterogeneous networks, but not designed for them. The reader will notice this when comparing model B with our final model C, and the simplification of many solutions we incorporated. Our final Markovian model, Model C, is specifically designed for highly heterogeneous networks, and we show how such a framework can substantially improve the configuration of today’s WLANs.

As a final note, we wish to make it clear that all three models can be read separately and independently of each other. Should the reader be interested in the development and the detailed explanation of our work, we recommend the models

be read in order. If only one model is to be read, then Model C is the right choice, as it is our last and most advanced work. However, if the details of the modeling approach are more important to the reader than its possible applications, we advise that Model B be read instead.

Chapter 5

Model A: Arbitrary Topologies and On/Off Traffic

Introduction

Well before using Markovian models, we tested a series of different approaches for modeling WLANs. Our aim was always to develop conceptually simple solutions that model WLAN performance at a high level of abstraction, i.e., to view the network as a set of nodes instead of modeling each node in detail. One of the early methodologies we implemented was a resource sharing model based on the nodes' CS ranges using a system of equations which we then solved with nonlinear optimization. Each equation described the neighborhood of a single node in which the node needs to share the resources with its neighbors. After several iterations in which we slightly modified the impact nodes have on each other in the system of equations, we would always converge to a stable system whose solution provided us the achieved throughputs of all the nodes. While this model worked reasonably well on small and saturated networks, it was neither scalable nor applicable to unsaturated networks.

As we were looking for solutions to the network size constraints, we decided to test several different approaches that all heavily relied on the notions of cliques in WLANs. The more we studied these cliques, the more we were convinced that their impact on the behavior of WLANs is far from negligible [20, 41]. By separating the network into cliques, we were able to virtually scale-down the network size and process several nodes as a single clique. We modified our system of equations so that it models the behavior of cliques instead of single nodes. We then found that a simpler solution was to replace our system of equations by a Markov chain. In the Markovian model a node's medium access depended on the set of cliques to which it belongs, and the position of those cliques in the entire network topology.

Relaxing the saturation constraint was also a step-by-step process. As many other works presented in Chapter 3, our model initially handled only saturated networks. We then tested the impact of each individual node in the network by

Parameter	Description
N	total number of nodes
w_n	neighborhood of node n , $w_n \subseteq \{1, 2, \dots, N\}$
w'_n	restricted neighborhood of node n without synchronized, blocked, and preempted nodes
S	set of possible sending states, $S \subseteq \{0, 1\}^N$
s_k	k -th sending state, $s_k \in S$
$s_k(n)$	state of node n in sending state s_k , $s_k(n) \in \{0, 1\}$
A	set of possible activity states, $A = \{ON, OFF\}^N$
A_k^t	an activity state associated to sending state s_k , $A_k^t \in A$
$A_{k\ell}$	set of all activity states associated to sending state s_l and compatible with s_k
$P_{A_k^t}$	probability of occurrence of activity state A_k^t
Tr	set of possible transitions between sending states
$Tr_{k\ell}$	transition from sending state s_k to s_l , $Tr_{k\ell} \in Tr$
$P_{k\ell}$	probability of transition from sending state s_k to state s_l
U	network utilization, $U \in [0, 1]$
π	stationary probability distribution of the Markov chain that describes the network
π_{s_k}	steady state probability of sending state s_k

Table 5.1: Modeling notation.

running our Markovian model when that node is present in the network and when it is removed. We relaxed the saturation constraint for a single node by supposing there is a linear transition between the scenario when the node is absent from the network and when it is present (and saturated). Finally, we entirely removed the saturation constraint by introducing states in our Markov chain in which nodes are not sending, even though they sense an idle medium. This last relaxation allowed us to develop our first stand-alone Markovian model for unsaturated networks with On/Off traffic and arbitrary topologies and present it at the WiOpt symposium in 2017 [21]. In this chapter, we show that with only a small set of well-chosen parameters we are able to make a reasonably accurate prediction of the AP's obtained throughputs.

Modeling Approach and Algorithm

The modeling framework is based on a single Markov chain that describes the network at a high level of abstraction. In building the Markov chain, we first present the set of possible states and transitions, we then find the transition probabilities and calculate the stationary probability distribution of the chain, and finally, we compute the nodes' achieved throughputs. We conclude this section with an algorithmic description of the proposed model.

Modeling approach

We now describe our high-level approach to modeling a WLAN as defined in 4.1. We rely on a single Markov chain that copes with the well-known complexity of CSMA/CA-based networks, e.g., the random backoff, hidden node problems, or starvation. However, we deal with two intertwined state spaces, each describing differently and approximately the current state of the network. For the purpose of readability, we illustrate each step of our approach with an example, whose conflict graph is depicted in Fig. 5.1. We invite the reader to consult Tables 5.1 of this

chapter and 4.1 in Section 4.1 for a complete listing of the used notation in the model and in the system, respectively.

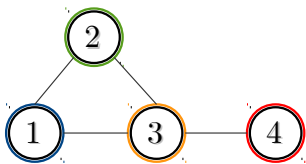


Figure 5.1: Conflict graph of a four-node network.

Our first state space, denoted by S and referred to as the *sending state* space, describes which nodes among the N are currently sending, and hence occupying the channel. We denote each sending state of S by s_k : $S = \{s_k\}_{k \geq 1}$. Each sending state s_k is a vector of size N in which the n -th value, $s_k(n)$, is set to 1 if node n is currently sending, and to 0 otherwise. For example, in the sample four-node network, the sending state $[0\ 1\ 0\ 0]$ denotes the scenario in which only node 2 is sending. Note that, in our work, a node's sending period includes the frame transmission itself, as well as all DCF and protocol overhead (see Eq. 2.3 of Chapter 2). Our objective is to derive a Markov chain representing the possible transitions between these sending states, to compute the probabilities of its state transitions, i.e., its transition matrix, and finally, to derive the steady-state probabilities of s_k .

In order to accommodate the fact that the network is unsaturated, we need to introduce a secondary state space, referred to as the *activity state* space and denoted by A . In any network, nodes typically alternate their activity between *ON* and *OFF* periods. When in an *ON* period, a given node n has at least one frame to send, and thus has a non-empty buffer. In other words, an *ON* node is either transmitting or willing to start a transmission. In the *OFF* regime, a node's buffer is empty. We consider that the nodes' regimes are independent of each other. As a consequence, the nodes' input rates are also independent of each other and can be calculated as:

$$x_n = \frac{T_{ON}}{T_{ON} + T_{OFF}}, \quad (5.1)$$

where T_{ON} and T_{OFF} represent the average amount of time node n spends in the *ON* and *OFF* regime, respectively. In reality, a node may postpone the transmission of a frame because of the activity of its neighbors, thus extending its *ON* period. In order to keep the model tractable, we decided to omit the potential dependencies among the nodes' *ON* periods. Analogously to S , each activity state, A_k^t , is a vector of size N whose n -th value denotes the activity of node n as being either *ON* or *OFF*. Let $P_{A_k^t}$ denote the (steady-state) probability that the activity state A_k^t occurs. The mutual independence of the nodes' input rates x_n enables us to obtain $P_{A_k^t}$ as follows:

$$P_{A_k^t} = \prod_{A_k^t(n)=ON} x_n \prod_{A_k^t(m)=OFF} (1 - x_m). \quad (5.2)$$

Note that a sending state, s_k , can typically be associated to one or up to T different activity states, each denoted by A_k^t with $t = 1, \dots, T$. While every sending

node necessarily has an *ON* activity, the inverse does not hold. A node that is not sending and has no sending neighbors is (undoubtedly) *OFF*, because if the node were *ON* then it would be sending as it detects an idle medium. However, if node n has a sending neighbor, we cannot be sure whether n is not sending because it has no frames to be sent (n is *OFF*) or because it detects a busy medium (n is *ON*). The result of this ambiguity is that several activities can be associated to a single sending state. For example, assuming a given sending state $s_k = [0\ 1\ 0\ 1]$ for our network of four nodes, i.e., nodes 2 and 4 are transmitting, a total of four activity states ($T = 4$) may be associated to s_k :

$$\begin{aligned} A_k^1 &= [OFF & ON & OFF & ON] \\ A_k^2 &= [ON & ON & OFF & ON] \\ A_k^3 &= [OFF & ON & ON & ON] \\ A_k^4 &= [ON & ON & ON & ON] \end{aligned} \quad (5.3)$$

Determining the sending states

An intrinsic property of 802.11-based networks is that two neighbor nodes cannot (or virtually not) be simultaneously transmitting. When two neighbors attempt to access the medium at the same time, a collision occurs resulting in the potential loss of one or both frames. In terms of our modeling approach, this means that any sending state of the network is possible as long as any two neighbors are not simultaneously sending.

In the case of our sample four-node network represented in Fig. 5.1, there is a total of $2^4 = 16$ different combinations where the four nodes are sending or not. Out of those 16 sending states, only the following seven are considered possible and they constitute the set S :

$$\begin{aligned} s_1 &= [1\ 0\ 0\ 1] & s_5 &= [0\ 1\ 0\ 0] \\ s_2 &= [0\ 1\ 0\ 1] & s_6 &= [0\ 0\ 0\ 1] \\ s_3 &= [0\ 0\ 1\ 0] & s_7 &= [0\ 0\ 0\ 0] \\ s_4 &= [1\ 0\ 0\ 0] \end{aligned} \quad (5.4)$$

Establishing the possible transitions

Having defined the set of possible sending states, S , we need to decide when a transition from sending state s_k to s_ℓ is allowed. This transition represents the fact that the network goes from state s_k to s_ℓ upon the end of a frame transmission. We apply a three-rule policy to discover the possible transitions.

First, at most one element of s_k changes its value from 1 in s_k to 0 in s_ℓ . This rule expresses the idea that no more than one node stops sending at the same time. The rule is derived from the observation that in practice it is highly unlikely for two nodes to independently end their transmissions at the exact same time. In our sample network, an example of a transition abiding by this rule is the transition from s_1 to s_2 in which node 1 ends its transmission and node 2 begins one. A negative example would be the transition from s_1 to s_3 , as it would require both nodes 1 and 4 to end their transmissions at the same time.

Second, at most one element of s_k changes its value from 0 in s_k to 1 in s_ℓ , unless in the case of a *synchronizing* node. Apart from this exception, this second rule is analogous to the first one and it implies that no more than one node starts sending at the same time. A node n is said to be synchronizing if *i*) it was sending in state s_k (i.e., $s_k(n) = 1$), and *ii*) it has at least two neighbors that have an *ON* activity but are not neighbors themselves. The exception allows to account for the situation where a central node's end of transmission triggers the beginnings of transmissions of several of its neighboring nodes. Said differently, in a case with two nodes, labeled m and p , and assuming that they have a common neighbor node n for which $s_k(n) = 1$ and $s_\ell(n) = 0$, we allow the transition from s_k to s_ℓ even if we have $s_k(m) = s_k(p) = 0$ and $s_\ell(m) = s_\ell(p) = 1$. In the case of our sample network, an example of this rule's application is the transition from s_4 to s_1 while a negative example is the transition from s_5 to s_1 . The exception to the rule applies twice, namely when we allow the transitions from s_3 to s_1 and from s_3 to s_2 .

Third, considering all potential activity states $\{A_k^t\}$ ($t = 1, \dots, T$) associated to s_k and all potential activity states $\{A_\ell^u\}$ ($u = 1, \dots, U$) associated to s_ℓ , there must be a combination (A_k^t, A_ℓ^u) in which at most one element changes its value from *ON* in A_k^t to *OFF* in A_ℓ^u or from *OFF* to *ON*. A clear negative example is the transition from $s_5 = [0\ 1\ 0\ 0]$ to $s_6 = [0\ 0\ 0\ 1]$ and vice versa. Nodes 2 and 4 can be simultaneously sending, so the only possible direct transition from s_5 to s_6 can happen if simultaneously node 2 switches *OFF* and node 4 switches *ON*. We consider that such changes have a negligibly small likelihood in practice and a zero probability in our modeling framework.

Overall, the rationale behind these three rules resorts to the fact that we consider as negligible the probability that two nodes undergo simultaneous changes in their behavior (be it their sending or activity). A consequence of the three rules is that self-transitions between the sending states are always possible, as no change is required in the sending state or in the activity.

By applying this three-rule policy, we discover all the possible transitions between the sending states, i.e., the transition matrix, in which a value of 1 in the k -th row and ℓ -th column indicates that the transition from s_k to s_ℓ is possible, while a value 0 indicated the transition is not possible. We represent below the 7×7 transition matrix obtained when applying the three-rule policy on the sample four-node network:

$$\begin{array}{c}
 s_1 \\
 s_2 \\
 s_3 \\
 s_4 \\
 s_5 \\
 s_6 \\
 s_7
 \end{array}
 \begin{array}{c}
 s_1 \\
 s_2 \\
 s_3 \\
 s_4 \\
 s_5 \\
 s_6 \\
 s_7
 \end{array}
 \begin{bmatrix}
 1 & 1 & 0 & 1 & 0 & 1 & 0 \\
 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 1 & 0 & 1 & 1 & 1 & 0 & 1 \\
 0 & 1 & 1 & 1 & 1 & 0 & 1 \\
 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
 0 & 0 & 1 & 1 & 1 & 1 & 1
 \end{bmatrix}. \tag{5.5}$$

Computing the transition probabilities

The next stage of our modeling approach is to find the probability of each transition. Let us consider the transition from sending state s_k to s_ℓ . First, we evaluate the

number of activity states associated to s_ℓ that are compatible with s_k . An activity state A_ℓ^u associated to s_ℓ is said to be *compatible* with s_k if there is at least one activity state associated to s_k that differs by at most one *ON* to *OFF* change (or vice versa) with A_ℓ^u . The reasoning behind this rule is similar in logic to the rules we derived before, i.e., we consider the probability that two nodes independently generate frames at the exact same time to be negligible. We denote by $A_{k\ell}$ the set of activity states associated to s_ℓ that are compatible with s_k . For example, in our sample network, there are four activity states associated to $s_1 = [1\ 0\ 0\ 1]$, namely:

$$\begin{aligned} A_1^1 &= [ON\ OFF\ OFF\ ON] \\ A_1^2 &= [ON\ ON\ OFF\ ON] \\ A_1^3 &= [ON\ OFF\ ON\ ON] \\ A_1^4 &= [ON\ ON\ ON\ ON] \end{aligned} \quad (5.6)$$

while those associated to $s_2 = [0\ 1\ 0\ 1]$ are:

$$\begin{aligned} A_2^1 &= [OFF\ ON\ OFF\ ON] \\ A_2^2 &= [ON\ ON\ OFF\ ON] \\ A_2^3 &= [OFF\ ON\ ON\ ON] \\ A_2^4 &= [ON\ ON\ ON\ ON] \end{aligned} \quad (5.7)$$

We observe that all the activity states associated to s_2 are compatible with s_1 , and thus belong to the set A_{12} , i.e., $A_{12} = \{A_2^1, A_2^2, A_2^3, A_2^4\}$.

Then, we simply derive the transition probability from s_k to s_ℓ as:

$$P_{k\ell} = \sum_{A_\ell^u \in A_{k\ell}} (P_{A_\ell^u} \prod_{s_\ell(n)=1} \frac{1}{1 + \sum_{m \in w_n} \mathbb{1}_{ON(m)}}), \quad (5.8)$$

where w_n represents the set of neighbor nodes of node n . The sum $\sum_{m \in w_n} \mathbb{1}_{ON(m)}$ returns an integer value equal to the number of neighbors of node n that are *ON* in the activity state A_ℓ^u . Equation (5.8) captures the idea that the transition probabilities depend both on the nodes' input rates and the underlying conflict graph. For each activity state A_ℓ^u that is compatible with the transition from the sending state s_k to s_ℓ , we calculate the associated transition probability by first assuming that all concerned nodes are equally likely to access the channel, and then, we weight the obtained value by the probability of occurrence of that activity state, $P_{A_\ell^u}$. The term $P_{A_\ell^u}$ in Eq. (5.8) ensures that the presence of a node with a high input rate will increase the probability that the whole network is in a sending state in which that node is sending. On the other hand, the other term within the sum represents the fact that a node that is well connected in the conflict graph experiences a lot of competition for channel access, which in turn diminishes its probability of gaining that access.

We now present two refinements to Eq. (5.8) that may occur, or not, depending on the network topology. This series of refinements allows us to finely tune the transition probabilities by incorporating specific network scenarios. We know that in these scenarios a node's topological position puts it in a (dis)advantageous position regarding its probability of gaining medium access.

Considering a given sending state s_k , the first modification regards nodes that are blocked or preempted by their neighbors. To cope with these nodes, we simply identify them and remove them from the considered neighbors of node n , i.e., $w(n)$, when using Eq. (5.8). A node is said to be *blocked* when it cannot start sending because it has two (or more) sending neighbors. For instance, node 3 is blocked when our sample network is in sending state $s_1 = [1\ 0\ 0\ 1]$. Indeed, it is very unlikely for node 3 to start sending, as this would require nodes 1 and 4 to simultaneously stop sending. Analogously, a node is said to be *preempted* if it has one neighbor that is sending, and at least one other neighbor that is *ON*, which itself has no sending neighbors. Such an example is node 3 when our sample network is in sending state $s_4 = [1\ 0\ 0\ 0]$ with an activity state $[ON\ OFF\ ON\ ON]$. Under these circumstances, node 4 does not have to wait for the end of node 1's sending in order to start sending, which in turn means node 3 is likely to be preempted. In our modeling approach, we simply account for these nodes by phasing them out from the actual neighborhood of node n when considering the sending state s_k . In other words, in Eq. (5.8), we replace the set of neighbor nodes of n , i.e., $w(n)$, by the restricted set of neighbor nodes which excludes blocked and preempted nodes.

The other refinement refers to the case when the z -th node of s_k is a synchronizing node that continues to have an *ON* activity in the next activity state A_ℓ^u associated to s_ℓ . We remind that a synchronizing node is a node sending in state s_k and having two neighbors that are *ON* and that are not neighbors themselves (a mode detailed definition is given in Section 5.2.1). First, for each activity state associated to s_ℓ , i.e., A_ℓ^u , we estimate the probability that the synchronizing node continues to send in s_ℓ using:

$$P_{z_\ell^u} = \frac{1}{1 + \sum_{m \in w_z} \mathbf{1}_{ON(m)}}, \quad (5.9)$$

where the sum $\sum \mathbf{1}_{ON(m)}$ returns an integer value equal to the number of neighbors of node z that are *ON* in activity state A_ℓ^u . For example, considering the sending state $s_3 = [0\ 0\ 1\ 0]$ of our sample network, node 3 is a synchronizing node. A possible activity state associated to s_3 is $[ON\ OFF\ ON\ ON]$ in which node 3 has two neighbors that are also *ON* (i.e., nodes 1 and 4). Using Eq. (5.9), we obtain: $P_{z_\ell^u} = \frac{1}{3}$. The second step consists in refining the computation of the transition probability $P_{k\ell}$ as follows:

$$P_{k\ell} = \sum_{A_\ell^u \in A_{k\ell}} (P_{A_\ell^u} \prod_{s_\ell(n)=1} \frac{1}{1 + \sum_{m \in w'_n} \mathbf{1}_{ON(m)}} (1 - \mathbf{1}_{z\ \text{sync}} \cdot P_{z_\ell^u})), \quad (5.10)$$

where $\mathbf{1}_{z\ \text{sync}} \cdot P_{z_\ell^u}$ returns the probability $P_{z_\ell^u}$ if a synchronizing node z exists in node n 's neighborhood, and 0 otherwise, and w'_n denotes the restricted neighborhood of node n (without blocked, preempted, and synchronizing nodes).

Finally, we ensure that the matrix is row-stochastic by dividing each $P_{k\ell}$ probability with the corresponding sum $\sum_{m=1}^N P_{km}$. Figure 5.2 show the complete chain for the four-node network with all sending states and some transition probabilities. Note that in this depiction, if the edge from state s_k to s_ℓ has no arrows, then both transitions $s_k \rightarrow s_\ell$ and $s_\ell \rightarrow s_k$ are possible.

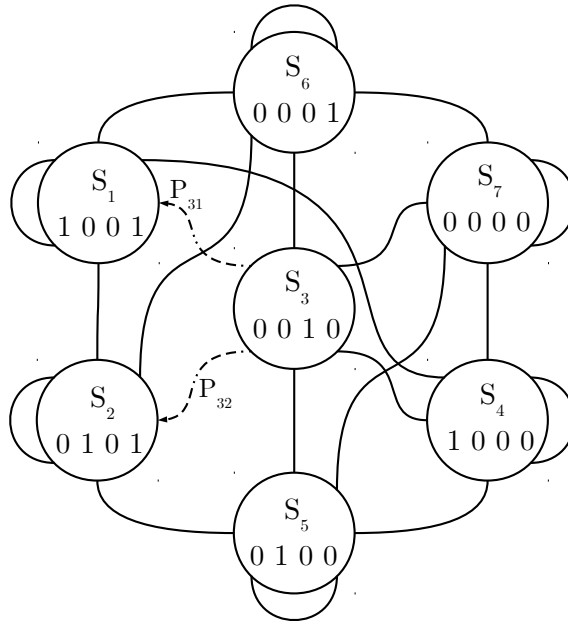


Figure 5.2: Markov chain of the four-node network in Fig. 5.1. Edges with no arrows represent transitions in both directions.

Deriving the output rates

Having evaluated all the transition probabilities, we can easily obtain the steady-state probability of each sending state s_k . Let us use the vector $\pi = [\pi_{s_1}, \pi_{s_2}, \dots]$ to denote the corresponding steady-state probabilities. We calculate the output rate of node i by summing the steady-state probabilities π_{s_k} over the set of sending state s_k in which node i is sending. Hence, we have:

$$y_n = \sum_{k; s_k(i)=1} \pi_{s_k}. \quad (5.11)$$

In our sample network, we notice that node 1's is sending only in sending states s_1 and s_4 . Thus, its output rate is given by: $y_1 = \pi_{s_1} + \pi_{s_4}$.

Having fully described the modeling approach, we provide its summary in an algorithmic form. We then test its accuracy in predicting the output rates in different network scenarios.

Algorithm

We use this section to show the algorithmic representation of the proposed modeling framework.

Algorithm 1 Model A

```

1: Sending States: Find all possible sending states:
2:  $S = \{0, 1\}^N$ 
3: for  $k = |S| \dots 1$  do
4:   if  $s_k$  contains two sending neighbor nodes then
5:     remove  $s_k$  from  $S$ 
6:   end if
7: end for
8: Activity States:
9: for  $s_k \in S$  do
10:   Find all the activity states associated to  $s_k$  as described in Section 5.2.1
11: end for
12: Blocked nodes:
13: For any node  $n$  in a network state  $s_k$  where  $s_k(n) = 0$ :
14: if  $\exists m, p \in \{w_n\}^2$  s.t.  $s_k(m) = s_k(p) = 1$  then
15:   node  $n$  is a blocked node in state  $s_k$ 
16: end if
17: Preempted nodes:
18: For any node  $n$  in a network state  $s_k$  and its activity state  $A_k^t$  where  $s_k(n) = 0$ :
19: if  $\exists m, p \in \{w_n\}^2, m \notin w_p$  s.t.  $s_k(m) = 1$  and  $A_k^t(p) = ON$  and  $\nexists r \in w_p$  s.t.
    $s_k(r) = 1$  then
20:   node  $n$  is a preempted node in state  $s_k$  for activity state  $A_k^t$ 
21: end if
22: Synchronized nodes:
23: For any node  $n$  in a network state  $s_k$  where  $s_k(n) = 1$  and an activity state
    $A_k^t$ :
24: if  $\exists m, p \in \{w_n\}^2, m \notin w_p$  s.t.  $A_k^t(m) = A_k^t(p) = ON$  then
25:   node  $n$  is a synchronizing node for nodes  $m$  and  $p$ 
26: end if
27: Transitions: Find the possible network state transitions
28: for  $k = 1 \dots |S|; \ell = 1 \dots |S|$  do
29:   if  $s_k \rightarrow s_\ell$  implies more than one  $1 \rightarrow 0$  change in the network state then
30:     transition  $Tr_{k\ell}$  is not possible
31:   else if  $s_k \rightarrow s_\ell$  implies more than one  $0 \rightarrow 1$  change in the network state
     except for having up to  $n$  such changes in the  $n$  nodes synchronized by the
     same node then
32:     transition  $Tr_{k\ell}$  is not possible
33:   else if there is no state  $A_\ell^u$  compatible with  $s_k$  then
34:     transition  $Tr_{k\ell}$  is not possible
35:   else
36:     transition  $Tr_{k\ell}$  is possible
37:   end if
38: end for

```

-
- 39: **Probabilities:** Calculate all transition probabilities
- 40: **for** $Tr_{k\ell} \in Tr$ **do**
- 41: Calculate activity state probabilities using (5.2)
- 42: If synchronizing nodes exist, calculate $P_{z_\ell^u}$ using (5.9)
- 43: Calculate the probability $P_{k\ell}$ of $Tr_{k\ell}$ with (5.10)
- 44: **end for**
- 45: **Normalize:**
 Normalize each transition probability $P_{k\ell}$ by dividing it with the sum of all transition probabilities of transitions leaving state s_k .
- 46: **Solve the chain:** Find the stationary distribution π of the Markov chain created by the states in S and the transitions in Tr
- 47: **Calculate output rates:** Use (5.11) to calculate the output rates of all N nodes.
-

Numerical Results

We begin this section with a study of our model's accuracy in predicting the output rates of a WLAN. We do so by comparing our model's prediction with the results obtained by simulation for several different networks. Then, we present two examples of possible applications in WLANs.

Simulation setup and performance metrics

All results presented in this section were obtained using the IEEE 802.11g standard in the ns2 discrete-event network simulator [77]. The nodes transmit frames with a fixed payload size $L = 500\text{B}$ over a link with $R = 54\text{Mbps}$ data rate. In our simulated networks, each node represents an AP that is sending downlink traffic to a single associated station, as discussed in Section 4.1. Although in practice an AP will have several associated stations most of the time, for the sake of simplicity we consider that the aggregated traffic sent to these stations can be averaged and modeled as a single downlink traffic. In the discussion at the end of this chapter, we present some of our simulation and real-life measurements that help us validate these hypotheses and show they are adapted to WLANs at a reasonable loss in accuracy.

With regards to the performance metrics, we study the nodes' output rates, y_n , and the *network utilization*, denoted by U and calculated as:

$$U = \frac{\sum_{n=1}^N y_n}{U_{max}}, \quad (5.12)$$

where U_{max} is the maximum network utilization. Otherwise stated, U_{max} is simply the maximum number of nodes that can be simultaneously sending, a quantity that clearly depends on the conflict graph. For example, in our sample four-node network of Fig. 5.1, at most two nodes can be sending simultaneously, i.e., nodes 1 and 4 or nodes 2 and 4. While the network utilization is very interesting from the provider's point of view, it can be a highly unfair performance metric from the

users' perspective as some users may obtain very low output rates while others can transmit almost continually. We use Jain's fairness index [78], J , to measure how fairly the throughput is divided among the nodes. Jain's index is a quantity in the interval $[0, 1]$, where 1 represents the highest fairness, meaning all nodes get an equal share. It is calculated as:

$$J = \frac{\left(\sum_{n=1}^N y_n \right)^2}{\sum_{n=1}^N y_n^2}. \quad (5.13)$$

Validation

We evaluate the accuracy of our model by comparing its values with those provided by the discrete-event network simulator ns2 [77]. Every simulation point presented in the results is the average of 20 independent simulation runs lasting 60 seconds each. The resulting confidence intervals are very narrow so we only use the midpoint in our validation. As for the input rates x_n , we represent their intensity in the simulator by alternating between active and inactive periods with exponentially distributed lengths. On average, the proportion of time spent in the active period is set to x_n . In active periods the node's queue is constantly backlogged, while in inactive period its queue is empty and the node is not attempting to access the medium. The throughput obtained in simulation is then normalized by the link speed, in order to obtain the dimensionless output rate, y_n , which is comparable with that of our model.

In our first example, we consider the four-node network of Fig. 5.1. We set the input rates of nodes 1, 3, and 4 to $x_1 = 0.5$, $x_3 = 1$, $x_4 = 0.5$, respectively. We let the input rate of node 2 vary from $x_2 = 0$ (constant *OFF*) to $x_2 = 1$ (constant *ON*), taking several values in that interval. We then evaluate the output rates of nodes 1 and 2, i.e., y_1 and y_2 , as well as the network utilization, as delivered by the simulator and by our model. Figure. 5.3 shows the corresponding results.

As expected, when the input rate of node 2 grows from 0 to 1, so does its output rate, though to a lesser extent. Because $y_2 < x_2$ (except for $x_2 = 0$), we conclude that node 2 cannot meet all its demands, regardless of their actual intensity. As for node 1, whose input rate x_1 is kept constant, its output rate tends to decrease with increasing values of x_2 . This behavior ensues from the competition between nodes 1 and 2. Note that, even for $x_2 = 0$, node 1 struggles to meet its demands as $y_1 = 0.4$ (while $x_1 = 0.5$). Finally, the network utilization grows steadily from around 0.6 to 0.75 as the input rate of node 2 increases. Overall, we observe that our proposed model is able to accurately capture these behaviors with a relative error typically less than 10%.

The second example involves a more complex network with a nine-node topology whose conflict graph is given in Fig. 5.4. In this example we consider several values for the input rate for node 6, x_6 , ranging from 0 to 1, while we keep the input rates of other nodes constant. The other input rates are set to $x_1 = 0.7$, $x_2 = 0.8$, $x_3 = 0.6$, $x_4 = 0.5$, $x_5 = 0.8$, $x_7 = 0.7$, $x_8 = 0.6$, and $x_9 = 0.9$. The corresponding results are shown in Fig. 5.5. In order to keep the figure legible, we choose to trace the output rates of nodes 5, 6, 7, and 9, bearing in mind that node 8's output rate

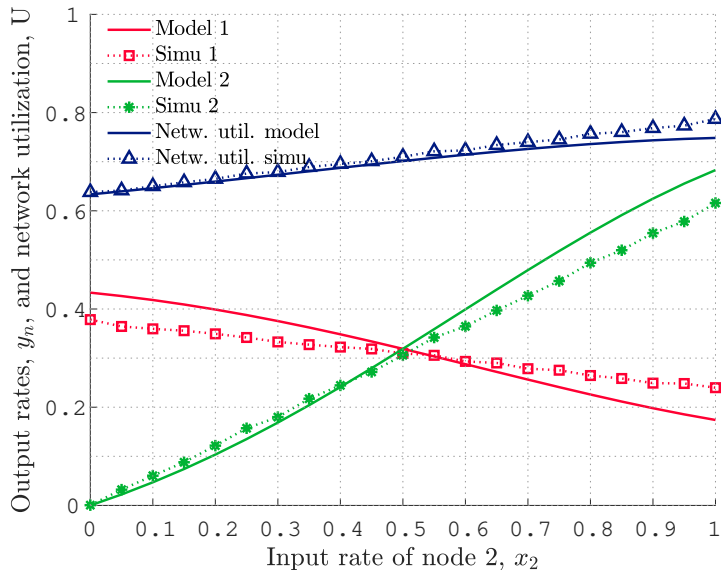


Figure 5.3: Four-node network of Fig. 5.1: varying the input rate of node 2.

evolves similarly to that of node 7, and that the first four nodes of the network are not highly influenced by the change in input rate of node 6. We observe that, as x_6 grows from 0 to 1, the nodes' behaviors can be significantly changed, with some increasing their output rates (e.g., nodes 6 and 9), while others experience a decrease (e.g., nodes 5, 7, and 8), that may result in a near node starvation as x_6 comes closer to saturation. In other words, node 6 allies with node 9 and together they manage to silence their neighbors by turning them into blocked or preempted nodes in many sending states. As shown by Fig. 5.5, our model was able to reproduce these non-trivial behaviors with a satisfying degree of precision.

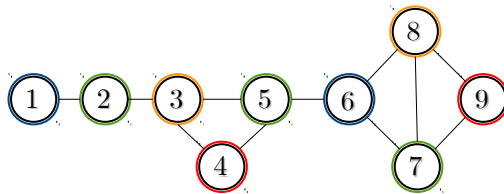


Figure 5.4: Conflict graph of a nine-node network.

In the aim of providing a more comprehensive idea of the accuracy of our model, Table 5.2 shows the overall error distribution obtained on over 250 samples. Each sample characterizes the precision of a node's output rate in the nine or four-node network. The median value indicates that, in 50% of the samples, the difference between y_n as returned by the simulator and by our model is less than 0.047. Table 5.2 also shows that, in all our samples, the difference between the model's and the simulator's y_n is never larger than 0.20.

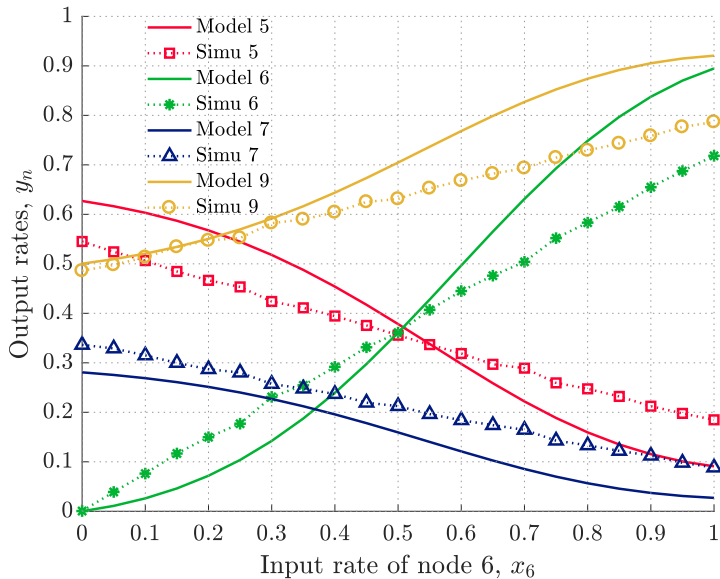


Figure 5.5: Nine-node network of Fig. 5.4: varying the input rate of node 6.

Median	<0.05	0.05-0.1	0.1-0.2	>0.2	Utilization median
0.047	52.7%	33.3%	14%	0%	0.053

Table 5.2: Distribution of the absolute errors for the output rates, y_n .

Note that in order to investigate the robustness of our approach, we explored several other examples with different network topologies, packet sizes, and means and distributions for the active and inactive periods. It was our experience that our model's accuracy was similar to that described by the two former examples.

Applications

Having validated the accuracy of our model, we now describe how its application can help to improve the general performance of a network by ensuring a better share of the channel resources. In the two scenarios that follow, we assume that all nodes are APs. Furthermore, we assume the presence of a network controller possessing the knowledge of the nodes' input rates and the network topology. We show how such a controller can be used to either restrain a node's input rate, or to completely turn off a node, in the goal of improving the network's performance.

In our first scenario, we suppose that the controller is able to regulate the input rate of a given node. Let us consider the nine-node network of Fig. 5.4 and that node 9 is selected. Note that although node 9 is in a remote position in the conflict graph, it plays a central role in the network topology as it is able to block or preempt nodes 7 and 8 with the help of node 6. Using our model, we discover the output rates of all nodes (including y_9) for values of x_9 ranging from 0.2 to 1, and we compute the associated value of Jain's index [78]. Figure 5.6 reports the corresponding results. It comes to no surprise that as we allow x_9 to increase,

nodes 7 and 8 undergo a significant decrease in their output rates, unlike node 6 that experiences a growth of its output rate. More interestingly, the network's Jain's index exhibits a peak value for x_9 close to 0.35. This peak is about 30% higher than the value obtained with $x_9 = 1$. In other words, to ensure a fair share of the channel resources among the nodes, the best scheme would be to restrain the input rate of node 9 to no more than $x_9 = 0.35$. As it is often the case, the trade-off when maximizing the fairness is a decreased network utilization. In this particular scenario, on average, 3 nodes are transmitting in the maximum fairness solution and 3.57 nodes are transmitting when $x_9 = 1$.

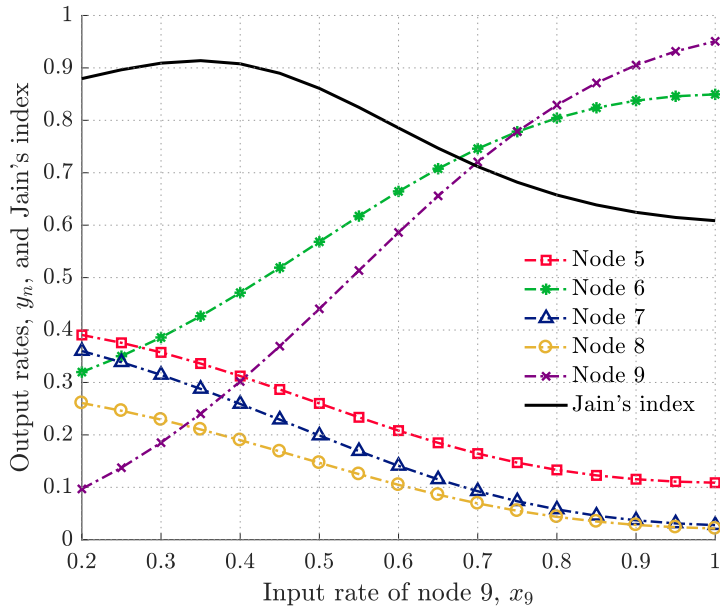


Figure 5.6: Jain's fairness index and output rates for the network of Fig. 5.4 as a function of node 9's input rate.

Our second scenario addresses the case in which the network controller can completely turn off a node (e.g., by moving it to another radio channel). Our goal is to show how our model can help determine which node should be turned off in order to maximize a given performance metric (Jain's index or the network utilization U). To that effect, we consider the ten-node network whose conflict graph is given in Fig. 5.7 in which the input rates have been set to $x_1 = 0.3$, $x_2 = 0.6$, $x_3 = 0.8$, $x_4 = 0.5$, $x_5 = 0.7$, $x_6 = 0.3$, $x_7 = 0.5$, $x_8 = 0.4$, $x_9 = 0.9$, and $x_{10} = 0.7$. One at a time, we turn off each node, and then we calculate the nodes' output rates, together with the Jain's index and the network utilization. The corresponding results are shown in Fig. 5.8. Note that Ni denotes the case where the i -th node is turned off, while Original refers to the original ten-node network without any nodes off. It appears that turning node 3 off would be the best option as not only does it allow to achieve the largest network utilization, but also maximizes the Jain's fairness index.

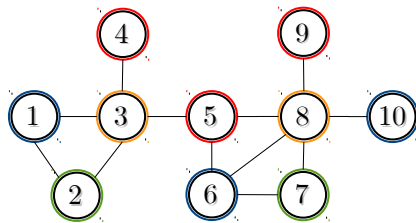


Figure 5.7: Conflict graph of a ten-node network.

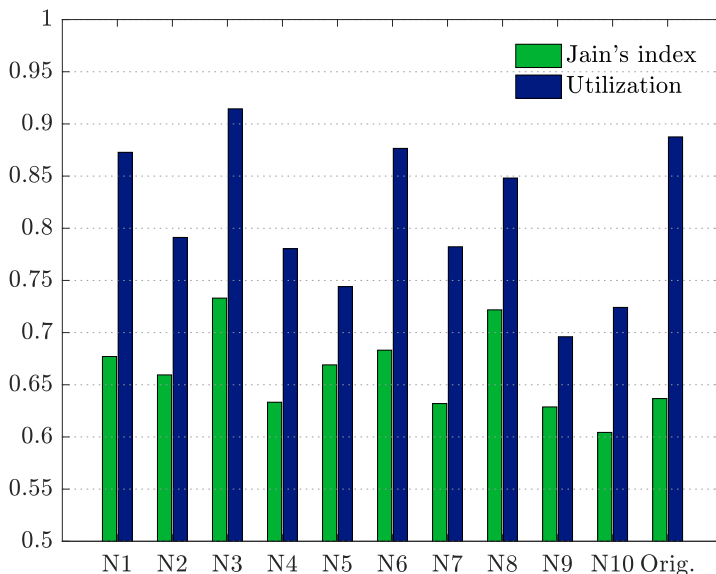


Figure 5.8: Jain's fairness index and network utilization for the network in Fig. 5.7.

Discussion

We use this final section to discuss some of the modeling and simulation choices we made throughout this work. We then conclude on several potential improvements we believe are crucial to the future development of the modeling framework.

Choosing the network simulator

Choosing the right validation tools for an analytical model is a task that requires careful consideration and (almost) always results in a trade-off. While we discussed the main aspects of the three performance evaluation approaches (modeling, simulation, experimentation) in Chapter 1, we wish to now briefly reflect upon the choices made regarding the numerical validation of the analytical model proposed in this chapter. As it is often the practice in WLAN modeling, we resort to using discrete-event simulators that are considered capable of rendering results close to reality at a low cost in both time and money. Given our previous experience and the need for simple simulation scenarios, we opted for the ns2 simulator [77]. Ns2



Figure 5.9: Conflict graph of a three-node chain network.

is an open source and adequately user-friendly network simulator that implements all the basic mechanisms of the IEEE 802.11 standard and the DCF procedure. While the system considered in this model can be fully simulated in ns2, it quickly became evident that extending it to more heterogeneous networks and more recent standard amendments is a requisite in order to keep the model relevant and up to date. However, ns2 maintenance has stopped a decade ago and all newer standard amendments and their features are not implemented in the simulator. An example is the simulation of links with different data rates which is not a straightforward task. We decided to test the more recent ns3 simulator, developed as the logical suite to ns2, though not backwards compatible. ns3 has more sophisticated PHY and MAC layers and it is actively kept up to date on new procedures and technologies, e.g., the 802.11ac standard amendment including frame aggregation, MCS indexes, and channel bonding.

Before completely switching to ns3, we wanted to compare the results provided by the two simulators when they are given the same network scenario. We chose the three-node chain network in Fig. 5.9 in which the input rates of nodes 2 and 3 are fixed to $x_2 = 1$ and $x_3 = 0.5$, and the input rate of node 1 is varying in the $[0,1]$ interval. Figure 5.10 shows the achieved throughput of node 1 as a function of the demanded throughput of node 1 for the two network simulators using the 802.11g standard amendment, payloads of 1000B, and a 54Mbps data rate. First, we reassuringly notice that both simulators provide similar results, especially in the mid-range input rates. However, on both extremities the maximum throughput the middle node achieves varies significantly. We searched through the source code of both simulators to find several possible explanations. The first one is the implementation of the capture effect, a phenomenon potentially happening when a node receives two frames in collision. In ns3, the capture effect allows a node to successfully receive a frame, even in collision, if the frame's signal is strong enough so that it can be correctly decoded. In ns2, a frame can be successfully received only if it was not in collision when the node began receiving the PHY header, i.e., if the frame is the first one of the two colliding frames to reach the node. Another issue relates to the implementation of 802.11 standard amendments. As 802.11g is not implemented in ns2, we had to modify the source code and manually set a list of parameters to simulate this amendment. Needless to say, the two implementations are not equivalent because simply modifying the DCF parameters of ns2 will not provide the exact same results as ns3. We also found issues in the physical layer and the backoff procedure, as the two simulators implement different default settings that we do not always know how to properly set and are out of the scope of this thesis.

In summary, we found that upgrading to the newer ns3 simulator allows us to obtain results similar to those of ns2 for simple networks, but to also simulate more elaborate scenarios closer to today's real-life implementations. The upgrade's trade-off lies in the simulator's complexity and its learning curve. While the fact

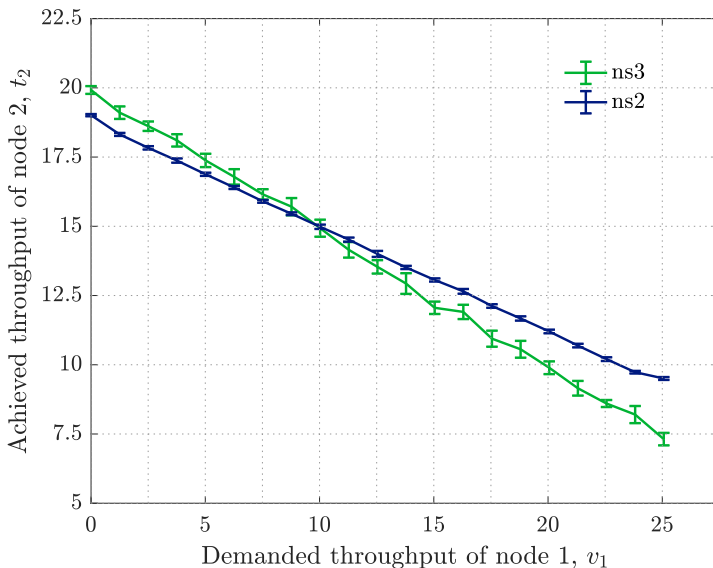


Figure 5.10: Comparing ns2 and ns3’s results for the network in Fig. 5.9.

that both simulators are coded in C++ eases the transition, ns3’s extensive set of features also leads to a much higher complexity in the source code and the simulation scripts, as well as a longer execution time. However, we believe that the benefits of using ns3 far outweigh the cost and we use it to obtain the rest of the results presented in this manuscript.

Testing the impact of uplink traffic

Our model is validated in a network where we use the strong assumption that all traffic is downlink, i.e., downloaded from the AP to the station. In order to assert the validity of this hypothesis, we measured the uplink/downlink ratio at three different locations: Lyon’s largest train station (Part Dieu), in the *grande école* where this work was developed (ENS Lyon), and a residential building. We used *tshark* [79] to sniff all the traffic transmitted on three different channels (1, 6, and 11) during 20 min intervals and then calculated the average over the three channels. The corresponding results reported in Table 5.3 showing that download data represent more than 90% of the traffic (in volume).

Location	Percentage of download traffic
Train station (Part Dieu)	90.97%
Graduate school (ENS Lyon)	93.79%
Residential building	92.10%

Table 5.3: Comparison of downlink and uplink traffic in real-life WLANs.

However, even if the upload traffic represents only about 10% of the total traffic, it does not by any means indicate that it can be neglected in our modeling

framework. To test if the user stations that generate some upload traffic can be reasonably omitted from the conflict graph, we run our ns3 simulations with no uplink traffic and compare it with simulations where uplink traffic represents 10% of the total network traffic. We simulated topologies of different sizes and complexities using several sets of input rates for every topology. We observe that, on average, having user stations generate 10% of the overall traffic changes the output rates of the APs by 5.5%. As a side note, this difference in output rate does not worsen the results of our model (in fact it improves them in most cases), as our model generally tends to slightly underestimate the AP's actual throughput (see network utilization in Fig. 5.3).

Contributions and limitations

At the time of its development, the largest benefit of Model A was the fact that it provided an accurate estimation of the nodes' achieved throughputs in a random-topology WLAN using only two input parameters, i.e., the input rates and the conflict graph. In hindsight though, the role of Model A is to be an important stepping stone for the two Markovian models that follow.

What was initially planned to be a simple Markovian model with a single chain, quickly became a difficult to unravel scheme of rules and exceptions. The reader surely took notice of our definitions of blocked, preempted, and synchronized nodes. However, Model A did serve as a good learning ground for understanding the complex system that are WLANs, and for understanding that such systems can still be accurately modeled with only several carefully selected parameters.

The single-chain solution of Model A can quickly become difficult to manage when the number of nodes increases. Even if larger Markov chains are easily solved with modern processing power, any single change in the input parameters requires the recalculation of the entire chain. Additionally, Model A imposes an important list of limitations on the network. The model can only be used when all nodes are homogeneous in data rates, payload sizes, and 802.11 standard amendments.

Every time we encountered an obstacle in the model or in the system description, we were forced to research further to find logical and feasible solutions. Model A helped us develop both the intuition and gather the knowledge needed for the model presented in the next chapter. This time, we opt for a highly-parallelized Divide-and-Conquer approach for heterogeneous networks.

Chapter 6

Model B: Larger and Heterogeneous WLANs

Introduction

Providing decision-making support has always been a crucial part of the modeling frameworks we develop. Whether in the design of a WLAN or in its management, having accurate and versatile analytical models can significantly lessen the workload of network architects and engineers. Today's WLANs, however, have experienced years of evolution and have come a long way since their first introduction over two decades ago. This evolution made us rethink and upgrade our single-chain Markovian model into a Divide-and-Conquer solution that accounts for larger and more heterogeneous networks.

The largest obstacle to overcome was the uniformity of the AP's traffic parameters. This chapter presents a Markovian model capable of handling WLANs with different data rates and frame sizes for all the APs. The enhancement allows us to also consider current standard features, such as frame aggregation. Moreover, we take into account the impact of the neighbors' traffic parameters on the maximum achievable throughput, a feature that has proven to be essential in accurately estimating the nodes' throughputs in some network configurations.

The growth of the Markov chain's size with the number of nodes is also an issue that highly impacts the model's complexity. Having a single chain allows us almost no space for parallelization (or partial solution) of Model A in case of a slight modification in the WLAN parameters. We now propose a Divide-and-Conquer approach that relies on a series of smaller and completely mutually independent Markov chains, opening up a new set of parallelization possibilities.

Finally, we wanted to improve the validation process and bring the simulation results a little closer to real-life WLANs. In order to do so, we traded the ns2 simulator for the more recent ns3 simulator, allowing us to assess the model's accuracy using a more detailed and precise depiction of WLANs. In this chapter, we present all the specificities of our Model B using a sample network, we then present the validation results and several possible applications for real-life WLANs, before finally empirically studying its complexity and comparing it to that of Model A.

Modeling Approach and Algorithm

Our Divide-and-Conquer Markovian approach relies on a series of Markov chains that each model a specific network configuration which we refer to as *subnetworks*. Figure 6.1 shows a schematic representation of the entire solution. We begin this section by first describing how we obtain these subnetworks in Stage 1. We then detail, as we did before, the process of creating and solving each Markov chain in Stages 2 and 3. Finally, in Stage 4 we combine their solutions and thus obtain the nodes' output rates and achieved throughputs. Figure 6.1 distinctly shows the parallelization capacity of the Divide-and-Conquer approach, as all the Markov chains belonging to different subnetworks can be solved individually of one another, as opposed to the single Markov chain of Model A shown in Fig. 5.2 of Chapter 5.

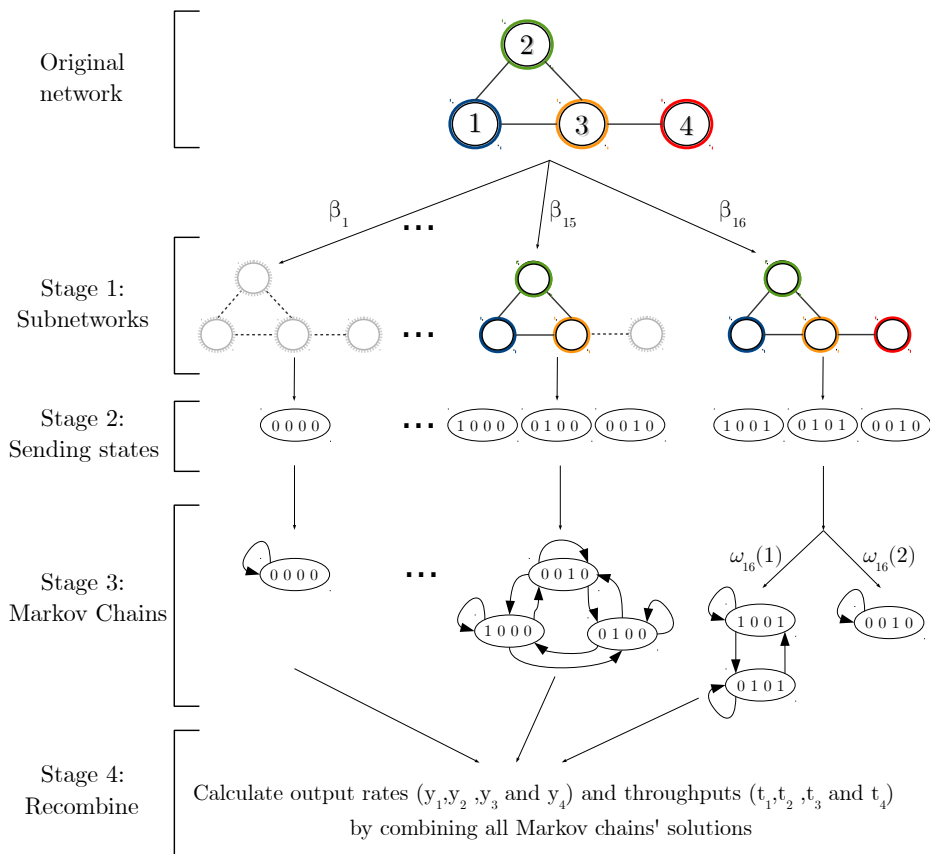


Figure 6.1: Schematic representation of the proposed solution.

Table 4.1 of Section 4.1 and Table 6.1 summarize the system and model notation, respectively.

Parameter	Description
α	backoff factor
B	set of possible subnetworks, $B = \{1, 0\}^N$
b_i	i th subnetwork, $b_i \in B$
$b_i(n)$	regime of the n th node in subnetwork b_i , $b_i(n) \in \{ON, OFF\}$
β_i	occurrence probability of the i th subnetwork
S	set of possible sending states, $S \subseteq \{0, 1\}^N$
s_k	k th sending state, $s_k \in S$
$s_k(n)$	state of the n th node in sending state s_k , $s_k(n) \in \{1, 0\}$
S_i	set of sending states associated to subnetwork b_i , $S_i \subseteq S$
c_i^m	m th irreducible Markov chain of subnetwork b_i
S_i^m	set of sending states associated to c_i^m , $S_i^m \subseteq S_i$
$\sigma_i(k)$	probability that sending state s_k of subnetwork b_i is initially chosen
$P_{k,\ell}$	probability of the transition from sending state s_k to s_ℓ
w'_n	restricted set of neighbors of node n with blocked nodes removed
M_i	number of irreducible Markov chains for the subnetwork b_i
π_i^m	steady-state probability distribution of c_i^m
π_i	steady-state probability distribution of subnetwork b_i
ω_i^m	occurrence probability of c_i^m
D_i	set of dominant chains in subnetwork b_i
d_i	set of dominated chains in subnetwork b_i
Q_n	set of cliques that contain node n
q_j	j th clique

Table 6.1: Modeling notation.

The backoff factor α

Before detailing the process of building the Markov chains, we first study how the nodes' different data rates can impact the resource sharing. In this section, we focus on the Flow in the Middle (FIM) topology shown in Fig. 6.2 and discussed in some detail in Chapter 1.



Figure 6.2: Conflict graph of a three-node FIM network.

Chaudet et al. [20] and Ducourthial et al. [41] study the impact of a frame's transmission duration in the FIM topology of Fig. 6.2 and in larger chain networks, respectively. They find that when transmissions are kept short the duration of the backoff becomes comparable to the duration of the frame transmission. As a result, it is more statistically likely that both nodes 1 and 3 of the FIM network are simultaneously in backoff, leaving the channel idle for a potential transmission of the starving node 2. It follows that shorter transmissions can increase the fairness of resource sharing. However, when transmissions times are long, node 2 almost continually senses the medium busy, as the edge nodes rarely simultaneously perform backoff.

We introduce the *backoff factor* α as a means to quantify the impact of the

transmission duration on fairness. The backoff factor is the ratio between the average backoff period duration and the duration of the entire transmission:

$$\alpha = \frac{T_{\text{backoff}}}{T_{\text{DIFS}} + T_{\text{PHY}} + T_{\text{FRAME}} + T_{\text{SIFS}} + T_{\text{PHY}} + T_{\text{ACK}}} . \quad (6.1)$$

Having defined α , we choose the FIM topology and the IEEE 802.11g standard amendment to test how it impacts resource sharing. We begin by varying α and tracing the evolution of the middle node's output rate, y_2 . In Fig. 6.3 we show this evolution for 11 different α values in the $[0.03, 0.50]$ interval obtained by changing the data rate and/or frame size. Next, we perform a quadratic fit (using the least squares method) of node 2's output rate as a function of α and discovered it is closely matched by the function:

$$y_2(\alpha) = -0.66\alpha^2 + 0.88\alpha + 0.01 . \quad (6.2)$$

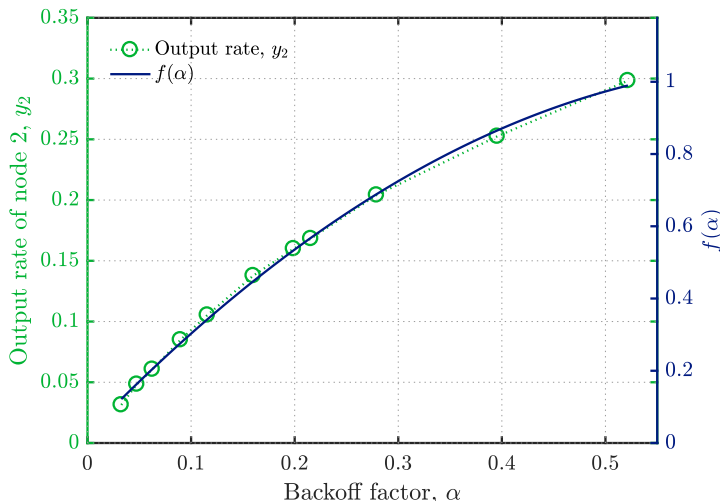


Figure 6.3: Influence of the backoff factor α on node 2's output rate in the FIM network of Fig. 6.2.

For the purpose of our modeling framework, we normalized the $y_2(\alpha)$ function so that we obtain an $f(\alpha)$ such that $0 \leq f(\alpha) \leq 1$, as shown by the right-hand side y-axis of Fig. 6.3. Thus, the best-case scenario for the middle node is when $\alpha = 0.5$, i.e., the larger α , the more the nodes experience a fair sharing of the medium. In fact, it is our experience that the value of α tends to significantly affect the performance of many other network topologies beyond the FIM example. Although we are fully aware that the exact quadratic function of α varies from one network to another, we believe that it will be difficult (if not impossible) to discover a general expression for α that applies equally to all networks. For this reason, we choose to extrapolate the knowledge we obtain on the FIM topology and reuse it on other networks. We detail this process at the end of our modeling approach description.

Modeling Approach

For the sake of clarity, we once again resort to the sample network depicted in Fig. 6.4 to show the step by step execution of the proposed modeling approach.

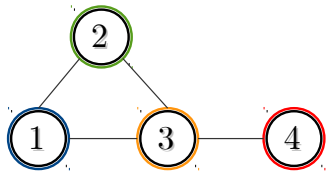


Figure 6.4: Conflict graph of a four-node network.

Decomposing into subnetworks

Analogously to the previous modeling framework, we consider that the nodes of the WLAN alternate their activity between *ON* and *OFF* periods. We recall that in the *ON* regime, a given node n has at least one frame to send, and thus has a non-empty buffer, while in the *OFF* regime, a node's buffer is empty. We consider that the nodes' regimes, and consequently their input rates, are independent of each other. At any time, the state of the network activity can be described by a vector of length N , where N is the number of nodes in the network and the n th element expresses the current regime of node n (be it *ON* or *OFF*). Thus, for a network with N nodes, there are 2^N such vectors that correspond to all the possible combinations of the two regimes over the N nodes.

In this chapter, we apply a Divide-and-Conquer approach by choosing to analyze the WLAN not as a single complex network in which any node can alternate between *ON* and *OFF*, but rather as a collection of 2^N simpler networks in which every node is permanently *ON* or *OFF*. We refer to these new networks as the *subnetworks* and we denote them by b_1, b_2, \dots, b_{2^N} . Hence $b_i(n)$ indicates the regime of node n in subnetwork b_i . We use B to designate the set that contains all subnetworks.

For the sample network of Fig. 6.4, as well as for any other four-node network, there is a total of $|B| = 16$ such subnetworks:

$$B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{16} \end{bmatrix} = \begin{bmatrix} OFF & OFF & OFF & OFF \\ OFF & OFF & OFF & ON \\ \vdots & \vdots & \vdots & \vdots \\ ON & ON & ON & ON \end{bmatrix}. \quad (6.3)$$

We refer to the probability that the current state of the network is subnetwork b_i as the *occurrence probability* of b_i and we denote it by β_i ($i = 1, \dots, 2^N$). Note that a subnetwork's occurrence probability depends only on the nodes' input rates and can be calculated as:

$$\beta_i = \prod_{n|b_i(n)=ON} x_n \prod_{m|b_i(m)=OFF} (1 - x_m). \quad (6.4)$$

For example, in our four-node network, one of the possible subnetworks is $b_{14} = [ON ON OFF ON]$. This subnetwork represents the case when nodes 1, 2, and 4 are in transmission or have a frame waiting to be sent, while node 3 has an empty buffer. Its occurrence probability is calculated as:

$$\beta_{14} = x_1 x_2 (1 - x_3) x_4 . \quad (6.5)$$

Next, we show how to solve each of the subnetworks separately and independently of the rest of the subnetworks.

Solving each subnetwork as one or more Markov chain(s)

We now detail how we analyze the behavior of each subnetwork using Markov chains. We start by defining the possible states and transitions of the corresponding Markov chains. Note that, in this subsection, the subject of study is any of the subnetworks b_i ($i = 1, \dots, 2^N$) resulting from the network decomposition (see Section 6.2.3).

Defining the possible states for the subnetwork

While the regime (*ON* or *OFF*) of each node is set and fixed (for the considered subnetwork b_i), knowing the regime is not sufficient to determine if a node is currently sending a frame or not. As per our definition, an *ON* node can be either transmitting or waiting for the medium to become idle. We eliminate this ambiguity by reusing the notion of *sending states*.

Like a subnetwork, a sending state is a vector of length N whose n th element refers to the activity of the n th node. However, unlike a subnetwork, a sending state indicates for each node n if the node is transmitting (marked 1) or not (marked 0). Let s_k denote the k th sending state (with $k = 1, \dots$). Thus, if node n is currently transmitting we have $s_k(n) = 1$, and $s_k(n) = 0$ otherwise, for $n = 1, \dots, N$. Let S denote the set of all possible sending states over all existing subnetworks. For the same of completeness, we remind that each possible sending state must comply with a common property of CSMA/CA: neighboring nodes cannot transmit successfully at the same time, i.e., Contrarily to Model A, we now use S_i to designate the set of possible sending states associated to the subnetwork b_i . Note that we can easily determine S_i since S_i is a subset of S whose elements satisfy the following properties:

1. if $b_i(n) = ON$ and n has no transmitting neighbors, then $s_k(n) = 1$;
2. if $b_i(n) = OFF$, then $s_k(n) = 0$.

Note that the rationale behind the second property is quite straightforward: a node that has no frames to be sent cannot be sending. The first property is derived from a phenomenon studied in [37]: CSMA/CA networks tend to increase the spacial reutilization of the medium by maximizing the number of simultaneous transmissions. As a result, in our model, we enforce any node that is *ON* and senses an idle medium to be in transmission.

In the case of our sample network, the subnetwork $b_{16} = [ON ON ON ON]$ has three possible sending states, $s_1 = [1 0 0 1]$, $s_2 = [0 1 0 1]$, and $s_3 = [0 0 1 0]$.

Note that other sending states may exist but we consider them to be negligible in b_{16} , however, some of them will appear in other subnetworks. For example, the sending state $[1\ 1\ 0\ 1]$ breaks the CSMA/CA condition, as nodes 1 and 2 are neighbors and cannot be simultaneously transmitting, i.e., the state is neither a part of S nor S_{16} . The sending state $[1\ 0\ 0\ 0]$ is deemed not possible since node 4 breaks the first condition. Indeed, b_{16} indicates that node 4 is *ON*, and because it has no sending neighbors, it should be sending its frames. However, should the current subnetwork be $[ON\ ON\ ON\ OFF]$, then $[1\ 0\ 0\ 0]$ is possible. This step of determining the sending states is illustrated by Stage 2 of Fig. 6.1.

Determining the possible transitions

The set of sending states found for the subnetwork b_i , denoted S_i , will serve as the Markov chain's set of states. We now detail how we decide which transitions are possible between those sending states. As it was before, our reasoning is based on the idea that the probability of two nodes starting (or ending) their transmission at the exact same time, in a CSMA/CA network, is negligibly small. However, this time we significantly simplify our transition rules by removing the conditions of blocked, preempted, and synchronizing nodes. Let s_k and s_ℓ be two possible sending states of S_i . The transition from sending state s_k to s_ℓ is deemed possible if and only if s_k and s_ℓ both verify that:

1. exactly one node alters from 1 in s_k to 0 in s_ℓ , and
2. exactly one node alters from 0 in s_k to 1 in s_ℓ .

Note that a self-transition on a given sending state s_k is always possible, as it implies no changes in the sending state.

For example, in our four-node network it is possible to go from sending state $[1\ 0\ 0\ 1]$ to $[0\ 1\ 0\ 1]$, as in this transition node 1 ends and node 2 starts a transmission. However, it is not possible to go from network state $[1\ 0\ 0\ 1]$ to $[0\ 0\ 1\ 0]$, as it implies both nodes 1 and 4 ending their transmissions at the exact same time. Figure 6.5 shows the existing transitions in our modeling framework between the possible sending states associated to the subnetwork b_{16} .

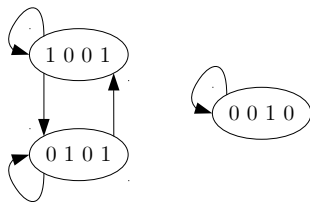


Figure 6.5: Possible sending states and corresponding existing transitions associated to the subnetwork $b_{16} = [ON\ ON\ ON\ ON]$.

Calculating the transition probabilities

We now explain how we determine the probability of the transitions between the possible sending states s_k composing our Markov chain. Note that impossible

transitions have zero probability. To evaluate the non-zero transition probabilities, we need to recall our definition of a *blocked node*. A node having at least one of its neighbors currently transmitting is said to be blocked as it is unable to start a collision-free transmission. For example, in the four-node network, node 3 can be blocked by the transmissions of any of the other three nodes.

We can now calculate $P_{k,\ell}$, the probability of the transition from sending state s_k to s_ℓ , as:

$$P_{k,\ell} = C \prod_{n|s_\ell(n)=1} \frac{1}{1 + \sum_{m \in w'_n} \mathbb{1}_{b_i(m)=ON}}, \quad (6.6)$$

where C is a normalizing constant such that $\sum_{\ell \geq 1} P_{k,\ell} = 1$, and w'_n defined as $w'_n = \{m \in w_n \setminus \{n\} \mid m \text{ is not blocked in } s_\ell \text{ by a node } \in w_m / \{n\}\}$ is the restricted neighborhood of node n , i.e., w'_n contains all neighbors of n that are not blocked by some node different from node n . As an example, in the subnetwork $b_{16} = [ON \ ON \ ON \ ON]$ and the sending state $[1 \ 0 \ 0 \ 1]$, the restricted neighborhood of node 1 contains only node 2, as node 3 is blocked by node 4. Note that the indicator function $\mathbb{1}_{b_i(m)=ON}$ returns 1 if $b_i(m) = ON$, and 0 otherwise.

The underlying logic behind Eq. (6.6) is that all nodes that are *ON* (whether they are sending or not) compete with their neighbors for accessing the medium. We also consider them equally likely to gain the medium access. On the other hand, nodes that are *OFF* do not affect the transition probability because they do not compete for medium access.

For instance, when node 3 of the subnetwork b_{16} in Fig. 6.1 competes with nodes 1, 2, and 4, it has a $\frac{1}{4}$ chance of gaining the medium. However, in this same scenario node 4 competes with only one neighbor, so its chance of gaining access would be $\frac{1}{2}$.

Calculating the steady-state probabilities

At this stage, the Markov chain associated to subnetwork b_i is fully characterized and we can calculate its steady-state probabilities. Let us remind that a Markov chain is *irreducible* if from any of its states there is a way to reach any other state. Depending on the setting of the subnetwork under study, the corresponding Markov chain may or may not be irreducible. Should the Markov chain not be irreducible, we consider each irreducible Markov chain separately. We denote by M_i the number of irreducible Markov chains in subnetwork b_i . For example, as shown by Fig. 6.5, the subnetwork $b_{16} = [ON \ ON \ ON \ ON]$ contains two irreducible chains, i.e., $M_{16} = 2$ (since it is not possible to go from the sending state $[0 \ 0 \ 1 \ 0]$ to the other two sending states). We use c_i^m , $m \in [1, \dots, M_i]$, to denote the m th irreducible chain of subnetwork b_i . Hence the left-hand chain of Fig. 6.5 is denoted by c_{16}^1 and the right-hand chain is c_{16}^2 .

We compute the steady-state probabilities of each irreducible chain c_i^m for the subnetwork b_i and we denote by π_i^m the vector containing the corresponding values. Note that we use S_i^m to refer to the set of sending states in chain c_i^m (while, as defined previously, S_i denotes the set of possible sending states associated to b_i). If the subnetwork has a single irreducible Markov chain ($M_i = 1$), then it follows that $S_i^1 = S_i$. Thus, the steady-state probabilities of the subnetwork's sending states are equivalent to those of the Markov chain c_i^1 and we have $\pi_i = \pi_i^1$, where π_i is

the vector containing the steady-state probabilities of subnetwork b_i . In this case, we can skip ahead to Section 6.2.5.

Combining several irreducible Markov Chains

For subnetworks that contain more than one irreducible Markov chain, we need to combine the steady-state probabilities found for each Markov chain into the steady-state probabilities for the whole subnetwork b_i . To do so, our approach consists in evaluating the odds of entering each irreducible chain. Let us denote by $\sigma_i(s_k)$ the probability that the sending state s_k of subnetwork b_i is initially chosen.

Clearly, we must have: $\sum_{k=1}^{|S_i|} \sigma_i(s_k) = 1$. Our way to evaluate $\sigma_i(s_k)$ is to consider all possible direct paths of reaching s_k starting from the empty sending state where no nodes are transmitting, namely $[0\ 0\ \dots\ 0]$, and then to sum their probabilities. For example, in the case of the subnetwork b_{16} , we can identify two paths leading to $s_1 = [1\ 0\ 0\ 1]$ from $[0\ 0\ 0\ 0]$, namely (a) node 1 starts transmitting, followed by node 4, and (b) vice versa. For path (a), the probability that node 1 gains medium access is $\frac{1}{4}$, since a total of 4 nodes are competing for the access. Once node 1 starts its transmission, its neighbors nodes 2 and 3 are blocked. This leaves node 4 alone to compete for the medium meaning that node 4 gains access to the medium with a probability of 1. Thus, the overall probability of path (a) is $\frac{1}{4}$. Turning to path (b), the probability that node 4 is the first to gain access to the medium is $\frac{1}{4}$. Once node 4 transmits, node 3 becomes blocked, while nodes 1 and 2 are still competing. Node 1's chance of transmitting can then be approximated to $\frac{1}{2}$. Therefore the overall probability of path (b) is $\frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8}$. It follows that $\sigma_{16}(1\ 0\ 0\ 1) = \frac{1}{4} + \frac{1}{8} = \frac{3}{8}$. With the same reasoning, we obtain $\sigma_{16}(0\ 1\ 0\ 1) = \frac{3}{8}$ and $\sigma_{16}(0\ 0\ 1\ 0) = \frac{1}{4}$.

Having calculated the probabilities of entering each sending state of b_i , we now introduce a *weighting factor*, denoted by ω_i^m , to express the probability that this particular irreducible Markov chain c_i^m is initially chosen. Keeping in mind that S_i^m denotes the set of sending states in the m th irreducible Markov chain of b_i , we compute ω_i^m as follows:

$$\omega_i^m = \sum_{k|s_k \in S_i^m} \sigma_i(k). \quad (6.7)$$

In Fig. 6.6 we show the entry probabilities as well as the weighting factors for the subnetwork $b_{16} = [ON\ ON\ ON\ ON]$ of our sample network of Fig. 6.4. The values found for the weighting factors are $\omega_{16}^1 = \sigma_{16}(1\ 0\ 0\ 1) + \sigma_{16}(0\ 1\ 0\ 1) = \frac{3}{4}$ and $\omega_{16}^2 = \sigma_{16}(0\ 0\ 1\ 0) = \frac{1}{4}$.

Finally, to calculate π_i (steady-state probabilities of the subnetwork b_i) from π_i^m 's (steady-state probabilities found for each of the M_i irreducible Markov chains associated with b_i) we simply proceed as follows:

$$\pi_i = [\pi_i^1 \times \omega_i^1, \dots, \pi_i^{M_i} \times \omega_i^{M_i}]. \quad (6.8)$$

In other words, π_i is obtained as a weighted sum of π_i^m 's.

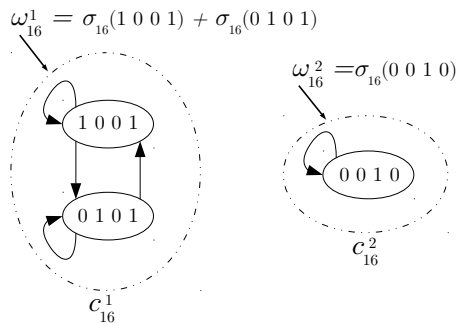


Figure 6.6: Probabilities of entering each sending state, i.e., $\sigma_i(k)$, and the corresponding weighting factors, i.e., ω_i^m for the subnetwork $b_{16} = [ON ON ON ON]$.

Adjusting to the IEEE 802.11 parameters

In this section, we refine the computation of the weighting factors ω_i^m to incorporate the value of the backoff factor α (see Eq. (6.1) in Section 6.2.1), which accounts for the mean length of the frames sent over the network, the data rates of the used wireless channels, as well as the particular amendment of IEEE 802.11 in use. In networks where nodes have different backoff factors, for example when not all nodes use the same data rate, we simply calculate the average α over all nodes. We have shown in Section 6.2.1, in the case of an FIM network topology, α correlates highly with the severity of the starvation experienced by the node in the middle (also shown in [20] and [41]).

We now detail how we incorporate the value found for α into our model by adjusting the weighting factors ω_i^m of the irreducible Markov chains c_i^m . To begin with, we introduce the notion of *dominant* and *dominated* Markov chains. The dominant Markov chains of a given subnetwork b_i are those containing the highest number of sending nodes. Conversely, irreducible Markov chains of b_i with a lower number of sending nodes than the dominant chain(s) are called dominated chains. Clearly, should a subnetwork have a single irreducible Markov chain, the discussion on dominant and dominated chains is of no relevance. We denote by D_i (resp. d_i) the set of all dominant (resp. dominated) Markov chains for the subnetwork b_i , while $|D_i| \geq 1$ (resp. $|d_i| \geq 0$) refers to the number of dominant (resp. dominated) Markov chains. In our example shown for the subnetwork b_{16} (see Fig. 6.6), the Markov chain on the left is a dominant chain while the Markov chain on the right is a dominated chain so that we have $D_{16} = \{c_{16}^1\}$, $|D_{16}| = 1$, $d_{16} = \{c_{16}^2\}$ and $|d_{16}| = 1$. To capture the increasing fairness between nodes for growing values of α , we modify the values of the weighting factor of all chains as follows:

$$\tilde{\omega}_i^m = \begin{cases} \omega_i^m \times f(\alpha), & \text{if } c_i^m \in d_i, \\ \frac{1-\Omega_i}{|D_i|}, & \text{if } c_i^m \in D_i, \end{cases} \quad (6.9)$$

where Ω_i the sum of the modified weighting factors of all the dominated irreducible Markov chains in b_i (i.e., $\Omega_i = \sum_{m|c_i^m \in d_i} \tilde{\omega}_i^m$). It is clear that if $M_i = 1$ then

$\omega_i^1 = \tilde{\omega}_i^1 = 1$, as the subnetwork has a single irreducible Markov chain and it will always be initialized in that chain.

Let us assume that the network of Fig. 6.6 uses the IEEE 802.11g standard amendment with frames of size $L = 1064$ bytes of which $H = 64$ bytes are headers, and a data rate of $R = 54$ Mbps. With these values we obtain an $\alpha = 0.268$ from Eq. (6.1) and the chains' weighting factors are adjusted as follows: $\tilde{\omega}_{16}^2 = \omega_{16}^2 \times f(\alpha) = 0.175$ and $\tilde{\omega}_{16}^1 = 1 - \omega_{16}^2 \times f(\alpha) = 0.825$.

Combining subnetwork solutions

So far we have divided the network into subnetworks and solved each one of them separately. The last phase of the model consists in combining the results obtained for different subnetworks and calculating the nodes' output rates. A node's output rate represents the portion of time when the node is occupying the medium, including the frame transmission itself and all the necessary DCF overhead. Thus, we calculate node n 's output rate, y_n , as:

$$y_n = \sum_{i=1}^{|B|} \left\{ \mathbb{1}_{b_i(n)=ON} \times \beta_i \times \sum_{m=1}^{M_i} \left(\tilde{\omega}_i^m \times \sum_{k|s_k \in S_i^m} \left(\mathbb{1}_{s_k(n)=1} \times \pi_i^m(k) \right) \right) \right\}, \quad (6.10)$$

Equation (6.10) gives the sum of the stationary probabilities of all the sending states in which node n is sending, times the occurrence probabilities of all the irreducible Markov chains in which those states appear, times the occurrence probabilities of the subnetwork to which those chains belong. Otherwise stated, it is simply the sum product of the probabilities of all the *subnetwork* \times *Markov chain* \times *sending state* combinations in which node n is sending. The normalization of the steady-state probabilities is ensured by the β_i and $\tilde{\omega}_i^m$ terms.

We can now transform node n 's output rate into its obtained throughput, t_n . When all the network nodes use the same standard amendment, data rate, and mean payload length, then all nodes have an equal maximum achievable throughput, $t_{n,max}$, and we can calculate the throughput of node n as:

$$t_n = y_n \times t_{n,max}. \quad (6.11)$$

However, when these parameters vary per node, nodes will have different maximum achievable throughputs, $t_{n,max}$, resulting in different node throughputs, t_n . First, we calculate the maximum achievable throughput of every node using Eq. (2.4) of Chapter 2. Next, we consider all the maximal cliques of the network and estimate their global clique throughput. We denote by q_j ($j \geq 1$), the j th maximal clique of the network, and by Q_n ($n \geq 1$) the set of maximal cliques that contain the node n . For example, our four-node network in Fig. 6.4 contains two maximal cliques, the first one containing nodes 1, 2, and 3, denoted q_1 , and the second containing nodes 3 and 4, denoted q_2 . Node 3 is the only node that belongs to both cliques, thus $Q_3 = \{q_1, q_2\}$. We calculate the global throughput of clique q_j , t_{q_j} as:

$$t_{q_j} = \frac{\sum_{m \in q_j} y_m \times L_m}{\sum_{m \in q_j} y_m \times \frac{L_m}{t_{m,max}}}, \quad (6.12)$$

where L_m denotes the mean payload length of node m .

The reasoning behind Eq. (6.12) is that every node m that belongs to the clique q_j will gain a portion of the medium access that is proportional to its output rate, y_m . If node m belongs to several cliques, we calculate the average throughput over all those cliques. Finally, node m 's throughput is calculated as the product of its output rate and its average clique throughput:

$$t_n = y_n \times \frac{\sum_{j|q_j \in Q_n} t_{q_j}}{|Q_n|} . \quad (6.13)$$

In the following section we provide a summarized version of the proposed model in an algorithmic form.

Algorithm

We use this section to show the algorithmic representation of the proposed modeling framework.

Algorithm 2 Model B

- 1: **Decomposing into subnetworks:**
 - 2: Find the set of all possible subnetworks B as given in Eq. (6.3)
 - 3: Calculate the occurrence probability β_i of each subnetwork b_i of B
 - 4: using Eq. (6.4)
 - 5: **for all** $b_i \in B$ **do**
 - 6: **States:**
 - 7: Create the set S_i , $S_i \subseteq S$, containing all the possible sending states
 - 8: associated to the subnetwork b_i using the method of Section 6.2.4
 - 9: **Transitions:**
 - 10: Determine the possible transitions between the sending states
 - 11: with the method described in Section 6.2.4
 - 12: **Transition probabilities:**
 - 13: Calculate the probability of each transition $P_{k,\ell}$:
 - 14: (i) $P_{k,\ell} = 0$ if the transition is not possible
 - 15: (ii) $P_{k,\ell}$ is calculated with Eq. (6.6), otherwise
 - 16: **Building the chain:**
 - 17: Create the Markov chain of subnetwork b_i with the states of step 6,
 - 18: the transitions of step 9, and the transition probabilities of step 12
 - 19: **Irreducibility and steady-state probabilities:**
 - 20: If the Markov chain is not irreducible, divide it into its M_i
 - 21: corresponding irreducible Markov chains, otherwise $M_i = 1$
 - 22: Calculate the steady state probability for each of the M_i chains
 - 23: **Weighting factors:**
 - 24: Calculate the weighting factor using Eq. (6.7)
 - 25: **Adjusting to the IEEE 802.11 parameters:**
 - 26: Calculate the backoff factor α using Eq. (6.1)
 - 27: Determine the set of dominant and dominated chains, D_i and d_i ,
 - 28: respectively, as described in Section 6.2.4
 - 29: Calculate the modified weighting factors using Eq. (6.9)
 - 30: **end for**
 - 31: **Combining network solutions:**
 - 32: Combine the solutions of the different subnetworks to calculate
 - 33: the output rate y_n of node n using Eq. (6.10)
 - 34: Calculate the throughput t_n of node n using Eq. (6.11) or (6.13)
-

Numerical Results

We start this section by assessing the accuracy of the proposed modeling approach by comparing the model's outcomes with those delivered by a discrete-event simulator under various scenarios. Then, we study the computational complexity of the modeling approach as a function of the network topology and compare it to

Model A presented in Chapter 5. At last, we explore two possible applications of the model relating to the configuration and performance improvement of a WLAN.

Model validation

To evaluate the accuracy of the proposed model, we explore several scenarios with different values of various network parameters, such as the IEEE 802.11 standard, the mean frame length, the data rate, the topology and size of the network, and we compare the model's estimations to the simulation results delivered by the discrete-event network simulator ns3 [80].

Various network topologies and standard amendments

We begin by examining the proposed model's accuracy under different topologies and IEEE standard amendments. We consider three topologies: the four-node network of Fig. 6.4, the larger six-node network depicted in Fig. 6.8, and the ten-node network of Fig. 6.10. We recall that the nodes of a conflict graph represent only the APs that belong to the same communication channel. and are transmitting traffic to an associated user station. The four-node network uses the IEEE 802.11g standard amendment while the six-node and the ten-node networks use the IEEE 802.11n standard amendment.

The simulation setup is similar in all three topologies and Table 6.2 sums up the used parameters. We fix the input rates of all nodes to the default values given in Table 6.3. We then choose one node and we gradually vary its input rate from 0 to 1 by step of 0.05, providing us a total of 21 different simulation scenarios. Let us recall that $x_n = 0$ indicates that node n never has a frame to be sent, while $x_n = 1$ denotes that the node is always willing to transmit frames. To account for the intrinsic uncertainty of the measured quantities in a simulator, we replicate each simulation scenario 20 times and we calculate the 95% confidence intervals. However, given the length of the simulation runs and the number of replications, the computed confidence intervals are virtually indistinguishable from their mean values and we decided not to represent them in the following figures.

Parameter	Four-node	Six-node	Ten-node
Standard amendment	IEEE 802.11g	IEEE 802.11n	IEEE 802.11n
Simulation replications	20	20	20
Replication duration [sec]	60	90	120
Payload length [bytes]	1000	1000	1000
Data rate [Mbps]	54	65	65

Table 6.2: Simulation parameters.

Four-node network In our first scenario, we consider the four-node topology (depicted in Fig. 6.4) with the IEEE 802.11g standard amendment and a data rate of 54Mbps (maximum data rate in the standard amendment). Figure 6.7 shows the throughput evolution of all four nodes, named N1 through N4, as a function of the varying input rate of node 2, x_2 . We observe that as x_2 grows, so does the

Scenario	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
Four-node	0.3	0.5	1	0.5	/	/	/	/	/	/
Six-node	0.5	0.4	0.6	0.3	0.7	0.9	/	/	/	/
Ten-node	0.3	0.6	0.8	0.4	0.7	0.3	0.5	0.4	0.9	0.7

Table 6.3: Default input rates.

throughput of node 4, while nodes 1 and 3's throughputs decrease. Larger values of x_2 imply more competition between nodes 1, 2, and 3 in accessing the medium. The gain in node 4's throughput is not directly caused by the increased throughput of node 2, but rather as a by-product of the decrease of node 3's throughput (with whom node 4 compete for medium access). In fact, Fig. 6.7 suggests that the amount of throughput lost by node 3 is gained by node 4. Finally, we observe that our model was able to accurately capture all these behaviors.

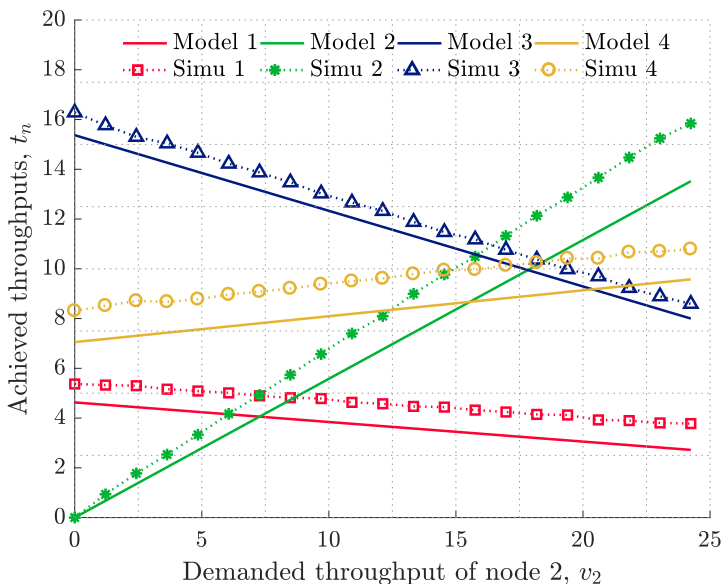


Figure 6.7: Four-node network of Fig. 6.4: varying the input rate of node 2.

For the sake of completeness, we repeat the same setup by varying the input rate of one of the other three nodes, giving us a total of $21 \times 4 \times 4 = 336$ points on which we calculate the relative error statistics presented in Table 6.4. We notice that in over 90% of the samples the relative error is less than 20%.

Scenario	Mean	Median	<5%	<10%	<20%	<30%	>30%
Four-node	12.67%	13.43%	10.00%	35.00%	91.25%	100%	0.00%
Six-node	9.80%	9.77%	21.10%	54.91%	97.75%	99.20%	0.80%
Ten-node	9.11%	7.47%	27.78%	68.12%	88.77%	99.36%	0.64%

Table 6.4: Distribution of the relative error for the throughput, t_n .

Six-node network Our second scenario addresses the larger six-node network of Fig. 6.8 and the IEEE 802.11n with a higher data rate of 65Mbps. Figure 6.9 shows the throughputs attained by each of the six nodes as the input rate of node 6 gradually increases from 0 to 1.

Not surprisingly, Fig. 6.9 shows that the throughputs of nodes 4, 5, and 6 are the most affected by the increasing input rate of node 6, as all three belong to the same clique. Node 6 increases its throughput mostly at the expense of node 5, that loses more than a third of its original throughput. Node 4's throughput decays to a lesser extent, however its already small throughput is even further decreased. We also notice that nodes 1, 2, and 3 are not directly affected by node 6 and they keep an almost steady throughput regardless of the value of x_6 . As in the four-node network, we repeat the same experience by letting the other nodes vary their input rate one by one. Because it is a network of 6 nodes, this gives us a total of $21 \times 6 \times 6 = 756$ samples that are used to derive the statistics shown in Table 6.4. The table shows the predictions made by our model fit well those delivered by the simulator with a mean relative error of less than 10%.

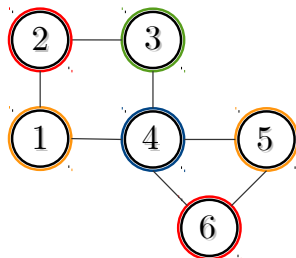


Figure 6.8: Conflict graph of a six-node network.

Ten-node network Our third scenario is a network of ten nodes (see Fig. 6.10) using IEEE 802.11n. We study the throughput attained by all nodes as a function of node 4's input rate.

Figure 6.11 shows the corresponding results. In order to keep the figure legible, we represent the attained throughput of only a subset of nodes. First, we observe that the variation of node 4's input rate causes its throughput to increase from 0 to approximately 20Mbps. On the other hand, as x_4 grows, the throughput of node 3 decays significantly (it is nearly halved). This agrees with the fact that node 3 is the only neighbor of node 4 (see Fig. 6.10). Because of node 3's declining throughput, nodes 1 and 2 experience a slight gain in their throughput as x_4 grows. As for the nodes far from node 4 such as nodes 8 and 10, their attained throughput is almost not influenced by the variations in x_4 . Finally, Fig. 6.11 shows that our model manages to capture all these behaviors with a good level of precision. Like in the two former scenarios, we repeat the same experiences letting the input rate of each of the other nine nodes vary from 0 to 1. This leads to a total of 2100 samples that we use to compute the statistics shown in Table 6.4.

Overall, we explore hundreds of examples with differing input rates, two different standard amendments, and three topologies. To provide a broader overview

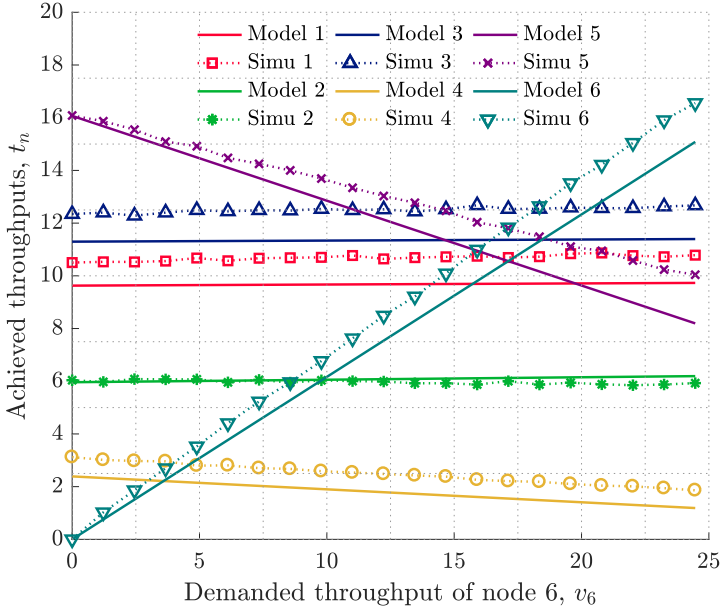


Figure 6.9: Six-node network of Fig. 6.8: varying the input rate of node 6.

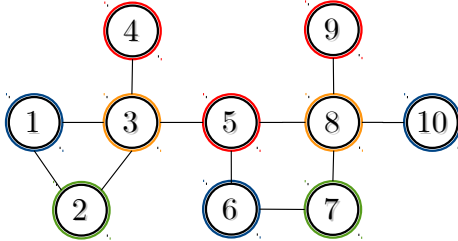


Figure 6.10: Conflict graph of a ten-node network.

of the accuracy reached by our model, we computed the mean and the median of the relative error as well as the distribution of the relative error attained on each scenario. Table 6.4 presents the corresponding results. The typical mean relative error is usually close to 10% and so is the median relative error. We also observe that in the vast majority of examples (around 90% of cases), the relative error made by our model is less than 20%.

Heterogeneous data rates

We now study the case when the nodes are heterogeneous with regard to their data rates. We reuse the six-node node network of Fig. 6.8, but we assign a different data rate to every node as indicated by Table 6.5. Note that under these settings, node 2 has a data rate that is five times that of node 5. Due to the heterogeneity in the data rates, each node has a different backoff factor, α (see Section 6.2.4) and

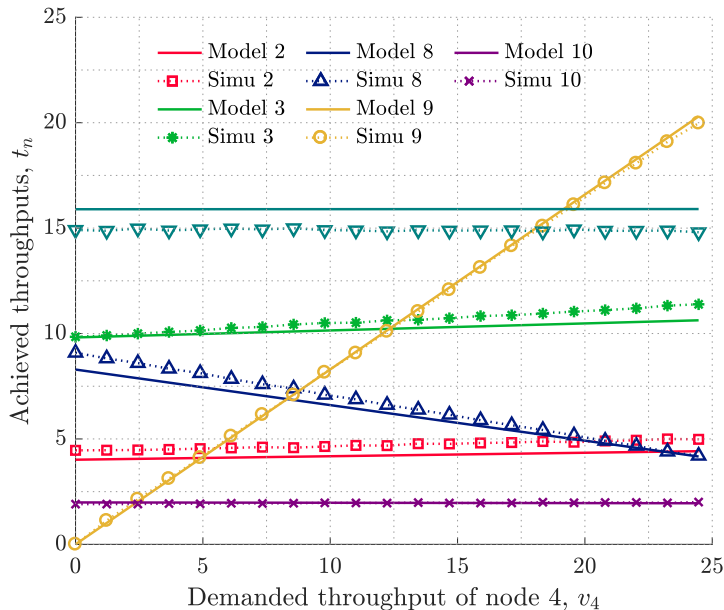


Figure 6.11: Ten-node network of Fig. 6.10: varying the input rate of node 4.

we derive the throughputs of nodes using Eq. (6.13). Analogously to the former scenarios, we let the input rate of each node vary from 0 to 1 while keeping the input rates of the other nodes to their default values (see Table 6.3). This gives us a total of 756 points on which we calculated the estimated throughputs using our model and compare these values to those delivered by the simulator.

Scenario	N1	N2	N3	N4	N5	N6
Six-node	18 Mbps	54 Mbps	24 Mbps	12 Mbps	9 Mbps	12 Mbps

Table 6.5: Data rates of the nodes in the six-node network in Fig. 6.8.

Table 6.6 presents the corresponding results. We notice that despite having nodes with significantly different data rates, our model is still able to deliver accurate estimations for the throughput. More precisely, the mean relative error of the model is 9% with 94% of the samples having an error less than 20%.

Scenario	Mean	Median	<5%	<10%	<20%	<30%	>30%
Six-node	9.31%	6.98%	20.29%	68.49%	94.13%	97.73%	2.27%

Table 6.6: Heterogeneous data rates: distribution of the relative error for the throughput, t_n .

Frame aggregation in IEEE 802.11n

In our last validation scenario, we study the model's precision when the nodes implement the aggregation feature. When frame aggregation is enabled, multiples

frames are concatenated into a single large frame before being transmitted, as described in Chapter 2. This tends to diminish the cost of the overhead introduced by the MAC protocol, thereby increasing the maximal achievable throughput.

Once again, we consider the six-node network of Fig. 6.8 with the input rates given in Table 6.3 and the simulation setup of Table 6.2. However, all six nodes now aggregate four MAC service data units (MSDUs) into a single frame at each transmission. While the simulator actually implements the aggregation features, in our model we simply introduce a four-fold extension of the frame length.

Figure 6.12 shows the attained throughputs of all nodes as a function of the input rate of node 6. We can assess the influence of the frame aggregation feature on this scenario by comparing Fig. 6.9 and Fig. 6.12. Although the trends exhibited by the throughputs are still comparable, we observe that the frame aggregation feature significantly increases (almost doubles) the attained throughput. Finally, Table 6.7 summarizes the mean, median, and distribution of the relative error for all the six scenarios when one of the nodes varies its input rate. Figure 6.12 along with Table 6.7 show that our modeling approach can successfully reproduce the behavior of a WLAN implementing frame aggregation.

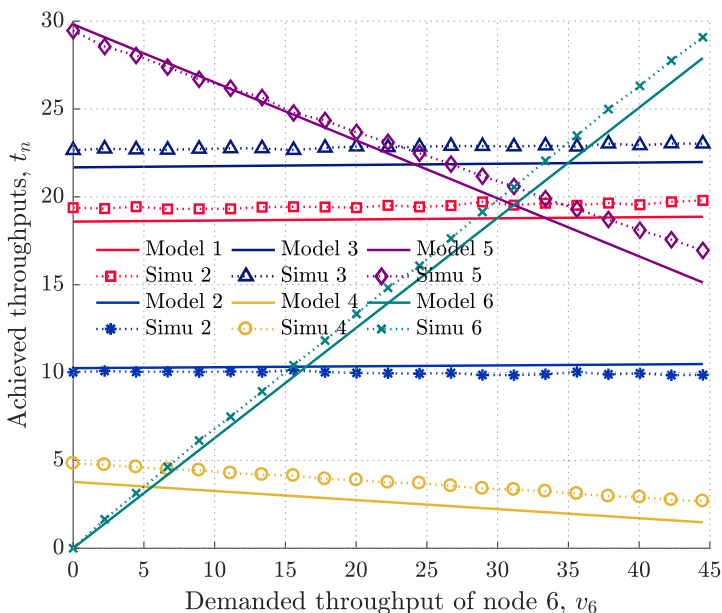


Figure 6.12: Frame aggregation for the network in Fig. 6.8: varying the input rate of node 6.

Scenario	Mean	Median	<5%	<10%	<20%	<30%	>30%
Six-node	6.04%	5.07%	49.06%	92.03%	98.28%	99.22%	0.78%

Table 6.7: Frame aggregation: distribution of the relative error for the throughput, t_n .

Applications

We provide two practical examples to illustrate how the proposed modeling framework can help in the deployment and configuration of an IEEE 802.11 WLAN.

Channel assignment

In our first example, we consider the well-known issue of channel assignment. In IEEE 802.11n and 802.11g networks, each AP can choose its channel among 14 different wireless channels in the 2.4GHz frequency range. However, out of these 14 channels, at most three can be chosen in a manner that no two channels have overlapping frequencies (see Chapter 2). Obviously, given the way APs share the channel, the choice of channel assignment considerably affects the network's performance.

We consider the 12-node network ($N = 12$) depicted in Fig. 6.13a with three non-overlapping channels. The nodes' input rates are given in Table 6.8. For the sake of convenience, we classify nodes into two categories: high-demanding nodes whose input rates are higher than 0.5, and low-demanding nodes whose input rates are below 0.5. Let a be a vector of length N that represents one possible allocation of the three channels among the N APs. We denote by $y(a) = \{y_1, y_2, \dots, y_N\}$ the set of output rates obtained when implementing the channel assignment a . Remind that y_n can be viewed as a measure of the normalized throughput attained by node n .

We consider four different performance metrics to evaluate the performance of the network:

1. Global satisfaction rate, GSR , or the proportion of the network's general throughput demand that has been met, calculated as:

$$GSR(y(a)) = \frac{\sum_{n=1}^N y_n}{\sum_{n=1}^N x_n}. \quad (6.14)$$

2. Jain's fairness index [78], J , that measures how fairly the throughput was divided among the nodes, calculated as:

$$J(y(a)) = \frac{\left(\sum_{n=1}^N y_n\right)^2}{\sum_{n=1}^N y_n^2}. \quad (6.15)$$

We also calculate the Normalized Jain's index, NJ . The normalization refers to accounting for the nodes' input rates when calculating Jain's index:

$$NJ(y(a)) = \frac{\left(\sum_{n=1}^N \frac{y_n}{x_n}\right)^2}{\sum_{n=1}^N \frac{y_n}{x_n}^2}. \quad (6.16)$$

3. Proportional fairness, PF , that is a trade-off between GSR and J as it tries to maximize both fairness and throughput by giving more throughput to nodes with higher demands:

$$PF(y(a)) = \sum_{n=1}^N \log \frac{y_n}{x_n} . \quad (6.17)$$

Scenario	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}
12-node	0.2	0.4	0.9	0.7	0.8	0.9	0.1	0.3	0.2	0.6	0.8	0.3

Table 6.8: Input rates of the 12-node network in Fig. 6.13a.

In practice, our model could be used jointly with existing solutions in the field of channel allocation, such as [81, 82]. A classical way of finding (sub)optimal channel allocations is to start from a given allocation, and then iteratively improve it with regard to some network performance parameters until convergence is found. In this context, our model could be used to quickly evaluate the performance parameters of interest at each iteration (rather than relying on long simulations). However, for the sake of simplicity and given the size of the network, we choose to explore all of the $3^{12} \simeq 530,000$ possible allocations and retain the ones maximizing one of the criteria given above. Figures 6.13b, 6.13c, and 6.13d illustrate the channel assignment that maximize GSR , J , and PF , respectively. For each of these three channel assignments, we also indicated in Table 6.9 their score over the other performance metrics.

Performance metric	GSR	J	NJ	PF
Fig. 6.13b max GSR	96%	0.725	0.983	-1.27
Fig. 6.13c max J	73%	0.796	0.955	-3.30
Fig. 6.13d max PF	95%	0.735	0.987	-1.08

Table 6.9: Evaluation of the proposed channel allocations.

When maximizing the global satisfaction rate, GSR , the retained solution maximizes the overall throughput obtained in the network and leads a GSR of 96%. Interestingly, we notice in Fig. 6.13b that all the high-demanding nodes (whose input rate is over 0.5) do not share the channel with any other node, thereby enabling them to obtain the highest possible throughput. On the other hand, when maximizing Jain's index, we observe in Fig. 6.13c that almost all nodes have a neighbor with whom they share the medium. In fact, with the exception of the pair of nodes 6 and 2, all the other pairs involve two nodes belonging to the same class (be it low-demanding or high-demanding nodes). As a consequence high-demanding nodes get lower output rates, as they have to share the medium with other high-demanding nodes. The optimal solution for the Jain's index increases its score from 0.725 to 0.796, at the expense of over 20% loss in the GSR . The last optimal solution maximizes the proportional fairness, PF . In Fig. 6.13d we observe that the only difference between the PF solution and the GSR solution lies in the selected channel of node 8. This similarity can be understood by the fact

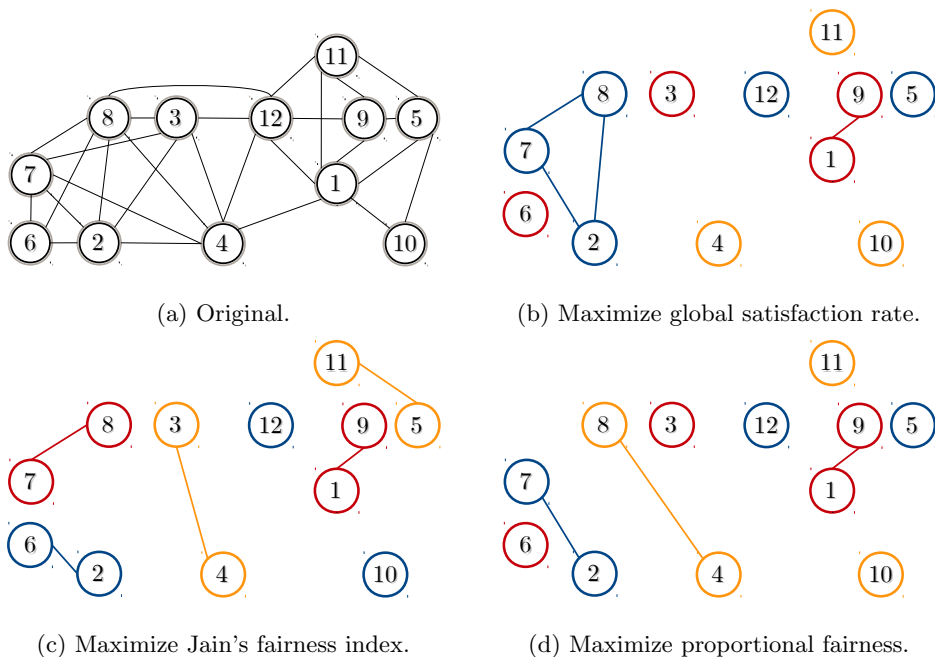


Figure 6.13: Different channel allocations for a randomly-generated 12-node network.

that proportional fairness, unlike Jain's fairness index, takes into consideration not only the output rate of each node but also its input rate. Overall, in this example, the optimal solution for Proportional fairness coincides with the optimal solution for Normalized Jain's index, and appears as a good trade-off between maximizing throughput or fairness, as it offers both a *GSR* value and *J* value that are remarkably close to their optimal values.

Upgrading from IEEE 802.11g to 802.11n

Our second example illustrates how our model can help when considering an upgrade of the IEEE 802.11 standard amendment deployed over the APs of a WLAN. More specifically, while the 802.11g amendment is still widely used, its maximum data rate of 54Mbps can be viewed as insufficient in some cases. Upgrading to 802.11n can be an attractive solution as it enables higher data rates and also implements the frame aggregation feature. However, by aggregating frames, there is a potential risk in deepening the effect of starvation that some nodes may already face. Therefore, a thorough analysis of an upgrade to 802.11n must include the benefits both in terms of overall throughput and fairness.

We consider the four, six, and ten-node networks presented in Section 6.3.1 together with the input rates given in Table 6.3. Let us denote by k the number of frames aggregated in each transmission. First, we run our model using IEEE 802.11g at 54Mbps (without aggregation, $k = 1$). Then, we rerun our model on the same network but using IEEE 802.11n at 65Mbps, while considering two possible

sizes for the frame aggregation, $k = 4$ and $k = 16$. We calculate the throughput gain the network experiences with aggregation, as opposed to without, as well as the Normalized Jain's index.

Table 6.10 shows the associated results. We observe that the gain in throughput is typically around 85% when aggregating four frames, while it reaches nearly 230% if frames are aggregated by batches of 16. We include in Table 6.10 the values found for the normalized Jain's index. It appears that frame aggregation has very little effect on Jain's index suggesting that the medium sharing between the nodes remains fair, regardless of the aggregation features. Based on these results, upgrading from IEEE 802.11g to 802.11n appears to be an attractive option.

Scenario	T.G. $k = 4$	T.G. $k = 16$	NJ $k = 1$	NJ $k = 4$	NJ $k = 16$
four-node network	86%	239%	0.981	0.965	0.953
six-node network	84%	233%	0.892	0.865	0.848
ten-node network	87%	238%	0.890	0.857	0.838

Table 6.10: Evaluating the gain in upgrading to IEEE 802.11n.

Discussion

We use this final section to discuss the complexity of the proposed modeling approach and compare it to our previous work. We then shortly review the solutions we implemented thus far as well as some potential improvements to the framework.

Modeling complexity

In this section, we explore how the computational complexity of our modeling framework increases as the size of the WLAN under study grows. Many existing modeling approaches [36, 37, 42, 83] and our previous model presented in Chapter 5, make use of a single Markov chain to describe the entire WLAN. The model presented here revolves around a Divide-and-Conquer approach that breaks the original problem into a set of smaller problems, each being solved individually as a smaller Markov chain.

Unfortunately, during this thesis we were not able to derive a closed-form expression for the number of states in the Markov chains involved in our modeling approach. This exercise is made difficult as the exact values depend significantly not only on the number of nodes in the network, N , but also on the network's density, aka the average node degree. We nonetheless provide an empirical study.

We randomly generate thousands of conflict graphs with size varying from $N = 5$ up to $N = 14$. We sort them into five groups based on their density: average node degree of less than 3, between 3 and 4, between 4 and 5, between 5 and 6, and between 6 and 7. Then, we calculate the mean number of (sending) states per Markov chain inside every density group. Figure 6.14 shows the corresponding results. As expected, the average number of sending states per subnetwork grows with increasing values of N . However, even for $N = 14$, the mean number of states per Markov chain tends to lie around 8, meaning that most involved Markov chains are very small. Figure 6.14 suggests that high-density networks tend to

result in slightly larger Markov chain. Indeed, consider the subnetwork $b_{16} = [ON ON ON ON]$ for the four-node network of Fig. 6.4. Three possible (sending) states exist: $[1 0 0 1]$, $[0 1 0 1]$, and $[0 0 1 0]$. However, if we remove the edge between nodes 1 and 2, then only two (sending) states are possible: $[1 1 0 1]$ and $[0 0 1 0]$. In the ultimate low-density network with no edges, there would be a single sending state: $[1 1 1 1]$. More generally, the fewer edges in a conflict graph, the smaller the density and the smaller the mean number of states per Markov chain.

The subplot in Fig. 6.14 shows the number of states in the classical single-chain Markovian model, aka Model A. As expected, the mean number of states is significantly larger (two orders of magnitude) when using a single Markov chain as opposed to a series of smaller Markov chains, and can lead up to several hundreds of states when the number of nodes closes 14. Hence, we chose to have a large number of smaller Markov chains, keeping in mind that the last stage of our approach, aiming to combine the solutions found for each subnetwork, is a simple summation of the stationary probability distributions over all the subnetworks using the law of total probability [84].

Overall, by splitting the original problem into many smaller problems, whose solutions can be easily parallelized, our Divide-and-Conquer strategy circumvents the dimensionality curse associated to large Markov chain for conflict graphs having up to a dozen nodes. In practice, with a non-optimized implementation, models are typically solved at a click-speed for N around 4 or 5, and within a couple of seconds for N near to 10.

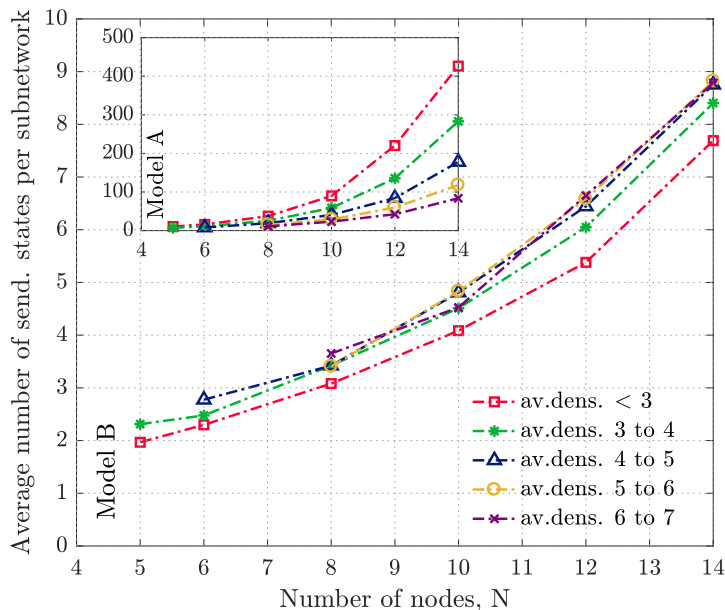


Figure 6.14: Number of (sending) states per Markov chain (subnetwork) as a function of the network's size and density.

Contributions and limitations

The presented modeling framework contains substantial improvements to its predecessor, Model A. The paradigm shift from a single Markov chain to a series of smaller Markov chains offers many possibilities for parallelization. Moreover, it allows us to define separate Markov chains and avoid having preempted or synchronizing nodes, making Model B significantly easier to comprehend. Although Model B has more information on input (data rate and frame size of each AP), and several more sophisticated computations are added, we believe that this intricacy is justified by the increased accuracy and applicability.

Another important advantage of Model B is that it incorporates different data rates and frame sizes. However, it can be argued that the implementation itself of the backoff function and the computation of each node's achieved throughput are done in a less than elegant manner. The backoff factor is simply the quadratic fit of an observed function. The heterogeneous data rates require an intricate computation of the final attained throughput that involves manipulating neighborhoods and cliques. The reason this mechanisms are complex is because they were additions to a modeling idea that already existed. Our initial idea was to develop a discrete time Markovian model that revolves around the Divide-and-Conquer approach. Only later did we decide that adding heterogeneity to the nodes would allow us to expand the set of possible applications by bridging the gap between our model and the simulated WLANs.

As we believe that these modeling problems can be cleverly solved, we propose in the next chapter an enhanced framework. Model C is specifically designed to accommodate current WLAN mechanisms, some of which were never consider in the modeling frameworks we proposed so far.

Chapter 7

Model C: Channel Bonding in IEEE 802.11ac

Introduction

During the course of this thesis, we constantly upgraded our models to make sure they follow current trends in WLANs. A major change in the performance of WLANs occurred as IEEE 802.11ac became widespread over the last several years. The 802.11ac standard amendment offers higher data rates than any of its predecessors, achieved by using a series of techniques such as channel bonding, frame aggregation, and high MCS indexes (detailed in Chapter 2). Our final Markovian model is conceived specifically for WLANs with highly heterogeneous nodes, such as those in 802.11ac. The model relies on the Divide-and-Conquer approach described in the previous chapter, however, this time we use a Continuous Time Markov Chain (CTMC) to model the network.

The transition from discrete to continuous time can be done almost seamlessly, yet its effect on the precision and versatility of the model is far from negligible. Having continuous time means that the nodes' output rates, provided by our model, are now a proportion of time and no longer a proportion of medium access. Thus, calculating a node's attained throughput becomes a trivial task instead of a complex manipulation of neighborhoods and cliques. More importantly, it allows us to avoid an entire layer of modeling, resulting in higher-accuracy predictions.

This versatility of the new framework allows us to perform an extensive exploration of the effects channel bonding has on resource sharing. We investigate issues of fairness and throughput maximization in different network topologies. The new study allows us to develop insights into the best practices for configuring channel bonding in 802.11ac networks.

In this chapter, we present our continuous time Markovian model that uses the Divide-and-Conquer approach. We begin by detailing how we can recalculate the backoff factor function so that we obtain its closed-form. We then present the modeling approach and validate its accuracy using ns3 simulation. In the application section, we present a study of channel bonding in 802.11 networks and its impact on the performance of the network. We conclude this chapter with a

short discussion on the strengths and limitations of the modeling approach, as well as some possible improvements.

Closed-form backoff factor α

There are two major improvements that we implement in Model C: continuous time Markov chains and a closed-form expression for the backoff factor α . We remind that α is the ratio between the average backoff duration and the average frame transmission duration (detailed in Section 6.2.1 of Chapter 6). Typically, a small α value indicates large frame sizes and/or low data rates. We found that in the FIM topology of Fig. 7.1, the middle node's throughput can be modeled as a quadratic function of the backoff factor, denoted by $f(\alpha)$. In the previous chapter, we simply provided the least squares fit for $f(\alpha)$ that we then implemented in Model B.



Figure 7.1: Conflict graph of a three-node FIM network.

After analyzing the behavior of the FIM topology in more detail, we concluded that there exists a straightforward computation of $f(\alpha)$. Figures 7.2 and 7.3 depict an example of a sequence of frame transmissions for a small and a large value of α , respectively. All three nodes are saturated and they are sending frames of the same size over a channel with the same data rate. The vertical dashed/white rectangles depict the periods when node 2 senses an idle medium.

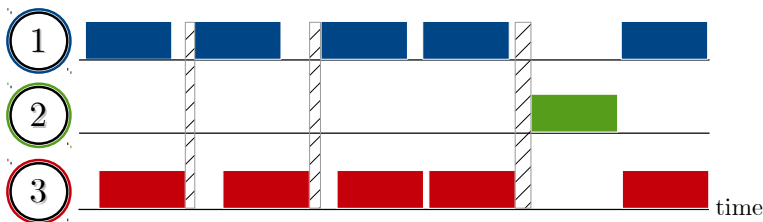


Figure 7.2: Resource sharing for a small α .

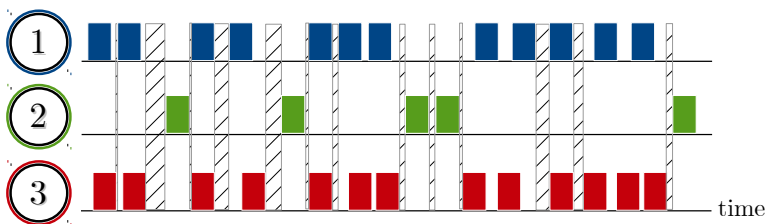


Figure 7.3: Resource sharing for a large α .

When α is small, it is unlikely that both nodes 1 and 3 are simultaneously in

backoff, leaving almost no opportunities for node 2 to transmit as shown in Fig. 7.2. As the value of α grows, it becomes more likely that node 2 senses an idle medium and thus the node obtains more channel access as shown in Fig. 7.3.

The probability that node 2 detects an idle medium can be calculated as:

$$P_{\text{idle}} = \left(\frac{T_{\text{backoff}}}{T} \right)^2, \quad (7.1)$$

where T_{backoff} and T are the average backoff and frame duration, respectively, as defined in Section 2.2 of Chapter 2, and assuming independence of node 1 and 3. Using Eq. 7.1 we can now calculate the mean time needed for node 2 to decrement its backoff to zero, i.e., the average time node 2 has to wait between two frame transmissions:

$$T_{\text{waiting}} = \frac{T_{\text{backoff}}}{P_{\text{idle}}}. \quad (7.2)$$

Analogously to Eq. (2.4) of Chapter 2, we can calculate node 2's achieved throughput as the ratio between its frame size and the time needed to send that frame:

$$t_2 = \frac{L}{T_{\text{waiting}}}, \quad (7.3)$$

and, finally, node 2's output rate is:

$$y_2 = \frac{t_2}{t_{2,max}}. \quad (7.4)$$

We verify our method by comparing the output rate node 2 obtains in simulation, the quadratic fit we used in Model B, and the newly calculated closed-form y_2 . Figure 7.4 shows the results. It is clear that the new alpha cannot significantly improve the accuracy of the model, however it represents a notable refinement to the conceptual complexity of Model C and a considerable advancement in our understanding of the system.

Modeling Approach and Algorithm

Our modeling framework is based on the idea that any unsaturated WLAN with an arbitrary topology can be modeled as a collection of simpler, saturated WLANs. We provide below a summary of the proposed model that implements Continuous Time Markov Chains (CTMCs) to represent the WLAN. We remind that this framework is largely based on Model B and we advise that Chapter 6 be read beforehand. However, the current chapter can be read on its own when only the model's high-level description is sufficient.

Network decomposition

In a WLAN, any unsaturated node that experiences periods of activity where it occupies or wishes to occupy the channel, and periods when it is idle because of the absence of frames to be sent. We refer to these two periods as the node's *ON* and *OFF* regimes. In a network of N nodes, there can be at most 2^N different combinations of nodes being in the *ON* or *OFF* regime. These combinations represent

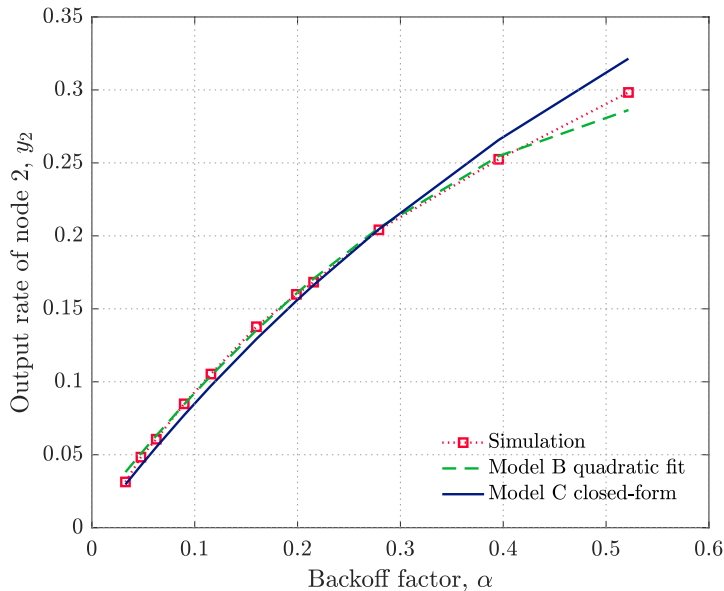


Figure 7.4: The backoff function in simulation and in the two models.

our *subnetworks*. In every subnetwork, a node is either saturated (if it is *ON*) or is completely removed from the subnetwork (if it is *OFF*). Depending on the nodes' input rates (see Section 4.1), some subnetworks are much more likely to occur than others. For example, in a WLAN where all nodes are saturated, i.e., all the input rates $x_n = 1$, all nodes have a frame to send at all times and only a single subnetwork exists. This leads us to calculate the subnetwork's *occurrence probability*. Let us denote by b_i ($i = 1, \dots, 2^N$) the i -th subnetwork whose occurrence probability is calculated as:

$$\beta_i = \prod_{n|b_i(n)=ON} x_n \prod_{m|b_i(m)=OFF} (1 - x_m), \quad (7.5)$$

where $b_i(n)$ is the regime of node n (*ON* or *OFF*) in subnetwork b_i .

Representing a subnetwork as a Markov chain

The next step is to solve all subnetworks independently of each other. We choose to model the subnetworks as CTMCs and in this section we detail how we calculate the states and transition probabilities of those Markov chains.

Finding the states of the Markov chain, their possible transitions and transition probabilities

Knowing that node n is *ON* or *OFF* does not fully inform us whether the node is currently sending or not. For this reason, we define the set of the network's *sending states*, denoted by S . S contains all combinations of sending states $s_k \in \{0, 1\}^N$ that satisfy the rule that two neighboring nodes cannot be simultaneously sending. We denote by S_i the subset of S containing all the sending states associated to the subnetwork b_i such that they satisfy:

1. if $b_i(n) = OFF$ then node n is never sending ($s_k(n) = 0, \forall s_k \in S_i$)
2. if $b_i(n) = ON$ and n has no sending neighbors, then n is sending.

The first rule is fairly easy to deduce, since a node that has no frames to send cannot be in transmission. The second rule is inspired by CSMA/CA's functioning and the works of Durvy et al. [37] where the authors show that CSMA/CA tends to maximize the number of concurrent transmissions in order to achieve higher spatial reutilization.

Once we have the set of sending states for every subnetwork, we proceed by calculating the probabilities of the transitions between them. We consider that a transition from sending state s_k to s_ℓ is possible if:

1. one node n changes from $s_k(n) = 1$ to $s_\ell(n) = 0$ and one node n changes from $s_k(n) = 0$ to $s_\ell(n) = 1$, or;
2. $k = \ell$, i.e., self-transition,

and all other transitions are considered to have a probability of zero. These two rules capture the fact that it is highly unlikely for two nodes to independently start (or finish) transmitting at the exact same time. If the transition from state s_k to s_ℓ has a non-zero probability, then we calculate that probability as:

$$P_{k,\ell} = C \prod_{n|s_\ell(n)=1} \frac{1}{1 + \sum_{m \in w_n} \mathbb{1}_{b_i(m)=ON}}, \quad (7.6)$$

where C is a normalizing constant that ensures $\sum_{\ell \geq 1} P_{k,\ell} = 1$. We denote by w_n the set of neighbors of n that compete with n for medium access, i.e., nodes that are ON and that do not have a currently transmitting neighbor. In short, we consider that n and its competing neighbors have an equal chance of gaining medium access. For a more detailed explanation, we refer to Section 6.2.4 of the previous chapter.

Calculating the steady state probabilities and introducing holding times

So far, for every subnetwork b_i we have a DTMC that is fully defined by its states and their transition probabilities. We now proceed by describing how we solve each DTMC and how we then transition to a CTMC.

Because some transitions between sending states are impossible, it may happen that the DTMC is not irreducible. Should that be the case, we simply divide the DTMC into its irreducible components and solve them separately. We denote by M_i the number of irreducible DTMCs in subnetwork b_i . We enumerate the irreducible DTMCs by c_i^m , $m \in [1, \dots, M_i]$ and we use π_i^m to denote their corresponding steady state probabilities. Section 6.2.4 in Chapter 6 explains this procedure using a sample network for more clarity.

Next, we transition from discrete to continuous time by introducing the *holding times* of sending states. In the case of a WLAN, it is logical that the holding times of the sending states represent the durations of the frame transmissions. For example, should the middle node in FIM network of Fig. 7.1 be sending, i.e., $s_k = [0 \ 1 \ 0]$, then we expect that the holding time of that sending state can be

modeled as the middle node's transmission time. As several nodes are potentially transmitting in a given state s_k in the subnetwork b_i , we calculate holding time of s_k , $h_i(k)$ as:

$$h_i(k) = \frac{1}{\sum_{n|s_k(n)=1} \frac{1}{T_n}}, \quad (7.7)$$

where T_n is the transmission duration of node n as defined in Eq. (2.3) of Chapter 2. Then, we obtain the solution to each irreducible CTMC by applying:

$$\mu_i(k) = \frac{\pi_k h_k}{\sum_{s_\ell \in S_i^m} \pi_\ell h_\ell}, \quad (7.8)$$

where $S_i^m \in S_i$ is the set of sending states associated to the m -th irreducible Markov chain (clearly, $S_i = S_i^m$ when the subnetwork has a single irreducible Markov chain).

It is important to note that in the case of multiple irreducible CTMCs, we need to calculate the entry probability of each chain. In such cases, we refer once again to the work of Durvy et al. [37] and try to replicate the idea that CMSA/CA networks tend to maximize the number of concurrent transmissions by assigning larger entry probabilities to the irreducible CTMCs with a larger number of transmitting nodes. We denote the entry probability of the m -th irreducible CTMC of subnetwork b_i by $\tilde{\omega}_i^m$ and we refer the reader to Section 6.2.4 of Chapter 6 for a detailed explanation of its calculation.

Calculating output rates and achieved throughputs

Finally, we can calculate node n 's output rate using Equation (7.9) that is simply the sum product of the probabilities of all the *subnetwork* \times *Markov chain* \times *sending state* combinations in which node n is sending.

$$y_n = \sum_{i=1}^{2^N} \left\{ \mathbb{1}_{b_i(n)=ON} \times \beta_i \times \sum_{m=1}^{M_i} \left(\tilde{\omega}_i^m \times \sum_{k|s_k \in S_i^m} \left(\mathbb{1}_{s_k(n)=1} \times \mu_i^m(k) \right) \right) \right\}. \quad (7.9)$$

It should be noted that the passage from DTMC to CTMC allows us to largely simplify the computation of a node's achieved throughput. When using DTMCs, the node's output rate can be seen as the proportion of medium accesses the node obtains in its neighborhood, thus its achieved throughput depends on its own data rate as well as the data rate of its neighbors (see Section 6.2.5 in the previous chapter). On the other hand, when using continuous time, the node's output rate y_n is the percentage of time the node occupies the channel and obtaining its achieved throughput is as simple as multiplying y_n by the node's maximum achievable throughput:

$$v_n = y_n \times t_{n,max}. \quad (7.10)$$

This simplification allows for two distinct improvements of the model. First, the computation itself is less complex and requires only the knowledge of the node's maximum achievable throughput. Second, we are no longer obligated to introduce another modeling hypothesis, i.e., derive the maximum throughput achievable per

clique, and instead we use an exact and measurable value, making our predictions more accurate. The second improvement is especially important in networks with highly-heterogeneous nodes. In 802.11ac WLANs, for example, nodes can easily have a ten-fold difference in their maximum achievable throughputs. In such cases, having the exact value and not a clique-average can highly influence the model's accuracy.

Algorithm

Because Model C is largely based on Model B, Algorithm 3 presents only the modifications highlighted in blue.

Algorithm 3 Model C

- 1: ...
 - 2: **Irreducibility and steady-state probabilities:**
 - 3: If the Markov chain is not irreducible, divide it into its M_i
 - 4: corresponding irreducible Markov chains, otherwise $M_i = 1$
 - 5: Calculate the steady state probability for each of the M_i chains
 - 6: Introduce the holding time $h_i(k)$ for sending state s_k of subnetwork
 - 7: b_i using Eq. (7.7)
 - 8: Calculate the solution of the CTMC using Eq. (7.8)
 - 9: ...
 - 10: **Adjusting to the IEEE 802.11 parameters:**
 - 11: Calculate the closed-form backoff factor α described in Section 7.2
 - 12: Determine the set of dominant and dominated chains
 - 13: Calculate the modified weighting factors
 - 14: ...
 - 15: **Combining network solutions:**
 - 16: Combine the solutions of the different subnetworks to calculate
 - 17: the output rate y_n of node n using Eq. (7.9)
 - 18: Calculate the throughput t_n of node n using Eq. (7.10)
-

Model Validation

We use this section to validate the accuracy of our model by comparing its predicted throughput to simulation results. Analogously to previous chapters, we validate the model by comparing its prediction to the results delivered by a network simulator. All our simulations are performed in the ns3 discrete-event network simulator [80] using the IEEE 802.11ac standard amendment. We simulate the system described in Section 4.1 of our Preamble, where a WLAN is composed of several APs that send traffic to their associated stations. APs have different *ON* and *OFF* periods that result in different input rates (see Eq. (4.1) in Preamble). All APs generate datagrams of 1500 bytes that they transmit to their associated stations. We present here the results for the nine-node network of Fig. 7.5 and the ten-node network of Fig. 7.6.

The specific parameters regarding the rate of frame aggregation, channel bonding, MCS, and the input rates of all nodes are summarized in Tables 7.1 and 7.2.

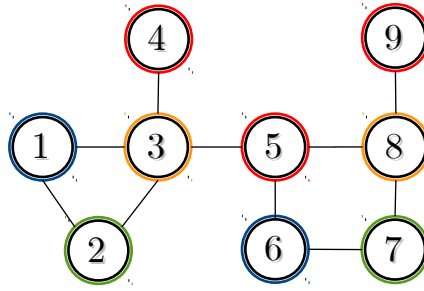


Figure 7.5: Conflict graph of a nine-node network.

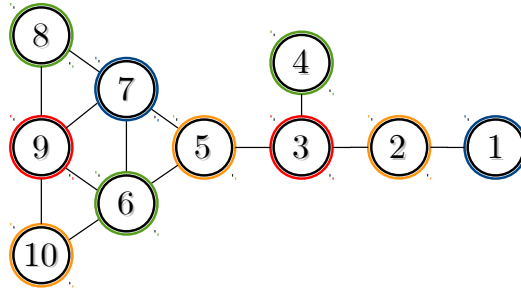


Figure 7.6: Conflict graph of a ten-node network.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
MCS	9	8	7	1	4	6	5	2	3
Channel size	40	20	80	40	20	80	40	20	80
Frame aggr.	4	4	8	2	8	4	2	4	8
Input rate	0.5	0.2	0.7	0.4	0.9	0.3	0.8	0.6	0.9

Table 7.1: Traffic parameters for the nine-node network.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
MCS	9	8	5	1	3	5	7	2	4	6
Channel size	40	80	20	40	20	80	40	80	20	40
Frame aggr.	2	4	8	8	8	4	4	2	2	8
Input rate	0.5	0.6	0.2	0.4	0.9	0.3	0.8	0.7	0.9	0.1

Table 7.2: Traffic parameters for the ten-node network.

In order to compare the model to the simulation data, we need a dataset of measurements and a definition of the relative error we use. Similarly to our previous model validations, we obtain our dataset by fixing the input rates of all nodes except for one, and we then vary the input rate of that one node in the interval $[0, 1]$ (recall that $x_n = 0$ and $x_n = 1$ mean that node n has no frames to send and node n is saturated, respectively). The varying input rate takes 11 different values $(0, 0.1, 0.2, \dots, 1)$. To make sure we obtain a representative value of the nodes' output rates we execute 20 simulation runs for every scenario. In total, we obtain $N \times 11 \times 20$

different samples for a network of N nodes. We then run our model using the same parameters as the simulation scenario and we obtain our predicted output rate, denoted by \tilde{y}_n . Finally, we calculate the relative error as:

$$e = \sum_{n=1}^N \frac{|\tilde{y}_n - y_n|}{y_n}, \quad (7.11)$$

where y_n is output rate of node n provided by the simulator, i.e., our ground truth. Note that when y_n and \tilde{y}_n have a value lower than 0.1 we remove that data point from the relative error dataset, as even small differences between the two values result in abnormally large relative errors.

The cumulative distribution function (CDF) of the relative errors for the nine and ten-node network is shown in Fig. 7.7. In general, we obtain a mean relative error of 9.03% for the nine-node network and 6.48% for the ten-node network, with a median of 7.09% and 4.86%, respectively. In both networks the relative error is rarely greater than 20% and it never exceeds 50%.

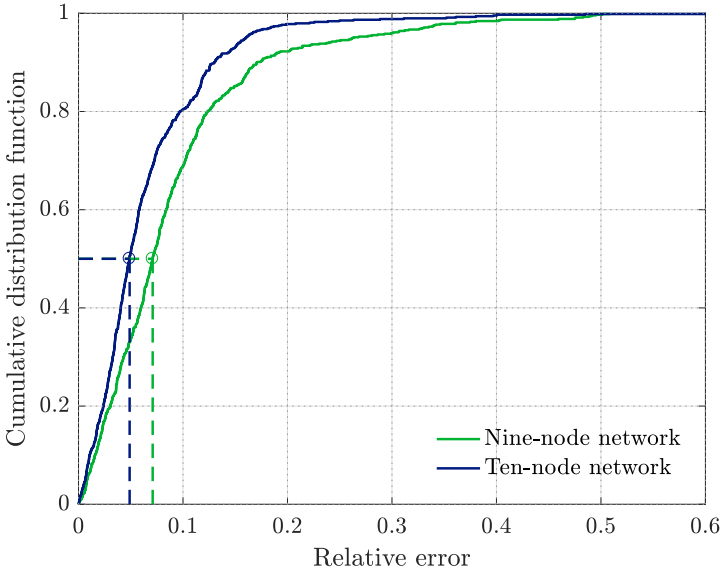


Figure 7.7: CDF of the relative error.

Figures 7.8 and 7.9 show the evolution of the achieved throughputs of several nodes when node 1 and node 10 vary their demanded throughput, i.e., input rate, in the nine and ten-node network, respectively. In the interest of legibility, both figures show only several selected nodes that are representative of the general behavior of all nodes. Both figures show node n 's estimated throughput in solid lines (named Model n) and the average simulation scenario throughput in discrete points (Simu n).

In the nine-node network, the varying demanded throughput of node 1 mostly impacts its one-hop neighbors. Nodes 2 and 3 need to forgo a part of the resources they use in order to accommodate the newly increased demand of their neighbor node 1. Nodes located at several hops from node 1, such as nodes 7 and 9, are only

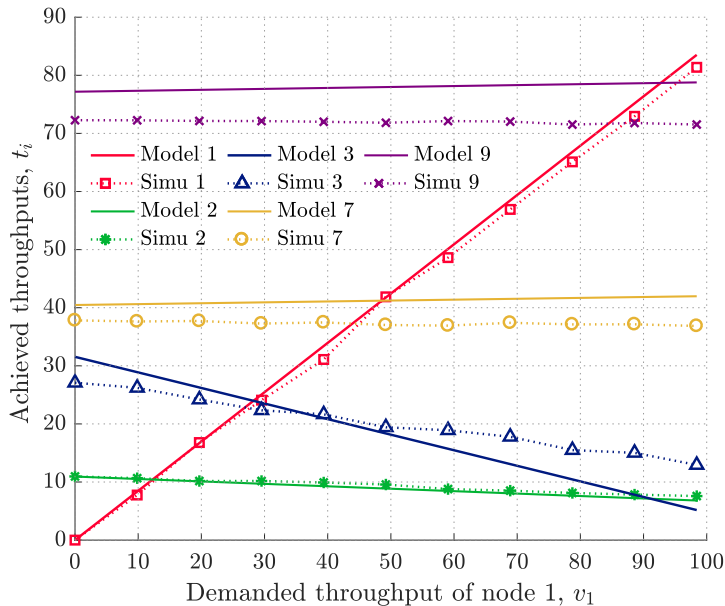


Figure 7.8: Nine-node network of Fig. 7.5: varying the input rate of node 1.

slightly affected by the dynamics happening at the right-hand side of the network. As node 1’s demand varies, nodes 7 and 9 keep an almost constant throughput both in the simulation and in our estimation. We notice similar behavior in nodes 5, 6, and 8.

The results for the ten-node network show similar tendencies, presented in Fig. 7.9. This time we vary the input rate of node 10. As node 10 approaches saturation, we notice the behavior of a flow-in-the-middle topology composed of nodes 8, 9, and 10. In these topologies, the edge nodes ally in order to maximize their throughput, at the expense of a starving node in the middle [20]. As node 10 goes from 0Mbps to almost 80Mbps, node 8 nearly doubles its initial throughput and node 9 slowly starts to experience starvation. Remote nodes show almost no change in their achieved throughputs, such as nodes 1 to 5. As in the nine-node network, the proposed model manages to accurately capture the tendencies of all nodes.

Finally, it should be noted that a single 60 seconds simulation run of the ten-node network with the parameters in Table 7.2 in ns3 takes, on average, 21 minutes to execute. The unoptimized implementation of the proposed model takes roughly 3 seconds for the same network. We used our laboratory’s Dell Inc. PowerEdge R630 workstation with 2 Intel^(R) Xeon^(R) CPUs E5-2697 2.60GHz, 28 cores, and 252 GB RAM.

In summary, we found that our model provides relative errors generally smaller than 10% for networks of different topologies and different levels of node heterogeneity. Moreover, we obtain our model’s solution at a fraction of the time needed for an ns3 simulation. In the next section, we take advantage of our model’s precision and its possibility to adapt to 802.11ac parameters to study the impact of channel bonding in several network scenarios.

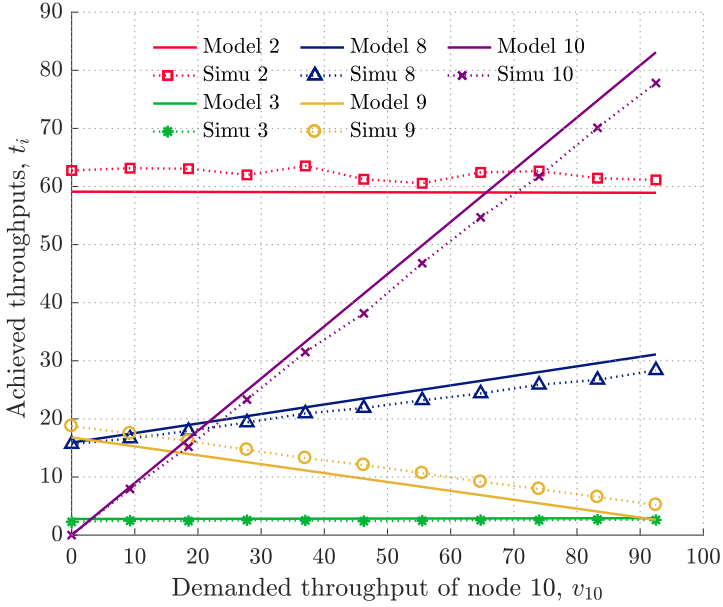


Figure 7.9: Ten-node network of Fig. 7.6: varying the input rate of node 10.

Channel Bonding in IEEE 802.11ac

Channel bonding is one of several techniques implemented in recent 802.11 standard amendments that allow for a significant increase in the available data rates. However, having larger channels can also lead to denser neighborhoods and there is a trade-off to be made between a high data rate and a large set of neighbors. We use this section to explore a series of channel bonding scenarios that gradually increase in complexity. First, we study the impact of channel bonding on a network's performance and examine how well channel bonding performs when the nodes' MCS indexes and frame aggregation rates vary. Next, we study a series of saturated four-node networks with different topologies, allowing us to have more insight into the best practices for proper configuration of WLANs under different given performance metrics. Finally, we present how the model helps us find the optimal channel bonding configuration of a given network with respect to a given performance metric.

Impact of MCS and frame aggregation on channel bonding

We consider a WLAN containing four APs in close proximity that either share a single 80MHz channel, or use four non-overlapping 20MHz channels. All APs are saturated and generate datagrams of 1500B on an error-free channel. We acquire the AP's attained throughputs through simulations in the ns3 network simulator [80]. The four APs, on average, obtain the same throughput, meaning that the only quantity we need to monitor is the overall *network throughput*, V ,

calculated as:

$$V = \sum_{n=1}^4 v_n, \quad (7.12)$$

where v_n is the achieved throughput of AP n . Figure 7.10 shows the network throughput for MCS indexes 0 to 8. We also vary the maximum number of MPDUs the nodes aggregate in every frame, called the aggregation rate. Note that the three aggregation limits defined in Section 2.1.3 of Chapter 2 override our aggregation rate. This means that for some MCS index and channel size combinations the actual number of aggregated MPDUs will be smaller than our aggregation rate. For example, when using MCS0 and a 20MHz channel, we can at most aggregate two MPDUs of 1500B due to the limit on the maximum transmission duration. In the following simulation results, we vary the aggregation rates from 2 to the maximum number of MPDUs and use either a shared 80MHz channel or four separate 20MHz channels.

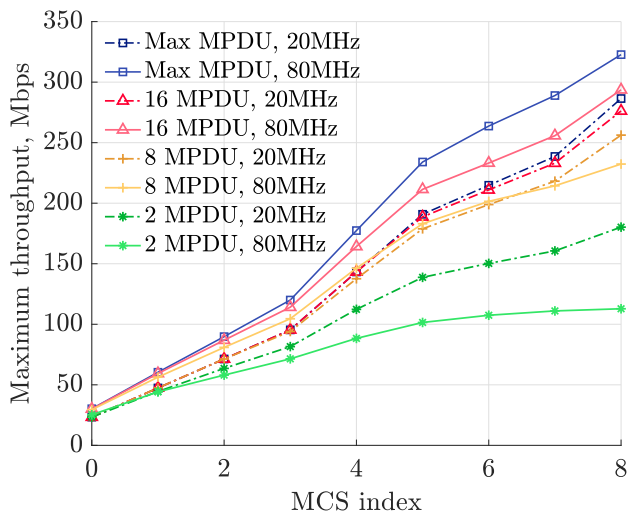


Figure 7.10: The maximum network throughput as a function of MCS with varying channel size and frame aggregation rate.

In agreement with previous works on the performance of frame aggregation [26, 49], Fig. 7.10 shows that the network throughput always increases with higher frame aggregation rates. The same is true for increasing the nodes' MCS indexes when the stations and APs are close enough to support the denser modulations. Moreover, we notice that the higher-order MCS indexes are more impacted by the changing frame aggregation rates and channel sizes. Specifically, the best and worst network throughputs for MCS0 are around 30Mbps and 23Mbps. In contrast, when using MCS8 we see an almost three-fold increase in throughput between the lowest (112Mbps) and the highest available throughput (313Mbps). Overall, we notice an over 15-fold increase in throughput when moving from the worst case scenario (MCS0, 20MHz channel, 2 MPDUs) to the best-case one (MCS8, 80MHz channel, maximum number of MPDUs).

We now study the impact of channel bonding under different network param-

ters. Figure 7.10 shows that both the highest and the lowest network throughputs for all MCS indexes are achieved using channel bonding, i.e., a single 80MHz channel. The lowest throughput is achieved with the aggregation rate of 2 MPDUs, and the highest with the maximum possible aggregation rate. In other words, when we have no information about the frame aggregation rate, using 20MHz channels appears to be the safe option, as it will never provide a throughput as low as a badly configured 80MHz network. However, if we know that the APs are likely aggregating more than 8 MPDUs, then we can easily gain up to 40Mbps by simply using an 80MHz channel.

We identified several reasons that explain the described behavior and conclude that when choosing an 80MHz or a 20MHz channel there is a trade-off to be made between four factors: the capture effect, the maximum aggregation rate, the guard intervals, and the DCF overhead. When all four APs share the 80MHz channel, sometimes transmissions will happen simultaneously and benefit from the capture effect. As explained in Section 2.1.1 of Chapter 2, the 80MHz channel includes the guard intervals separating the different 20MHz channels, resulting in data rates slightly over four times higher than a single 20MHz channel. Because of the higher data rates, transmissions are shorter. Shorter transmissions also allow for the aggregation of more MPDUs at once, giving a large advantage to the 80MHz channel (see Section 2.1.3 in Chapter 2). However, another side effect of short transmissions is that the DCF overhead on the 80MHz channel becomes too important relative to the data transmission time. This gives the 20MHz channels an advantage as, on average, we send four frames on every DCF overhead.

In summary, our comparison shows us that using channel bonding in our sample network of four fully-connected APs is more beneficial when the APs aggregate more MPDUs, especially when using higher MCS indexes. When nodes aggregate less than 8 MPDUs, it is better to use separate channels and have concurrent transmissions to limit the effect of DCF overhead. Overall, we come to the conclusion that finding the optimal channel configuration is far not straightforward even in a simplified (saturated, fully-connected) network. In the rest of this section we try to clarify and interpret the practical implications of these findings on networks with different topologies.

Saturated networks with arbitrary topologies

We are now interested in how introducing the notion of topology affects the performance of channel bonding. In the previous section, all nodes were statistically identical and obtained, on average, the same throughput. We now investigate the impact of channel bonding on issues of fairness in channel access in networks with arbitrary topologies, i.e., not fully-connected or edgeless conflict graphs. We use proportional fairness to quantify the degree of equity in resource sharing experienced by the nodes. We calculate the proportional fairness for a given demanded/achieved throughput combination as:

$$PF = \sum_{n=1}^N \log \frac{v_n}{t_n} , \quad (7.13)$$

where t_n and v_n are the demanded and achieved throughput of node n , respectively. When maximizing proportional fairness we are looking for the channel allocation

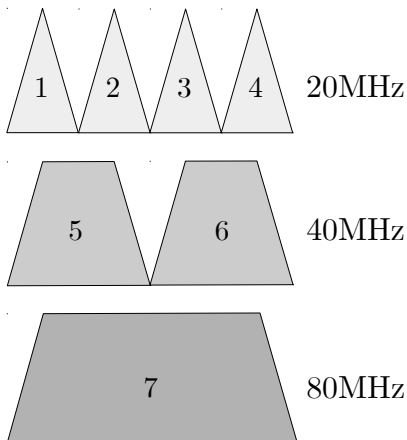


Figure 7.11: Enumeration of available channels.

that renders the PF value closest to zero, i.e., allow nodes with higher throughput demands to have higher obtained throughputs without creating starvation in low-demanding nodes.

We study the four-node networks in Fig. 7.12 and we consider the scenario where every AP can choose one of the seven wireless channels depicted in Fig. 7.11. This setup results in $7^4 = 2,401$ possible channel allocations. We use a quadruplet to denote the current channel allocation, e.g., $\{5, 2, 7, 6\}$ denotes the allocation of channel 5 to node 1, channel 2 to node 2, channel 7 to node 3, and channel 6 to node 4. The channels in Fig. 7.11 that are vertically aligned use the same frequency, e.g., the 80MHz channel 7 contains the four 20MHz channels 1 to 4, as well as channels 5 and 6. Our initial conflict graphs are based on the physical proximity of nodes that allows them to detect each others' transmissions supposing that they are all using the same channel. When a given channel allocation is chosen, we can construct a new conflict graph as a sub-graph of the initial conflict graph. The new graph reflects the modified neighborhoods as nodes are not necessarily using overlapping channels. In the new conflict graph we keep the edges between the nodes whose frequencies are at least partially overlapping. For example, the allocation $\{1, 2, 3, 4\}$ for the network in Fig. 7.12a renders an edgeless graph as the four nodes use four separate channels. If the current channel allocation is $\{7, 1, 2, 7\}$ then we keep the edges between nodes 1 and 2 and between nodes 3 and 4.

We assume that all nodes are saturated, and they have an aggregation rate of 8 MPDUs using MCS0 or MCS8. We remind that when aggregating 8 MPDUs and using MCS0 the highest throughput is obtained on the 80MHz channel, while for MCS8 it is better to use 20MHz channels (see Fig. 7.10). This threshold behavior allows us to have a wide variety of different optimal solutions by simply changing the MCS index of one or several nodes. Once we fix the MCS indexes for all nodes, we have fully defined all the parameters needed to run our model. We find the optimal channel allocation by running our model on the 2,401 possible allocations and we choose the allocation with the highest network throughput V or the highest proportional fairness PF . Given how fast the number of channel allocations grows

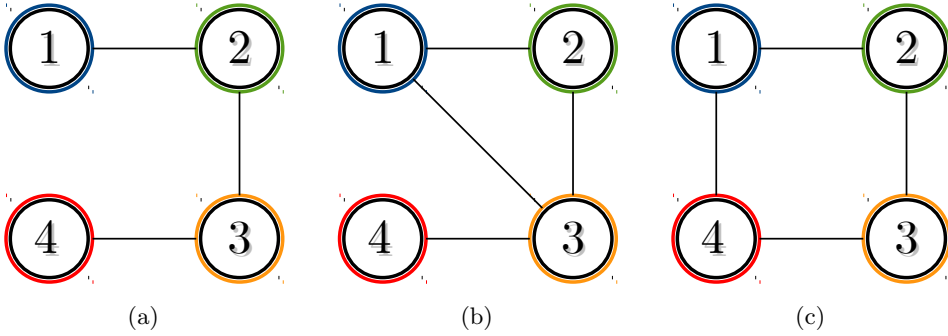


Figure 7.12: Initial conflict graphs of the four-node networks.

with both the network size and the number of available channels, the benefits of our modeling approach are important as the model provides us a speedy and accurate estimation of the nodes' throughputs.

We begin with the four-node chain network in Fig. 7.12a in which all nodes are using MCS8. For this scenario, we obtain the same channel allocation $\{5, 6, 5, 6\}$ as the optimal solution for maximizing both V and PF . Our results show that if one of the edge nodes falls back to the MCS0, then the optimal allocation does not change. However, should node 2 (or 3) switch to MCS0, the model proposes us to switch to the channel allocation $\{7, 7, 5, 6\}$ (or $\{5, 6, 7, 7\}$) if we wish to optimize V . This allocation creates starvation in node 2 (or 3) as this node is now in a the FIM topology [50]. The edge node takes advantage of having only one neighbor that never gains medium access and increases its obtained throughput to almost its maximum achievable throughput. While this solution maximizes throughput, it does so by creating starvation in one node. If we wish to maximize proportional fairness, it is still better to keep the nodes on separate channels and use the allocation $\{5, 6, 5, 6\}$.

Next, we test the network in Fig. 7.12b. Once again we find that many of the maximum throughput solutions consist in creating starvation in the middle node 3. In fact, whenever node 3 and node 1 (or node 2) both use MCS 8, then the maximum throughput solution is always $\{5, 6, 5, 7\}$, i.e., node 3 shares the channel with node 1 and node 4 and thus experiences FIM starvation. In this network, it is logical that maximizing throughput often means creating starvation in node 3, as it will, on average, allow two other nodes to transmit simultaneously. In general, we find that the solutions that maximize throughput tend to use larger channels (channels 5, 6, and 7) than those maximizing fairness. When maximizing fairness, we often have to separate the nodes on different channels, so that the unfavorable topological position of a node, such as node 3, does not prevent it from achieving a higher throughput.

Our last network is a cyclic network of four nodes depicted in Fig. 7.12c. Except for the configurations when all nodes or all nodes but one use MCS0, the solution that maximizes both network throughput and fairness is to use the $\{5, 6, 5, 6\}$ channel allocation. Interestingly, in this network, the maximum network throughput is never achieved by creating starvation. Because of the circular nature of the network, it is impossible to recreate the FIM topology without forcing starvation

in two nodes. However, in that case the two high-throughput nodes cannot compensate for the low throughput of the two others, and as a result the solution is to always have a more fair resource sharing.

From the study of the four node networks, we conclude that the network's topology can highly influence the optimal choice of channel allocation, even in the simplified conditions of a small saturated network. In the next section we extend this study by applying our model to a larger unsaturated network.

Larger unsaturated network

Having studied several simplified network configurations, we are now interested in the impact of channel bonding on a larger network where nodes are unsaturated and have more diverse sets of MCS indexes and aggregation rates. We study the network in Fig. 7.6 using the parameters defined in Table 7.3. Each of the ten APs can choose to use one of the three channels in Fig. 7.13, where channels 1 and 2 are 80MHz wide and channel 3 is 160MHz wide. Analogously to the four-node scenarios, we define the new conflict graph for every channel allocation by keeping the conflicts between nodes belonging to at least partially overlapping channels. This setup results in $3^{10} = 59,049$ possible channel allocations.

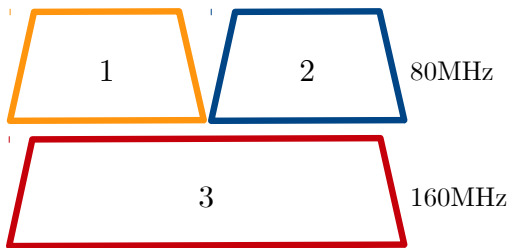


Figure 7.13: Enumeration of available channels.

Figure 7.14 shows the results we obtained when we maximize the network throughput and the proportional fairness. We first notice that the two metrics provide the same configuration for the first four nodes. This is the low-density part of the network, where it is possible to completely separate the four nodes even with only two non-overlapping channels. Although only node 1 can fully obtain its demanded throughput on a 80MHz channel, all four nodes achieve at least 50% of their demanded throughputs and none of them are in starvation. On the left-hand side where the network becomes denser, we can no longer provide a separate channel for every node and the two performance metrics provide very different solutions. If we maximize the network throughput V , we still obtain a connected component containing nodes 5 to 10. In this component, nodes 6 and 9 are analogous to the middle node of an FIM topology and they experience starvation by obtaining a throughput of 0.7Mbps and 1.1Mbps while demanding a throughput of 45Mbps and 50Mbps, respectively. Their neighbors, however, obtain throughputs as high as 79Mbps and the network throughput is $V = 262$ Mbps. When maximizing the proportional fairness PF , we notice that a chain network equivalent to the one in Fig. 7.12b is created using nodes 6, 7, 8, and 10. Characteristically for the four-node chain, the throughput is shared fairly as long as the node's throughput demands

are somewhat similar as it is too costly to create starvation in the two middle nodes for the benefit of the edge nodes. However, the overall network throughput in this case is almost 60Mbps lower than the maximum throughput configuration.

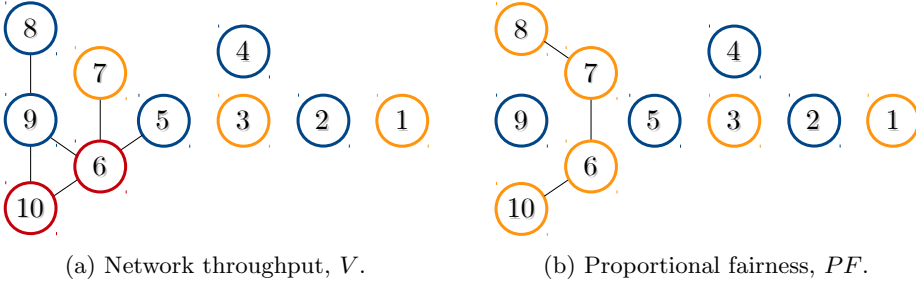


Figure 7.14: Channel allocation solutions for maximizing different performance metrics.

In general, we conclude that the optimal solution is highly dependent on the nodes' demanded throughputs and their levels of saturation by testing several sets of configurations. However, we find that when optimizing the network throughput V , the solution often involves using larger channels at the expense of creating starvation in some nodes. The use of smaller (and separate) channels when maximizing proportional fairness PF leads to smaller overall network throughputs, as we avoid the network's inherent working that maximizes the number of concurrent transmissions.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	V	PF
MCS	8	7	4	0	2	4	6	1	3	5	NA	NA
Aggr. rate	2	4	8	8	8	4	4	2	2	8	NA	NA
t_n	40	60	50	10	25	45	70	20	30	100	NA	NA
v_n : max V	40	48.7	34.4	5.8	17.5	1.1	44.0	10.7	0.7	79.7	282.6	-4.6
v_n : max PF	40	48.7	34.4	5.8	17.8	12.2	9.5	8.8	20.4	27.7	225.3	-3.3

Table 7.3: Input parameters, demanded and obtained throughputs for the ten-node network.

Discussion

Contributions and limitations

This chapter presents our final Markovian model for IEEE 802.11 networks. The continuous time component makes Model C especially adapted to highly heterogeneous networks and increases its accuracy. By using continuous time, a node's output rate is the portion of time that node occupies the medium, a quantity from which we can easily derive the node's achieved throughput as shown in Chapter 2.

While the computational complexity of the model stays the same as for Model B, we decreased its conceptual complexity by implementing a close-form expression for the backoff factor and a straightforward computation of the nodes' achieved throughputs.

The detailed study of channel bonding in 802.11ac WLANs offers insight into the best practices to properly configure WLANs that use channel bonding. We also show the importance of choosing the right performance metric when configuring a network, as the optimal solutions may differ significantly from one metric to another. However, the task of developing definitive and general guidelines for channel bonding proves to be more complicated than we initially estimated and remains to be completed.

Possible improvements

This chapter concludes our work on Markovian models, however there are several extensions that, mainly due to time limitations, we were not able to complete. We briefly discuss several of those extensions here and leave the more detailed discussion for Chapter 9.

In all three models, we propose applications that are mostly brute-force based, i.e., we perform a search in the complete set of possible configurations. While this approach undoubtedly results in finding the optimal solution given a performance metric, its execution time is far from optimal. With regards to our application in channel bonding and assignment, we believe that there are smarter heuristics that we could implement in the future that would render our model of greater practical use.

Although our model's execution is fully automatized, its implementation is not optimized and could benefit from a significant speed-up if properly coded. Moreover, parallelization and dynamic programming (where we keep already known solutions in memory) could also help in reducing the cost of running our model, especially on large WLANs.

Finally, as we discussed in Section 4.1, our system description can sometimes be seen as limiting. Although throughout this thesis we made sure to constantly evolve our system, there are widely used WLAN mechanisms that we still have not considered. In our work we assume symmetric conflict graphs, ideal radio conditions, and only downlink traffic. In spite of having a future extension could, at least partially, alleviate these constraints by modifying the proposed model as well as the used simulation setup.

Chapter 8

Graph Signal Processing for 802.11 WLANs

Introduction

In the performance evaluation of networks, one can broadly identify at least two large groups of modeling approaches: constructive and descriptive models [85]. Constructive models use the knowledge of the system to reproduce its behavior, i.e., for a given set of input parameters, a constructive model provides an output similar to the real system by mimicking its internal mechanisms. Descriptive models suppose (almost) no knowledge of the internal system and simply require a dataset of input/output values from which they derive a model that describes the output as a function of the input. The availability of large datasets have made descriptive models popular in many different fields of study.

The modeling of WLANs is mostly done by networking experts who use their knowledge of the underlying system and, as a result, most of the models developed are of constructive nature, as shown in Chapter 3. Constructive models can be beneficial for both improving the network performance and for extending our understanding of that performance, as discussed in Section 1.2 of Chapter 1. Because constructive models explicitly take into account some of the network parameters, they can be easily used to study the impact of those parameters. However, a large drawback of constructive models is their complexity and, generally, their lack of scalability. The models reviewed in Chapter 3, as well as our Markovian models of Chapters 5, 6, and 7, all suffer from scalability issues as even conceptually simple models often demand high computational power.

Descriptive modeling, in most cases, completely surpasses the dimensionality curse by modeling the network as a black box to which we provide an input and from which we recover an output. In WLANs, some descriptive models are used in problems such as device localization, traffic classification, and security, and more rarely for performance evaluation. This chapter presents our last modeling approach that focuses on one such descriptive method inspired by the recent works in the field of Graph Signal Processing (GSP). To the best of our knowledge, GSP methods have so far never been applied to the performance modeling of WLANs.

Related Works

A detailed overview of current GSP methods is out of the scope of this thesis and of the author’s expertise. Instead, we provide several works that can guide an interested reader to a number of pioneering works in the field.

A growing part of the data available to us today has an underlying network structure, e.g., the friendship connections in a social network, the neural pathways in the brain, or the links in a communication network. GSP emerged as a solution to process the available data while accounting for its structure by representing the underlying structure as a graph and the data as a signal on that graph. The two pioneering papers [86, 87] of GSP were published only five years ago, making GSP a relatively new field of study. Shuman *et al.* [87] propose a spectral analysis approach using the graph’s Laplace matrix as the building block of GSP. From the Laplace matrix, they go on to define elementary operations including filtering, translation, and convolution. Sandryhaila and Moura [86] approach the subject from an algebraic signal processing perspective and develop the discrete signal processing of graphs (DSP_G) methodology. The authors provide the basis of DSP_G by defining the adjacency matrix as the *graph shift*, i.e., the graph signal equivalent of the time shift in time series. Using the graph shift they define graph filters, the graph Fourier transform, and show how their methods can be used for Linear Prediction (LP).

These two works established the groundwork for many following research questions. Sandryhaila and Moura adapted their methodology to Big Data analysis [88], where they use product graphs to increase the efficiency of DSP_G on Big data by lowering the arithmetic cost of the algorithms, as well as making them suitable for parallel and distributed implementations. Chen *et al.* [89] extend these works further and develop a sampling theory on DSP_G and show how their theory can be used for semi-supervised learning. There exists a large number of other ML applications using GSP, including denoising [90] and graph signal reconstruction [91], semi-supervised classification [92], or community detection and clustering on graphs [93, 94].

GSP for WLANs

Graph signal processing is an emerging field of research that translates the theory of classic Signal Processing (SP) to data that has an underlying structure represented as a graph. We use this section to first define the basic notions of GSP for WLANs and to then present some of the methods we designed especially for WLANs. The methods presented here are to a great extent inspired by the Discrete Signal Processing on Graphs (DSP_G) theory developed in [86].

Before diving into the methodologies we tested, we first wish to discuss the intuition and reasoning behind the idea of applying GSP to WLANs. The mere presence of an underlying structure in data does not mean that the data itself is impacted by that structure. Hence, before applying GSP methods to WLANs, we need to make sure that the structure is an intrinsic part of the network’s behavior. WLANs, as was shown in Chapters 1 and 3, are undeniably influenced by their topology. A node that has no neighbors is, in the general case, capable of achieving a higher throughput than a node that has to share the available resources with other

nodes. Moreover, we know that there are certain topologies, such as the three-node FIM network of Section 1.3 in Chapter 1, in which a node can experience great (dis)advantage because of its location in the topology. This chapter summarizes our efforts to reproduce the behavior of a WLAN using GSP methods.

Basic definitions: Graph and signal

There are two definitions that are essential to any GSP problem: the graph and the graph signal. A classic representation of a WLAN as a graph is the *adjacency matrix*, defined as:

$$\mathbf{A}_{n,m} = \begin{cases} 1, & \text{if } n \text{ and } m \text{ are neighbors} \\ 0, & \text{otherwise.} \end{cases} \quad (8.1)$$

Our definition of neighboring nodes is the one given in Section 4.1 of Chapter 4, i.e., our adjacency matrix is simply a matrix representation of the network's conflict graph.

The graph signal should be a quantity that is proper to each node in the graph and that can be measured on both input and output. Hence, appropriate signal values in our networks can be the input and output rate of a node (defined in detail in Section 4.1 of Chapter 4). We remind that the input (output) rate of a node is the normalized demanded (achieved) throughput of that node.

Having defined the graph and the graph signal, the question we want to answer is: Can we use GSP to predict the nodes' output rates given only their input rates, the conflict graph, and no other WLAN-related knowledge?

Moving average filters

Classical SP provides us with the tools needed to analyze and transform signals with the help of *signal filters*. A filter takes an input signal, transforms it, and then provides an output signal. The most basic filter is the *time shift*, which takes an input signal and delays it by one sample. In classic time series, the time shift is easily defined, as time is an intrinsic component of the series and by definition the sample at time t is preceded by the sample at time $t - 1$. Thus, the time shift at time t , \tilde{x}_t , is simply defined as $\tilde{x}_t = x_{t-1}$.

Our work on GSP for WLANs focuses on a single type of simple, yet powerful, linear filters: the Moving Average (MA) filters. In classic SP, an MA filter can be seen as a sum of weighted time shifts, as it computes the weighted average of the P past input of a time series to obtain the current output signal. By taking the average of a sequence of inputs and not solely a single value, we obtain a smoother function on output, i.e., we create a low pass filter. For example, the left-hand side of Fig. 8.1 shows a time series provided on input and then a time series obtained on output from some arbitrary MA filter. We can formally write the MA filter as:

$$y_t = \sum_{p=0}^P h_p x_{t-p}, \quad (8.2)$$

where y_t and x_t are the output and input signal at time t , P is the *filter order*, and h_p is the *filter coefficient* for order p . The filter order simply tells us how far

into the past input signals we need to look to construct the present output signal. The coefficients, usually summarized in the vector $\mathbf{h} = [h_0 \ h_1 \ h_2, \dots, h_P]$, define the importance of the input signal at a given order. For example, in SP we sometimes encounter filters where the coefficients are all equal to $\frac{1}{P}$ so that the output signal is the arithmetic mean of the last P input signals.

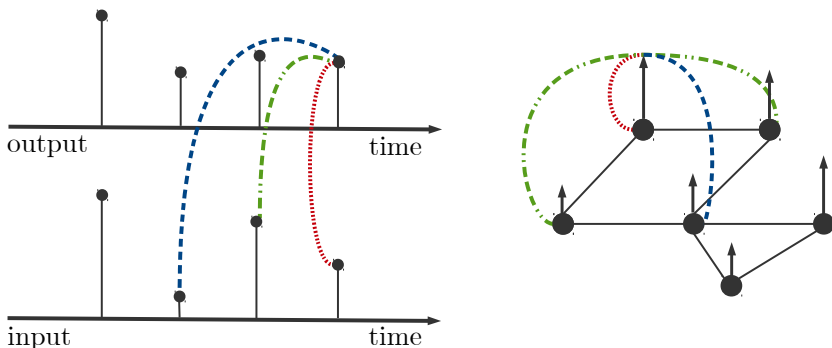


Figure 8.1: Classic signal processing (left) and GSP (right).

In classic SP, time is inherently ordered and we only need to decide how far back we need to look to calculate an output signal. In GSP, however, we replace the notion of time by the notion of space and we have to define how a node's signal propagates through the network. We do so by using a *shift operator*, denoted by \mathbf{S} , and slightly modifying Eq. (8.2) so that we obtain:

$$\mathbf{y} = \sum_{p=0}^P h_p \mathbf{S}^p \mathbf{x} , \quad (8.3)$$

where \mathbf{x} and \mathbf{y} denote the vectors of input and output rates, respectively.

One of the most common shift operators is the adjacency matrix, given in Eq. (8.1). The popularity of the adjacency matrix can, at least partially, be attributed to the fact that it allows for a quick diffusion of the signal in the network. At order p , the entry of $\mathbf{A}_{n,m}^p$ represents the number of different walks of length p from node n to node m . As a consequence, in a connected graph, the signal from one node can reach any other node when the order of the filter is at most the network's diameter.

The right-hand side of Fig. 8.1 depicts the propagation of a signal in a six-node network from the point of view of a single node. At order $p = 0$, the node only has information about its own input signal, depicted in red. At order $p = 1$, the signal from one-hop neighbors has reached the node, depicted in green, and so on. The figure simply summarizes the idea that as we elevate the orders of \mathbf{A} we reconstruct the classic mechanism of message flooding by reaching one-hop neighbors, two-hop neighbors, three-hop neighbors...

Our first attempts to model WLANs using GSP all revolved around using classic GSP methods. Thus, we tested a series of shift operators, such as the Laplace operator, before coming to the same conclusion as the authors of [73], and deciding that incorporating WLAN-specific knowledge into the model can be highly

beneficial. The next section introduces a series of shift operators we developed specifically for WLANs.

Adapting GSP to WLANs

Our adaptation of GSP methods for WLANs consists in modifying the shift operators and, to a lesser extent, the graph signal. From the different graph signals that we tested, we found that using the input/output rate as input/output signal provides the lowest errors and all following results are based on that signal definition. However, different shift operators have shown significantly varying results and we use this section to present the reasoning behind their development. It should be noted that we only present several selected shift operators that are (currently) the final result of an extensive trial and error learning process, as we believe that previous operators are merely a stepping stone to the present ones, just as the present ones will be to the future operators.

There are three main WLAN notions that we consider to be relevant to each node and that our homemade operators capture:

1. The number of neighboring cliques.
2. The number of neighboring nodes.
3. The nuisance value of every neighbor based on its particular traffic parameters.

To illustrate these issues, we begin by studying the two sample WLANs of Fig. 8.2. The node in light gray has four neighbors in both WLANs, however, our experience shows that its performance is notably different in the two topologies. In the left-hand side topology, the node is an aggravated version of the three-node FIM topology (see Section 1.3 in Chapter 1). In such topologies, the gray node experiences starvation as it has four nodes that can send frames independently of each other. As a result, the node rarely senses an idle medium that would allow it to send frames. In the right-hand side topology, i.e., a full-mesh network, all five nodes are essentially equal, as they can all detect each others' transmissions and thus share the channel in a round-robin fashion. The work on the 802.11 performance anomaly in [19] has shown that even in a full-mesh network, equity is only achieved in the number of medium accesses and not in the duration of medium occupation. This implies that the particular traffic characteristics of a neighbors will, to a great degree, define how inconvenient that neighbor is: a neighbor with a high input rate and high data rate might sometimes be less of a nuisance than a neighbor with a lower input rate but also a lower data rate.

In order to quantify these metrics, we propose two homemade operators based on modifications of the adjacency matrix: the **N**uisance **V**alue operator \mathbf{A}_{NV} , and the **C**liques and **N**eighbors operator \mathbf{A}_{CN} .

Nuisance value operator \mathbf{A}_{NV}

The nuisance value operator quantifies the impact of neighbors by taking into account each neighbor's specific traffic parameters. We already know that having a neighbor that is highly demanding, i.e., has a large input rate, can be of great

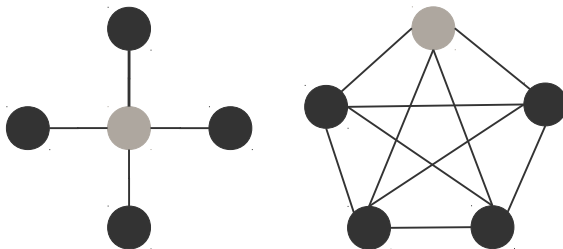


Figure 8.2: Conflict graphs of five-node networks.



Figure 8.3: Conflict graph of a two-node network.

nuisance. We also know that neighbors that use lower data rates can have a larger nuisance value than those using higher data rates, as they need a longer time to transmit their frames. In order to properly quantify the nuisance value of a neighbor, we tested a simple topology of only two nodes, shown in Fig. 8.3.

First, we define the nuisance value experienced by node 2 because of node 1, denoted by $\phi_{1,2}$. We calculate $\phi_{1,2}$ as the difference between the output rate node 2 can achieve when node 1 is not present in the network, i.e., $y_2 = x_2$, and the output rate node 2 achieves when node 1 is present. Figure 8.4 reports the results we found in simulation using ns3. Node 2 is always saturated, i.e., $x_2 = 1$, and has a fixed MCS index of MCS0 in Fig. 8.4a and MCS7 in Fig. 8.4b (the lowest and highest MCS indexes available in both 802.11n and 802.11ac). Node 1 varies its input rate in the $[0, 1]$ interval and also varies its MCS index from MCS0 to MCS7. We choose this type of variation for node 1 as we wish to discover how both the input rate and the data rate impact the neighborhood of node 1. We first notice that both figures represent two highly distinct patterns of behaviors, first a linear increase in the nuisance value and then a constant period during which the nuisance value has reached its maximum. In fact, we found that there are three distinct regimes of resource sharing. The first one, not represented in our figures, happens when $x_1 + x_2 < 1$ and in that case node 1 does not cause any nuisance to node 2 as they can both achieve their demanded throughputs and $\phi_{1,2} = 0$. The second regime starts when $x_1 + x_2 = 1$ and $\phi_{1,2}$ linearly increases from 0 until it reaches its approximated maximum value, denoted by $\Phi_{1,2}$:

$$\Phi_{1,2} = \frac{1}{\frac{1}{t_{1,\max}} + \frac{1}{t_{2,\max}}} . \quad (8.4)$$

Interestingly, the maximum nuisance value is achieved when node 1's input rate matches that value, i.e., when $x_1 = \Phi_{1,2}$. Any further increase in the input rate of node 1 will now change the nuisance value, representing the third and final regime of resource sharing.

A more detailed study of Fig. 8.4 reveals that the largest nuisance happens when node 2 has the highest MCS index of 7 and node 1 the lowest MCS index of

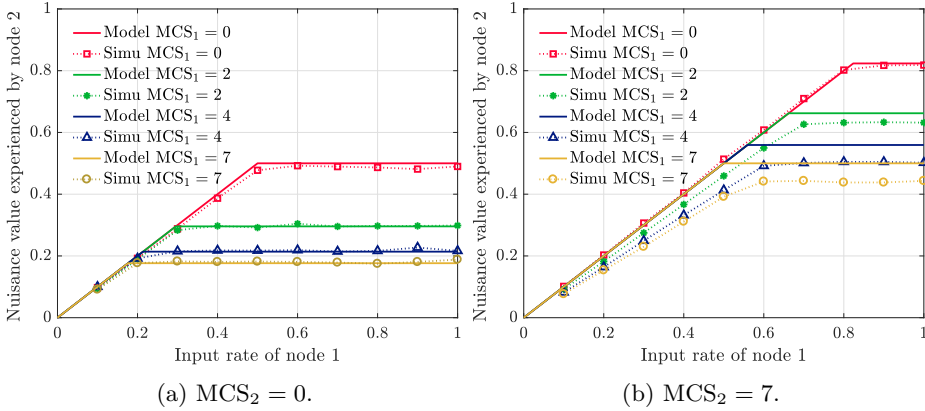


Figure 8.4: Nuisance value experienced by node 2 in Fig. 8.3 when node 1 is varying its input rate and MCS index.

0. This is the classic example of the 802.11 performance anomaly, and our results show that node 2 loses a little over 80% of the output rate it obtains when node 1 is not present. However, if node 1 is also using the slowest index MCS0, then it can at most lose 50% of its output rate.

These results encouraged us to try and implement the maximum nuisance value into our operator, \mathbf{A}_{NV} . Our idea is that the knowledge we have obtained on the two-node sample can be generalized to a network of any arbitrary size by calculating the nuisance value of any two nodes. We do so by defining the (n, m) -th entry of \mathbf{A}_{NV} as:

$$\mathbf{A}_{\text{NV}}(n, m) = \frac{\Phi_{m,n}}{\sum_{\ell=1}^N \Phi_{\ell,n}} \mathbf{A}(n, m), \quad (8.5)$$

where the multiplication of the nuisance value by the adjacency matrix allows us to keep the topological information and add to it the traffic information.

Cliques and neighbors operators \mathbf{A}_{CN}

A significant part of our work on 802.11 WLANs focuses on the importance of the number of neighbors a node has and the number of cliques to which it belongs. Our simulations have shown us a precise modeling of WLANs should include both values, as a node that belongs to many maximal cliques of two nodes or to a single maximal clique of many nodes can experience radically different resource sharing, as the two topologies of Fig. 8.2 have shown. To quantify these values, we introduce the matrix Ψ such that each entry on the main diagonal, $\Psi(n, n)$, represents the inverse of the sum of the number of neighbors and number of maximal cliques to which node n belongs:

$$\Psi(n, n) = \frac{1}{\text{neighbors of } n + \text{maximal cliques of } n}, \quad (8.6)$$

and all other entries of Ψ are zero.

We can now define our new shift operator as:

$$\mathbf{A}_{\text{CN}}^p = \begin{cases} \mathbf{A}^0, & \text{if } p = 0 \\ \mathbf{1}_{\mathbf{A}^p} \Psi^p, & \text{otherwise,} \end{cases} \quad (8.7)$$

where $\mathbf{1}_{\mathbf{A}^p}$ returns a matrix whose (n, m) -th entry is 1 if $\mathbf{A}^p(m, n) \neq 0$, and 0 otherwise. Similarly to the previous operator, the multiplication of these matrices allows us to keep both the topological information of \mathbf{A} and add to it node-specific information contained in Ψ . It should, however, be noted that from a purely SP standpoint, this new matrix does not fully respect the definition of an MA filter as the shift operator at every order is no longer simply a power of the initial shift operator. While this deviation from the standard definition can cause a loss of spectral interpretation for the obtained results, in no way does it harm the applicability to WLANs or reduces the validity of the presented results.

Finally, our last operator is a convex combination of \mathbf{A}_{NV} and \mathbf{A}_{CN} :

$$\delta \mathbf{A}_{\text{NV}} + (1 - \delta) \mathbf{A}_{\text{CN}}, \quad (8.8)$$

where $\delta \in [0, 1]$. The convex combination operator, originally proposed in [95], should allow us to integrate the knowledge about the network's topology as well as the neighborhood of a node and its traffic parameters. It should be noted that because it is a combination of two operators, this new operator will have twice as many coefficients for any given order P .

Model Validation

The validation section of this chapter is slightly different than the ones presented for the Markovian models. The GSP approach has a considerable learning phase for which we need to provide a large dataset. Moreover, the networks we consider need to be of several dozen nodes, so that the conflict graph is diverse enough to allow for a real learning of its behavior and avoid quickly stepping into overfitting. For these reasons, we decided to turn back to the ns2 simulator. The advantage of ns2 over ns3 is that results can be obtained much faster, as the simulator codes the WLAN in much less detail than its successor and in this initial phase of testing we can afford to gain some speed at the cost of system fidelity. The speediness of ns2 has allowed us to test the model on several networks and make sure that we obtain consistent errors. In the interest of brevity, we only present the results for two topologies that are representative of the general behavior: a 42-node network with an average node degree of 3 in Fig. 8.5a and a 41-node network with an average node degree of 2 in Fig. 8.5b.

Before presenting the results, we first define the network scenarios. We use the 802.11g standard with nodes generating frames of 1000B sent over links of 54Mbps. One of the main differences between our Markovian models and the GSP model is the fact that the GSP model has two distinct phases of learning and testing. In the learning phase, the model estimates the filter coefficients on a series of training samples. We then use a different set of test samples to assess the model's accuracy in predicting the nodes' output rates. More specifically, for each network, we generate 200 samples: 100 training and 100 testing samples. Every sample has its own set of input rates that are uniformly chosen in the $[0.2, 1]$ interval. All

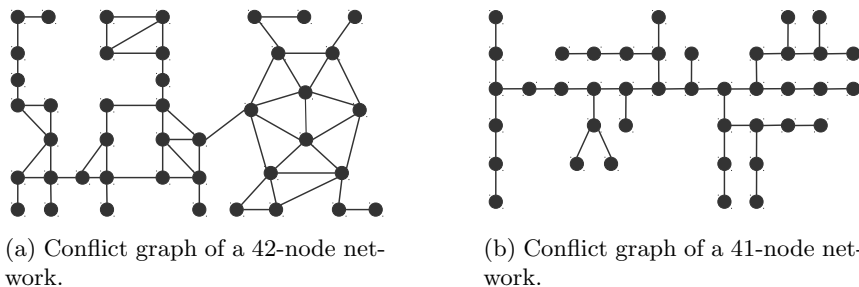


Figure 8.5: WLANs used for the model validation.

nodes are generating Poisson traffic whose rate is chosen so that it satisfies the node's input rate. Each training and test sample is the average of 10 simulation runs of 2 minutes each, so as to make sure that we obtain a representative value for the node's output rates. This means that in total we have $200 \times 10 = 2000$ simulation runs for any WLAN.

In the training phase, we first calculate the vector of filter coefficients, \mathbf{h} , for every sample in the training set. We then simply calculate the average over all the coefficients of the training phase to obtain the filter coefficients for the testing phase. To assess the accuracy of both the training and the testing phase, we calculate the mean squared error for the prediction made for every sample:

$$e = \sqrt{\frac{\sum_{i=1}^N (y_n - \tilde{y}_n)^2}{\sum_{i=1}^N y_n^2}}, \quad (8.9)$$

where \tilde{y}_n is the filter's prediction for node n 's output rate, y_n , provided by ns2.

Figures 8.6 and 8.7 present the mean squared error as a function of the filter order for four shift operators (\mathbf{A} , \mathbf{A}_{NV} , \mathbf{A}_{CN} , and $\mathbf{A}_{\text{NV}} + \mathbf{A}_{\text{CN}}$) in modeling (left) and prediction (right) for the 42-node and the 41-node network, respectively. We first notice that there is a notable similarity between the results provided by the operators \mathbf{A} and \mathbf{A}_{NV} on one side, and \mathbf{A}_{CN} and $\mathbf{A}_{\text{NV}} + \mathbf{A}_{\text{CN}}$ on the other. Moreover, the first two operators largely underperform, as they only achieve modeling errors of around 35% at order 15 for the 42-node network, and 30% at order 13 for the 41-node network. Given that all the nodes use the same data rate, it is no surprise that both matrices provide similar results, as their core information remains the same. In prediction (Fig. 8.7b and 8.6b) the mean squared error is systematically above 40% for both networks. These results lead us to believe that the proposed operator lacks information and is not able to capture the network's behavior even for filter orders as high as 15.

The cliques and neighbors operator, \mathbf{A}_{CN} , and the convex combination provide more promising results. In modeling, the mean squared error can be as low as 15% for the 42-node network and close to 10% for the 41-node network. As expected, the results in prediction are slightly less accurate, with relative errors little below 20%. When comparing the two operators one should be careful as the convex

combination, at any order, contains twice as many coefficients as the cliques and neighbors operator at the same order. However, this gives an ever larger advantage to the \mathbf{A}_{CN} shift operator, as it provides similar results to the convex combination operator with half the number of coefficients.

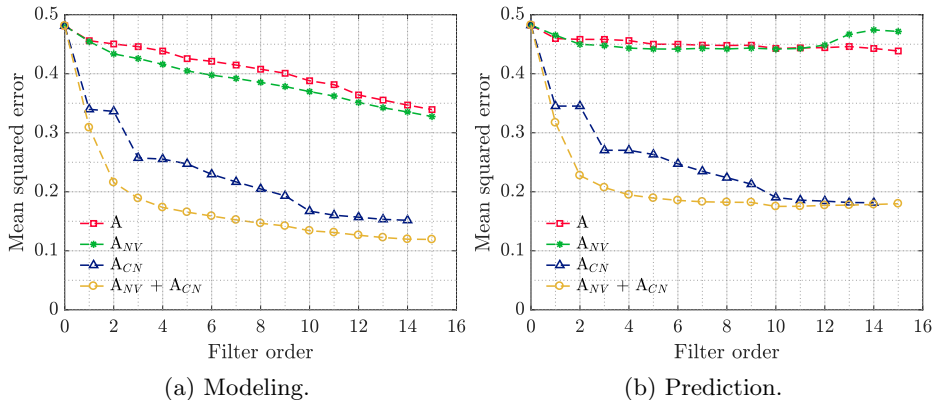


Figure 8.6: Results for the 42-node network of Fig. 8.5a.

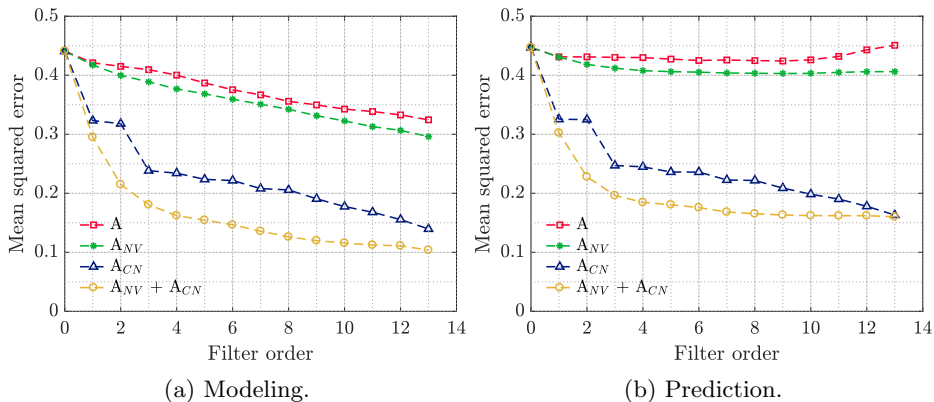


Figure 8.7: Results for the 41-node network of Fig. 8.5b.

Lastly, we want to make sure that the obtained error is not highly dependent on the size of our training/testing set. We compare the mean squared error as the number of samples varies between 1 and 150 for the 42-node network in Fig. 8.5a when using the \mathbf{A}_{CN} operator. Figure 8.8 shows these results. We first notice that even for small dataset sizes, around 25 samples, the errors in both modeling and prediction does not deviate more than 2 points between two consecutive sample sizes. As we increase the size of the dataset, the error quickly approaches its asymptotic value and we consider that in modeling it stabilizes around 50 samples, while in prediction we need 75 samples to make sure the results are coherent. In order to have a safety margin, we decided to use 100 samples in both modeling and prediction. Note that other networks have been tested and provide similar results.

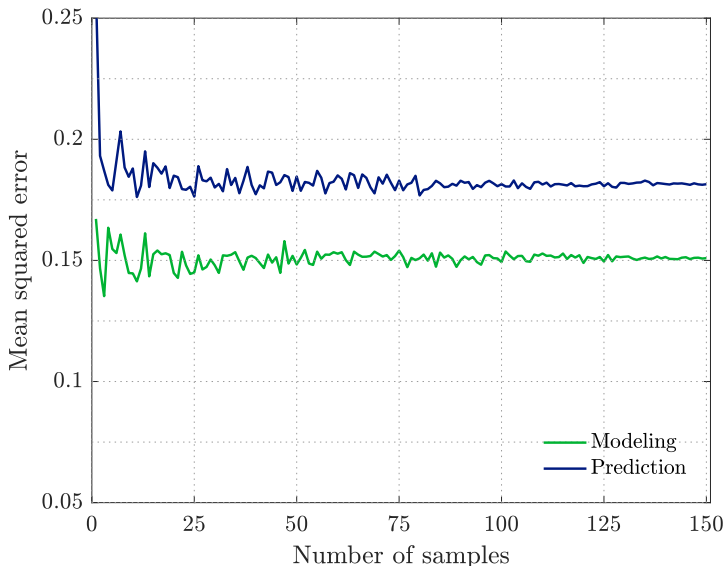


Figure 8.8: Mean squared error as a function of the number of samples in the training set.

While these results are of lower precision than the modeling approaches we presented before, there are several encouraging reasons:

- The prediction error does not rapidly grow with the increasing orders, meaning that the model is not overfitting to the network, i.e., the model learns a real behavior of the network and not simply reproducing every sample.
- We can do better than the adjacency matrix by using operators specifically designed for WLAN networks.
- The dimensionality curse is completely circumvented, as the most complex part in solving each filter lies in the inversion of a matrix that can be easily computed by today's computers.

Conclusions

This chapter represents our latest work, and as such it contains many aspects that need to be improved and/or completed. Specifically, due to time constraints on our last collaboration, the application was never fully implemented and tested. We use this concluding section to discuss one possible application and then several drawbacks and possible improvements to the proposed modeling approach.

Application

The most important advantage of the GSP approach lies in its scalability with the size of the WLAN. For this reason, we believe that such models can be used to

reconfigure already existing large networks in real time. One such application is the load balancing among the APs of a WLAN.

In a load balancing scenario, we envision a WLAN with a given number of APs each with one or more connected stations. As many of our examples have shown, the position of the AP inside the WLAN can highly impact its achieved throughput and result in the AP having a much lower achieved than demanded throughput. In reality, when an AP is experiencing starvation it means that its associated stations do not acquire all the throughput they demanded. In some scenarios, redistributing the demanded throughput of each AP by shuffling the associated stations among neighboring APs can result in either a much higher overall network throughput or in a more fair sharing of the available throughput among stations.

The GSP model can potentially be helpful in finding more optimal association schemes for the stations and APs. The fact that the model is highly scalable would allow us to adapt it to WLANs with dozens of APs and associated stations. Our idea is to include the stations into the model and representing the input rate of a node as the sum of the input rates of its associated stations. We can then try to detect the APs that are struggling the most to achieve their input rates and redistribute their load to neighboring APs by associating the stations with those APs. We imagine that our approach can at first be applied only to one AP at a time, before developing a more complete version in which the association process in the entire network is optimized.

Contributions and limitations

To the best of our knowledge, our work is the first effort to apply GSP methods to the modeling of WLANs. As such, it benefits from great novelty and gave us a lot of opportunities for exploration, but it also suffers from a lack of established and confirmed theoretical foundation.

The work presented in this chapter is fully based on a trial and error exploration during which we tested a large number of different shift operators and graph signals. An important upside of the model is that we managed to improve the original adjacency matrix filter and incorporate many WLAN-specific parameters. In the author's opinion, the most important part of this work is that it forces us to rethink what we know about WLANs. Developing new shift operators requires finding the set of parameters that are crucial to the current network behavior. The notions of cliques, neighbors, and data rates are merely a small subset of the parameters we have tested and of the parameters that can still be tested.

The computational simplicity of the approach itself, and not its development, is remarkable. Solving large WLANs comes down to the inversion of a matrix whose dimensions are of the order of the WLAN's size, an easy task for any modern computer.

There is a number of limitations of the approach that still need to be overcome. The prediction result of the filters is acceptable, with errors of around 20%, yet needs to be improved. Improving these errors will require working on new shift operators, and also possibly new graph signals.

It should be noted that even though our methods are adapted to heterogeneous networks, so far they have only been tested in WLANs where all APs have the same data rate and frame size. If there are new shift operators that offer higher

precision, the model would need to be tested not only on heterogeneous networks, but ideally in real WLAN testbeds.

Chapter 9

Conclusions

The three years of a thesis are paradoxically both too fast and too slow, since the last year goes by an order of magnitude faster than the first one. In hindsight of the thesis, we realize that its purpose was never to solve all the problems of WLAN performance evaluation, but to learn how to recognize and properly evaluate the important ones. This concluding chapter discusses those potential future works that, mostly due to time constraints, we did not prioritize and that I hope will one day be completed. Before addressing these improvements, we shortly state the contributions made thus far.

Contributions

The contributions of this thesis can be summarized into two types of WLAN models:

1. Constructive Markovian models for small to medium-sized WLANs with highly heterogeneous nodes.
2. Descriptive graph signal processing models for large and homogeneous WLANs.

Our three generations of Markovian models have an increasing system fidelity and a decreasing conceptual complexity. Since the beginning of this thesis, we were always focused on developing models that handle WLANs with two important features: unsaturated nodes and arbitrary network topologies. In order to incorporate these features and to have a model applicable to networks of more than a handful of nodes, we opted for a high-level description of the network. Over the course of the three years, and throughout this manuscript, this description has significantly evolved so that it more accurately represents today's WLANs. While comparing our most recent model, Model C in Chapter 7, with its oldest predecessor, model A of Chapter 5, the reader will notice that we have come a long way from the 802.11g WLAN whose nodes have homogeneous rates. Model C is adapted to the (currently) most recent 802.11ac standard amendment and to WLANs containing nodes with heterogeneous MCS indexes, channel bonding schemes, and frame aggregation rates.

The decreasing conceptual complexity is evident when comparing our three Markovian models: we avoid a series of complex modeling notions when transition-

ing from Model A to Model B, we then simplify several computations in Model C by replacing the discrete time Markov chains with their continuous time equivalents. As a result, the relative errors of the Model C's estimations are generally between 5% and 10% for networks of different sizes and complexities.

The development of our models has a strong application incentive. Hence, each modeling generation is presented with its subset of possible applications in the context of centrally-managed WLANs. These applications relate to optimizing the WLAN's performance with regards to throughput and/or fairness by choosing a better network configuration. We show, for example, how a centralized controller could use our model to choose the appropriate channel allocation given several different performance metrics. Using Model C, we scratch the surface of optimal channel bonding configuration and show that, even in largely simplified scenarios, the abundance of choices makes finding the optimal one an elaborate problem.

Our last model of Chapter 8 takes a radically different descriptive approach. The Markovian models are all of constructive nature, i.e., we used our networking expertise to replicate some essential mechanisms of the network. Our last model focuses on a Graph Signal Processing (GSP) method in which the network is modeled as an unknown black box to which we input a demanded throughput and from which we recover an obtained throughput. We use GSP techniques and homemade shift operators to try and replicate the WLAN's behavior using moving average filters.

The most important part of applying GSP to WLANs consisted in choosing the appropriate shift operator. While the existing literature on the topic provided us with a multitude of classic operators, such as the adjacency matrix or the Laplace operator, we quickly discovered that the modeling of WLANs most likely requires operators specifically designed for WLANs. The operators we developed try to implement knowledge about a node's topological position, i.e., number of cliques to which the node belongs or number of neighbors it has, and traffic characteristics, i.e., transmission capacity and packet size.

Currently, the GSP model provides relative errors of, at best, 15% that could potentially be improved using different WLAN-centric shift operators. Given that the computational complexity of the GSP model is much lower than that of our Markovian models, we envision applications where a global optimization of a larger WLAN (several dozens of nodes) is needed. This configuration could, for example, try to better redistribute the associated stations among nearby APs so that the traffic demand is more equilibrated.

Limitations and Possible Extensions

Our modeling approaches and the system they describe are based on a series of hypotheses that can be deemed too restrictive or unrealistic in some scenarios. It should be noted that most of these hypotheses can be relaxed in future extensions of our models, and we discuss only briefly some possible solutions.

Validation in simulation

Although the author is a keen supporter of simulation as a validation tool for analytical models (as discussed in Chapter 1), the lack of experimental validation

can objectively be seen as restrictive. Future improvements need to contain at least partial experimental validation of the proposed modeling approaches, either using in-house testbeds or the anechoic chamber testbeds already available for use. Using experimentation will also allow us to have a (more) realistic physical layer in our WLANs, as simulators have often been criticized about the fact that they implement somewhat idealized physical conditions.

Network of APs generating downlink traffic

For the sake of simplicity, in our simulations each AP has a single associated station to which it sends downlink traffic. The reasoning behind this assumption is that our models are mainly interested in how the APs are sharing the available resources, and not how each AP distributes the acquired resources among its associated stations. To make sure these assumptions were not too restrictive, we measured the real-life distribution of downlink vs. uplink traffic. We then simulated the impact of a corresponding amount of uplink traffic to conclude that it can be neglected at a reasonable loss of accuracy.

In future experimental works, the setup should contain WLANs with a single station per AP and WLANs with several stations per AP, as well as WLANs with and without uplink traffic. These kinds of tests will allow us to *i*) properly quantify the impact of stations and uplink traffic and *ii*) modify our models to accommodate the findings, if such modifications are deemed necessary.

Traffic

In our Markovian models and the system they consider (including simulation), our nodes are generating traffic using exponentially distributed On/Off periods. It is important to note that there is no perfect choice of traffic distribution, as its characteristics depend on too many factors and will result in completely different distributions for different WLANs. However, the On/Off traffic has the benefits of allowing us to arbitrarily define the average length of each period, thus allowing us to generate anything from a Poisson traffic to fully saturated sources. While On/Off traffic is not ideal, it is important to know that a large portion of streaming (non-elastic) traffic does have an On/Off pattern after the initial buffering stage [96,97]. Moreover, video traffic is expected to surpass 80% of all IP traffic by 2022 [6].

MIMO and beamforming

Throughout the manuscript we tried to show how our system evolved over the years: from an 802.11g with homogeneous data rates to an 802.11ac with frame aggregation, channel bonding, and different MCS indexes. A connoisseur of 802.11 standards will notice that there are still several technologies widely used by today's WLANs that we have never tested. Two such technologies are Multiple Input Multiple Output (MIMO) and beamforming. In MIMO, the AP can send data to several stations at a time using several antennas, or use multiple antennas to send data to a single station. Together with MIMO, beamforming can be used to increase the energy concentration in a single direction and isolate concurrent transmissions.

Something that MIMO and beamforming have in common is that they affect how the physical layer handles transmissions, yet the medium access procedure stays the same. We believe that in future works our validation setup should contain devices implementing MIMO and beamforming. In the modeling approach, these techniques will probably only require modifications in the conflict graph definition.

Brute-force application

Most of the applications we presented are based on a brute-force approach, i.e., testing all possible configurations to find the optimal one. Finding the optimal channel configuration for a four-node network with seven wireless channels results in only $7^4 = 2401$ channel allocations. However, let us assume the network size increases to ten APs, a completely reasonable size for an office WLAN, and the network switches to the 5Ghz frequency band where anywhere from 15 to 45 different channels are potentially available. The execution of the model on a ten node network is reasonably fast, yet testing the 15^{10} to 45^{10} possible allocations cannot result in a timely solution.

A part of our future works will focus on developing heuristics for choosing a subset containing only fairly logical allocations, and then using our model only on that subset to find the (sub)optimal solution. We believe that such heuristics are certainly feasible and will be a part of our upcoming works.

Performance Evaluation for WLANs: Concluding Remarks

General issues

In my modest experience, the performance evaluation of computer networks community has (at least) two major oversight: a profound lack of analytical models with experimental validation and a lack of reproducible results.

In the state of the art portion of this thesis, we reviewed about 20 different analytical models of WLANs developed over the last two decades. Out of those 20 works, only a single model was validated using experimental work and not solely simulation. I do realize that given the validation we provided in this thesis, we do not have much ground to stand on when stating that more experimental validation is needed. However, the community should be aware of the lack of overlap between theoretical and experimental works and try to remedy it. It is also my experience that substantial efforts are being made to encourage young researchers in performance evaluation to have a more hands-on experience with the available experimentation platforms, which I believe will encourage us to try and bridge the existing gap.

Major advancements have been made that contribute to more reproducible results, and we have personally started to contribute more to the community by including the simulation scripts and the code for the modeling approach in our last paper submission. Reproducing analytical results often requires recoding the approach from scratch when the original code is not available. The same is true for simulation studies, where often some details are provided and we can reproduce similar results, yet the lack of simulation scripts and system information makes it

nearly impossible to obtain the exact same values. Lastly, experimental studies suffer the most from reproducibility, as there is an intrinsic irreproducibility tied to the hardware in use. The problem often arises not from the lack of details about the setup, but from the instability of the used material, e.g., a change in driver or firmware on the same device can result in different results.

Are (our) analytical models still relevant?

The rapid development of 802.11 standards quickly renders some performance evaluation approaches irrelevant, forcing us to ask whether the models we have developed so far will have a practical use in a decade?

There are two especially encouraging points for WLAN performance evaluation: the quantity of Internet traffic is in constant increase largely due to the mobile and wireless environments, and the densification of WLANs makes them more and more centralized. In such conditions, performance evaluation helps centralized controllers to find optimal solutions given a performance metric.

Our Markovian models are adapted to all IEEE 802.11 versions up to, and excluding, the upcoming 802.11ax. For a long time we were able to avoid changing the basics of our modeling approach, as our approximation of the conflict graph and the medium access remained almost intact in several generations of amendments. The 802.11ax introduces two notions that will inevitably change how conflicts are managed and how nodes access the medium: BSS coloring and OFDMA. Without going into the details of the two techniques, we shortly describe the general ideas. BSS coloring will allow nodes on the same channel, but in different Basic Service Sets (BSSs) to access the channel at the same time by using different colors for different BSSs. OFDMA (Orthogonal Frequency Division Multiple Access) allows neighboring nodes to simultaneously transmit on the same channel using different sub-carriers.

BSS coloring and OFDMA render our current Markovian models unadapted to 802.11ax. Nevertheless, I personally find it encouraging that 802.11ax forces us to rethink the notions of conflicts and medium sharing, as centralized control and efficient algorithms will be needed more than ever in tomorrow's denser networks. In my opinion, analytical models will become increasingly more important as future WLANs become increasingly more complex. It is up to the performance evaluation community to reassess existing and conceive new models, simulations, and experiments for tomorrow's WLANs.

Bibliography

- [1] Class Central. By The Numbers: MOOCs in 2018. <https://www.classcentral.com/report/mooc-stats-2018/>.
- [2] SM Riazul Islam, Daehan Kwak, MD Humaun Kabir, Mahmud Hossain, and Kyung-Sup Kwak. The Internet of Things for Health Care: a Comprehensive Survey. *IEEE Access*, 2015.
- [3] Joseph Romm, Arthur Rosenfeld, and Susan Herrmann. The Internet Economy and Global Warming. *The Center for Energy and Climate Solutions*, 1999.
- [4] Climate Care Org. The Carbon Footprint of the Internet. <https://climatecare.org/infographic-the-carbon-footprint-of-the-internet/>.
- [5] GSM Association. The Mobile Economy. 2019.
- [6] VNI Cisco. Cisco Visual Networking Index: Forecast and Trends, 2017–2022. *White Paper*, 2018.
- [7] Apurv Bhartia, Bo Chen, Feng Wang, Derrick Pallas, Raluca Musaloiu-E, Ted Tsung-Te Lai, and Hao Ma. Measurement-Based, Practical Techniques to Improve 802.11 ac Performance. In *Proceedings of the 2017 Internet Measurement Conference*. ACM, 2017.
- [8] *Aruba Networks*. <https://www.arubanetworks.com/>.
- [9] *Cisco Wireless Systems- Mobility*. <https://www.cisco.com/c/en/us/products/wireless/index.html>.
- [10] *Cisco Meraki*. <https://meraki.cisco.com/>.
- [11] *Meru Networks*. <http://www.merunetworks.com/>.
- [12] Krzysztof Pawlikowski, H-DJ Jeong, and J-SR Lee. On Credibility of Simulation Studies of Telecommunication Networks. *Communications magazine*, 2002.
- [13] Ramon dos Reis Fontes, Mohamed Mahfoudi, Walid Dabbous, Thierry Turetletti, and Christian Rothenberg. How Far Can We Go? Towards Realistic Software-Defined Wireless Networking Experiments. *The Computer Journal*, 2017.

- [14] HyungJune Lee, Alberto Cerpa, and Philip Levis. Improving Wireless Simulation Through Noise Modeling. In *Proceedings of the 6th international conference on Information processing in sensor networks*. ACM, 2007.
- [15] Guillaume Kremer, Philippe Owezarski, and Pascal Berthou. Cross Fertilization Between Wireless Testbeds and NS-3 Simulation Models. In *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2017.
- [16] Svilen Ivanov, André Herms, and Georg Lukas. Experimental validation of the ns-2 wireless model using simulation, emulation, and real network. In *Communication in Distributed Systems-15. ITG/GI Symposium*. VDE, 2007.
- [17] Giuseppe Bianchi, Antonio Di Stefano, Costantino Giaconia, Luca Scalia, Giovanni Terrazzino, and Ilenia Tinnirello. Experimental assessment of the back-off behavior of commercial iee 802.11 b network cards. In *INFOCOM*. IEEE, 2007.
- [18] Pierre Brunisholz, Franck Rousseau, and Andrzej Duda. Anatomie des Retransmissions dans les Implémentations de 802.11. In *Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication*, 2018.
- [19] Martin Heusse, Franck Rousseau, Gilles Berger-Sabbatel, and Andrzej Duda. Performance Anomaly of 802.11 b. In *INFOCOM*. IEEE, 2003.
- [20] Claude Chaudet, Isabelle Guérin Lassous, Eric Thierry, and Bruno Gaujal. Study of the Impact of Asymmetry and Carrier Sense Mechanism in IEEE 802.11 Multi-Hop Networks Through a Basic Case. In *Proceedings of the 1st ACM international workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*. ACM, 2004.
- [21] Marija Stojanova, Thomas Begin, and Anthony Busson. Conflict Graph-Based Markovian Model to Estimate Throughput in Unsaturated IEEE 802.11 Networks. In *IEEE WiOpt'17*, 2017.
- [22] Marija Stojanova, Thomas Begin, and Anthony Busson. Conflict Graph-Based Model for IEEE 802.11 Networks: A Divide-and-Conquer Approach. *Performance Evaluation*, 2019.
- [23] Marija Stojanova, Thomas Begin, and Paulo Gonçalves. Traitement du Signal sur Graphe Pour Modéliser les WLANs. In *Gretsi*.
- [24] IEEE Computer Society LAN MAN Standards Committee and others. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Standard 802.11-1997*, 1997.
- [25] IEEE Computer Society LAN MAN Standards Committee and others. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Standard 802.11-2016*, 2016.
- [26] Boris Ginzburg and Alex Kesselman. Performance Analysis of A-MPDU and A-MSDU Aggregation in IEEE 802.11 n. In *Sarnoff symposium*. IEEE, 2007.

- [27] Federico Cali, Marco Conti, and Enrico Gregori. IEEE 802.11 Wireless LAN: Capacity Analysis and Protocol Enhancement. In *INFOCOM*. IEEE, 1998.
- [28] Giuseppe Bianchi. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *Journal on selected areas in communications*, 2000.
- [29] Katarzyna Kosek-Szott. A Comprehensive Analysis of IEEE 802.11 DCF Heterogeneous Traffic Sources. *Ad Hoc Networks*, 2014.
- [30] Neeraj Gupta and CS Rai. New Analytical Model for Non-Saturation Throughput Analysis of IEEE 802.11 DCF. In *International Conference on Advances in Communication, Network and Computing*, 2014.
- [31] Emad Felemban and Eylem Ekici. Single Hop IEEE 802.11 DCF Analysis Revisited: Accurate Modeling of Channel Access Delay and Throughput For Saturated and Unsaturated Traffic Cases. *Transactions on Wireless Communications*, 2011.
- [32] Jinsung Lee, Hojin Lee, Yung Yi, Song Chong, Edward W Knightly, and Mung Chiang. Making 802.11 DCF Near-Optimal: Design, Implementation, and Evaluation. *IEEE/ACM Transactions on Networking*, 2016.
- [33] Emma Fitzgerald, Ulf Körner, and Bjorn Landfeldt. An Analytic Model for Throughput Optimal Distributed Coordination Function (TO-DCF). *Telecommunication Systems*, 2017.
- [34] Zhefu Shi, Cory Beard, and Ken Mitchell. Analytical Models for Understanding Space, Backoff, and Flow Correlation in CSMA Wireless Networks. *Wireless networks*, 2013.
- [35] Thomas Begin, Bruno Baynat, Isabelle Guérin Lassous, and Thiago Abreu. Performance Analysis of Multi-Hop Flows in IEEE 802.11 Networks: A Flexible and Accurate Modeling Framework. *Performance Evaluation*, 2016.
- [36] Bruno Nardelli and Edward W Knightly. Closed-Form Throughput Expressions for CSMA Networks with Collisions and Hidden Terminals. In *INFOCOM*. IEEE, 2012.
- [37] Mathilde Durvy, Olivier Dousse, and Patrick Thiran. Self-Organization Properties of CSMA/CA Systems and Their Consequences on Fairness. *Transactions on Information Theory*, 2009.
- [38] Libin Jiang and Jean Walrand. A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks. *IEEE/ACM Transactions on Networking*, 2010.
- [39] Robert Boorstyn, Aaron Kershenbaum, Basil Maglaris, and Veli Sahin. Throughput Analysis in Multihop CSMA Packet Radio Networks. *Transactions on Communications*, 1987.
- [40] Xin Wang and Koushik Kar. Throughput Modelling and Fairness Issues in CSMA/CA Based Ad-Hoc Networks. In *INFOCOM*. IEEE, 2005.

- [41] Bertrand Ducourthial, Stéphane Mottelet, and Anthony Busson. Improving Fairness Between Close Wi-Fi Access points. *Journal of Network and Computer Applications*, 2017.
- [42] Caihong Kai and Shengli Zhang. Throughput Analysis of CSMA Wireless Networks with Finite Offered-Load. In *International Conference on Communications*. IEEE, 2013.
- [43] Rafael Laufer and Leonard Kleinrock. The Capacity of Wireless CSMA/CA Networks. *Transactions on Networking*, 2015.
- [44] Thomas Bonald and Mathieu Feuillet. Performance of CSMA in Multi-Channel Wireless Networks. *Queueing Systems*, 2012.
- [45] Eldad Perahia and Robert Stacey. *Next Generation Wireless LANs: 802.11 n and 802.11 ac*. Cambridge University Press, 2013.
- [46] IEEE 802.11 p Working Group et al. IEEE Standard for Information Technology-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput. *IEEE Std; IEEE: Piscataway, NJ, USA*, 2010.
- [47] IEEE Computer Society LAN MAN Standards Committee and others. IEEE P802.11ac. Specification framework for TGac. *IEEE 802.11-09/0992r21*, January 2011.
- [48] Yuxia Lin and Vincent WS Wong. Frame Aggregation and Optimal Frame Size Adaptation for IEEE 802.11 n WLANs. In *Globecom*. IEEE, 2006.
- [49] Eng Hwee Ong, Jarkko Knecht, Olli Alanen, Zheng Chang, Toni Huovinen, and Timo Nihtilä. IEEE 802.11 ac: Enhancements for Very High Throughput WLANs. In *22nd International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 2011.
- [50] Jiyoung Cha, Hu Jin, Bang Chul Jung, and Dan Keun Sung. Performance Comparison of Downlink User Multiplexing Schemes in IEEE 802.11 ac: Multi-user MIMO vs. Frame Aggregation. In *WCNC*. IEEE, 2012.
- [51] Katarzyna Kosek-Szott. Improving DL-MU-MIMO Performance in IEEE 802.11 ac Networks Through Decoupled Scheduling. *Wireless Networks*, 2018.
- [52] Mihaela-Diana Dianu, Janne Riihijärvi, and Marina Petrova. Measurement-Based Study of the Performance of IEEE 802.11 ac in an Indoor Environment. In *ICC*. IEEE, 2014.
- [53] Konstantinos Pelechrinis, Theodoros Salonidis, Henrik Lundgren, and Nitin Vaidya. Experimental Characterization of 802.11 n Link Quality at High Rates. In *Proceedings of the fifth ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*. ACM, 2010.
- [54] Yunze Zeng, Parth H Pathak, and Prasant Mohapatra. Throughput, Energy Efficiency and Interference Characterisation of 802.11 ac. *Transactions on Emerging Telecommunications Technologies*, 2017.

- [55] Lito Kriara, Edgar Costa Molero, and Thomas R Gross. Evaluating 802.11 ac Features in Indoor WLAN: an Empirical Study of Performance and Fairness. In *Proceedings of the Tenth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation, and Characterization*. ACM, 2016.
- [56] Syed Hashim Raza Bukhari, Mubashir Husain Rehmani, and Sajid Siraj. A Survey of Channel Bonding for Wireless Networks and Guidelines of Channel Bonding for Futuristic Cognitive Radio Sensor Networks. *Communications Surveys & Tutorials*, 2015.
- [57] Mun-Suk Kim, Tanguy Ropitault, Sukyoung Lee, and Nada Golmie. A Throughput Study for Channel Bonding in IEEE 802.11 ac Networks. *IEEE Communications Letters*, 2017.
- [58] Sree Vasthav SV, S Srikanth, and Venkatesh Ramaiyan. Performance Analysis of an IEEE 802.11 ac WLAN With Dynamic Bandwidth Channel Access. In *National Conference on Communication*. IEEE, 2016.
- [59] Boris Bellalta, Alessandro Checco, Alessandro Zocca, and Jaume Barcelo. On the Interactions Between Multiple Overlapping WLANs Using Channel Bonding. *Transactions on Vehicular Technology*, 2015.
- [60] Sergio Barrachina-Muñoz, Francesc Wilhelmi, and Boris Bellalta. Dynamic Channel Bonding in Spatially Distributed High-Density WLANs. *arXiv preprint arXiv:1801.00594*, 2018.
- [61] Lara Deek, Eduard Garcia-Villegas, Elizabeth Belding, Sung-Ju Lee, and Kevin Almeroth. Intelligent Channel Bonding in 802.11 n WLANs. *IEEE Transactions on Mobile Computing*, 2014.
- [62] Ljiljana Simić, Janne Riihijärvi, and Petri Mähönen. Measurement Study of IEEE 802.11 ac Wi-Fi Performance in High Density Indoor Deployments: Are Wider Channels Always Better? In *IEEE WoWMoM*, 2017.
- [63] IEEE 802.11h Working Group et al. IEEE Std 802.11h-2003 Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Amendment 5: Spectrum and Transmit Power Management Extensions in the 5 GHz Band in Europe. *IEEE Std; IEEE: Piscataway, NJ, USA*, 2003.
- [64] NS Ravindranath, Inder Singh, Ajay Prasad, and VS Rao. Performance Evaluation of IEEE 802.11 ac and 802.11 n Using NS3. *Indian Journal of Science and Technology*, 2016.
- [65] Saber Malekmohammadi. Parameterizing Enterprise WiFi Networks: The Use of Wide Channels. Master’s thesis, University of Waterloo, 2019.
- [66] Raouf Boutaba, Mohammad A Salahuddin, Noura Limam, Sara Ayoubi, Nashid Shahriar, Felipe Estrada-Solano, and Oscar M Caicedo. A Comprehensive Survey on Machine Learning for Networking: Evolution, Applications and Research Opportunities. *Journal of Internet Services and Applications*, 2018.

- [67] Justin Manweiler, Naveen Santhapuri, Romit Roy Choudhury, and Srihari Nelakuditi. Predicting Length of Stay at WiFi Hotspots. In *INFOCOM*. IEEE, 2013.
- [68] Antonio J Ruiz-Ruiz, Henrik Blunck, Thor S Prentow, Allan Stisen, and Mikkel B Kjærgaard. Analysis Methods for Extracting Knowledge from Large-Scale WiFi Monitoring to Inform Building Facility Planning. In *International Conference on Pervasive Computing and Communications*. IEEE, 2014.
- [69] Siegfried Rasthofer, Steven Arzt, and Eric Bodden. A Machine-Learning Approach for Classifying and Categorizing Android Sources and Sinks. In *NDSS*. Citeseer, 2014.
- [70] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Stefanos Gritzalis. Intrusion Detection in 802.11 networks: Empirical Evaluation of Threats and a Public Dataset. *Communications Surveys & Tutorials*, 2015.
- [71] Sinno Jialin Pan, James T Kwok, Qiang Yang, and Jeffrey Junfeng Pan. Adaptive Localization in a Dynamic WiFi Environment Through Multi-View Learning. In *AAAI*, 2007.
- [72] Han Zou, Hao Jiang, Xiaoxuan Lu, and Lihua Xie. An Online Sequential Extreme Learning Machine Approach to WiFi Based Indoor Positioning. In *World Forum on Internet of Things*. IEEE, 2014.
- [73] Carlos Figuera, José Luis Rojo-Álvarez, Mark Wilby, Inmaculada Mora-Jiménez, and Antonio J Caamaño. Advanced Support Vector Machines for 802.11 Indoor Location. *Signal Processing*, 2012.
- [74] Jean-Gabriel Krieg, Gentian Jakllari, Hadrien Toma, and Andre-Luc Beylot. Acrux: Indoor Localization Without Strings. In *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2017.
- [75] Arpit Gupta, Jeongki Min, and Injong Rhee. WiFox: Scaling WiFi Performance for Large Audience Environments. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*. ACM, 2012.
- [76] Francesco Malandrino, Carla-Fabiana Chiasserini, and Scott Kirkpatrick. Cellular Network Traces Towards 5G: Usage, Analysis and Generation. *Transactions on Mobile Computing*, 2018.
- [77] *The Network Simulator ns2*. <http://www.isi.edu/nsnam/ns/>.
- [78] Rajendra K Jain, Dah-Ming W Chiu, and William R Hawe. A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer System. *Digital Equipment Corporation*, 1984.
- [79] *tshark: Terminal-Based Wireshark*. https://www.wireshark.org/docs/wsug_html_chunked/AppToolstshark.html.
- [80] *The Network Simulator ns3*. <https://www.nsnam.org/>.

- [81] Surachai Chiochan, Ekram Hossain, and Jeffrey Diamond. Channel Assignment Schemes for Infrastructure-Based 802.11 WLANs: A Survey. *Communications Surveys & Tutorials*, 2010.
- [82] Chunyi Peng, Haitao Zheng, and Ben Y Zhao. Utilization and Fairness in Spectrum Assignment for Opportunistic Spectrum Access. *Mobile Networks and Applications*, 2006.
- [83] Michele Garetto, Theodoros Salonidis, Edward W Knightly, et al. Modeling Per-Flow Throughput and Capturing Starvation in CSMA Multi-Hop Wireless Networks. In *INFOCOM*, 2006.
- [84] Mor Harchol-Balter. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge University Press, 2013.
- [85] Kavé Salamatian and Serge Fdida. A framework for interpreting measurement over internet. In *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, pages 87–94. ACM, 2003.
- [86] Aliaksei Sandryhaila and José MF Moura. Discrete Signal Processing on Graphs. *Transactions on signal processing*, 2013.
- [87] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *Signal Processing Magazine*, 2013.
- [88] Aliaksei Sandryhaila and Jose MF Moura. Big Data Analysis with Signal Processing on Graphs: Representation and Processing of Massive Data Sets with Irregular Structure. *Signal Processing Magazine*, 2014.
- [89] Siheng Chen, Rohan Varma, Aliaksei Sandryhaila, and Jelena Kovačević. Discrete Signal Processing on Graphs: Sampling Theory. *transactions on signal processing*, 2015.
- [90] Sunil K Narang and Antonio Ortega. Perfect Reconstruction Two-Channel Wavelet Filter Banks for Graph Structured Data. *Transactions on Signal Processing*, 2012.
- [91] Siheng Chen, Aliaksei Sandryhaila, José MF Moura, and Jelena Kovacevic. Signal Recovery on Graphs: Variation Minimization. *IEEE Transactions on Signal Processing*, 2015.
- [92] Venkatesan N Ekambaram, Giulia Fanti, Babak Ayazifar, and Kannan Ramchandran. Wavelet-Regularized Graph Semi-Supervised Learning. In *Global Conference on Signal and Information Processing*. IEEE, 2013.
- [93] Nicolas Tremblay and Pierre Borgnat. Graph Wavelets for Multiscale Community Mining. *Transactions Signal Processing*, 2014.
- [94] Xiaowen Dong, Pascal Frossard, Pierre Vandergheynst, and Nikolai Nefedov. Clustering on Multi-Layer Graphs Via Subspace Analysis on Grassmann Manifolds. *Transactions on Signal Processing*, 2014.

- [95] Fei Hua, Cédric Richard, Haiyan Chen, Jie Wang, Pierre Borgnat, and Paulo Gonçalves. Designing Convex Combination of Graph Filter. *Signal Processing Letters (preprint)*, 2019.
- [96] Shane Alcock and Richard Nelson. Application Flow Control in YouTube Video Streams. *ACM SIGCOMM Computer Communication Review*, 2011.
- [97] Ashwin Rao, Arnaud Legout, Yeon-sup Lim, Don Towsley, Chadi Barakat, and Walid Dabbous. Network Characteristics of Video Streaming Traffic. In *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*. ACM, 2011.