Contents

1 Introduction **8**

1.1 Importance of the topic 8

1.2 Limitations of existing approaches 9

1.3 Research gap 10

1.4 Contribution and significance of the study 10

2 Literature Review **12**

2.1 Overview of IEEE 802.11 standard 13

2.1.1 Transmission medium 13

2.1.2 Frequency band 13

2.1.3 Signal quality and its metrics 14

2.1.4 Radio Resource Management 16

2.2 Radio Resource Management in IEEE 802.11 17

2.2.1 IEEE 802.11h 18

2.2.2 IEEE 802.11k 19

2.2.3 IEEE 802.11ax 19

2.3 Previous Works 20

2.3.1 Radio Resource Management Approaches 21

2.3.2 Mathematical Models for Radio Resource Management . 24

2.4 Proprietary RRM Solutions 27
2.4.1 Cisco 27
2.4.2 Juniper Networks 29
2.4.3 Ruckus Networks 30
2.4.4 Aruba Networks 30
2.5 Conclusion 34

3 Baseline Solution Analysis and Algorithm Design 37
3.1 Problem statement 37
3.2 Baseline solution analysis — RRMGreedy 38
3.2.1 RRMGreedy: flow description 38
3.2.2 Evaluating asymptotic complexity of RRMGreedy . . . 46
3.2.3 RRMGreedy: limitations and conclusions 47
3.3 RRMGreedy++ design 48
3.4 Conclusion 53

4 Implementation and Evaluation 54
4.1 Simulation methodology 55
4.1.1 RRM support in ns-3 55
4.1.2 Providing scanning capabilities in ns-3 56
4.2 Implementation 58
4.2.1 RRMGreedy 58
4.2.2 RRMGreedy++ 59
4.3 Evaluation 59
4.3.1 Simulation settings 59
4.3.2 Metrics 62
4.3.3 Results 63

4.4 Conclusion 65

5 Discussion and Conclusion 66

5.1 Simulation Results 66

5.2 Limitations 66

5.3 Further Work 67

5.4 Conclusion 68

Bibliography Cited 69

A Used Terms and Definitions 76

List of Tables

I Simulation Parameters 60

II Test Case Configurations 60

III Gathered Metrics 62

IV Simulation Results 63

V Used Terms and Definitions 76

List of Algorithms

1 RRMGreedy 41
2 PreprocessRRMData 42
3 ChannelInterference 43
4 OnIfaceInterference 44
5 FromIfaceInterference 45
6 ChannelInterference++ 50
7 OnIfaceInterference++ 51
8 RRMGreedy++ 52

Abstract

As IEEE 802.11 Wireless Local Area Networks (WLANs), also known as Wi-Fi, become ubiquitous, the increasing density of WLAN deployments leads to congestion of frequency bands and performance degradation due to interference. Thus, Radio Resource Management (RRM) algorithms for mitigating these problems are in demand, especially for dense WLAN deployments.

Although there are several commercial RRM solutions, their implementation details remain undisclosed. Moreover, such solutions are incompatible between devices produced by different vendors. At the same time, most of the previous studies have considerable obstacles for production usage.

In this study, I perform a theoretical analysis of *RRMGreedy*, a centralized RRM algorithm used by a major Russian telecommunications vendor. I show that *RRMGreedy* possess a number of flaws and tends to yield suboptimal results. After identifying flaws of the algorithm, I propose several improvements for *RRMGreedy*. The updated version of the algorithm is referred to as *RRMGreedy++*. This algorithm adjusts frequency channel and transmission power parameters of access points based on physical-layer metrics, and uses WLAN Controller (WLC) as a central entity to gather data and perform computations.

Furthermore, this study develops a novel framework for the ns-3 network simulator that is used to implement *RRMGreedy++* together with other RRM algorithms. The simulation results demonstrate that *RRMGreedy++* achieves up to a 12% higher throughput on 2.4 GHz IEEE 802.11n networks and is capable of improving signal quality, measured in SNR (Signal-to-Noise Ratio) by 10%.

Chapter 1

Introduction

I Importance of the topic

Wireless local area networks (WLANs) that implement the IEEE 802.11 standards, commonly known as Wi-Fi, have become an increasingly popular solution for last-mile Internet access with a diverse population of users, from home Wi-Fi routers to large campus and city-scale WLANs with coverage areas reaching several square kilometers. As a result, the density of the Wi-Fi network increases, so the frequency band allocated for 802.11 networks becomes more congested, leading to interference and signal cancellation between different WLANs, resulting in network performance degradation. In addition, other appliances, operating on frequencies that overlap with the Wi-Fi band, affect the performance of WLANs. To meet the current bandwidth and latency expectations of modern network applications, such as video streaming, cloud computing, and video conferencing, the wireless network must be able to provide sufficient capacity to all clients. In this light, proper radio resource management becomes crucial for operating wireless networks.

The problem of managing radio resources is extensively studied in the context of **cellular networks**, which are characterized by extensive frequency reuse, a large number of clients, and large coverage areas spanning multiple kilometers, so proper spectrum management is vital for operation of cells. This problem is broken down to the following: given a set of access points (or base stations in cellular terminology) $\boldsymbol{B}$, which can communicate over a set of channels $\boldsymbol{C}$, with a maximum transmit power of P_{max}, establish a radio link between a client device and an access point by assigning it a triplet (b, c, p), where $b \in \boldsymbol{B}, c \in \boldsymbol{C}, p \leq P_{max}$. Essentially, RRM algorithms aim to provide such assignments that maximize the overall network performance.

II Limitations of existing approaches

A growing demand for RRM solutions, especially for large enterprise-grade multi-AP WLAN deployments, has led to the development of multiple commercial solutions, such as Cisco RRM [1], Aruba Adaptive Radio Management (ARM) [2], and others. However, existing solutions offered by major vendors are proprietary, so their source code, algorithms used, and operation details are not disclosed. At the same time, most studies on RRM have focused only on cellular networks, while RRM in 802.11 networks received much less attention. Many existing works on RRM in 802.11 networks have applicability problems, since real-world hardware poses constraints on what metrics can be retrieved from the wireless interface and which physical and link-level parameters can be adjusted.

III Research gap

Therefore, I can identify a research gap: the need for an open-source RRM algorithm that matches the performance of proprietary solutions and is feasible for real-world WLAN deployments.

This study aims to fill this gap by proposing a new RRM algorithm that, unlike existing market solutions, is publicly available and at least as effective while being practical and applicable to real-world WLAN deployments.

IV Contribution and significance of the study

By analyzing theoretical works on 802.11 RRM, I came up with a suitable optimization approach that combines Adaptive Channel Selection (ACS) and Transmit Power Control (TPC) techniques to improve radio frequency (RF) spectrum utilization, and channel reuse, which in turn leads to improved capacity of a wireless network.

The RRM approach that I use adheres to the centralized architecture, introduced by Cisco as the Unified Wireless Network (UWN) [3], a highly centralized wired-wireless architecture controlled by a Wireless LAN Controller (WLC). This approach allows for a more efficient spectrum management, as the WLC can collect data from all access points and make decisions based on the global view of the network, which is not possible in a distributed architecture, where each access point makes decisions independently.

The rest of this thesis is structured as follows: Chapter 2 reviews existing academic research on managing radio resources and publicly available information about proprietary RRM solutions; in Chapter 3 I analyze *RRMGreedy*, identify its limitations, and propose improvements that result in the new version of the algorithm called *RRMGreedy++*; in Chapter 4, I present implementation and evaluation results using the ns-3 network simulator; Chapter 5 contains the results and discussion.

Chapter 2

Literature Review

The purpose of this chapter is to explore existing approaches on Radio Resource Management (RRM) in IEEE 802.11 networks, including surveying what is proposed as a part of the 802.11 standard itself, what research has been done, and what is offered by existing commercial solutions. This chapter is organized as follows:

- Section 2.1 briefly reviews the fundamentals of Radio Frequency (RF) communications, and IEEE 802.11 standard for wireless LAN;
- Section 2.2 provides an overview how RRM is facilitated within the IEEE 802.11 standard;
- Section 2.3 provides a synthesis on previous research in RRM;
- Section 2.4 provides an overview of proprietary RRM solutions from major vendors;
- Section 2.5 summarizes the chapter.

Table V contains the list of definitions used in this chapter.

I Overview of IEEE 802.11 standard

Throughout this thesis, I refer to domain-specific terms whose definitions are given in Table V. In this chapter, I briefly review the IEEE 802.11 standard and highlight the aspects most relevant to this study.

A. Transmission medium

The primary medium for communications in IEEE 802.11 is electromagnetic (EM) radio frequency (RF) waves operating within the microwave range [4].

B. Frequency band

Main frequency bands are: the 2.4 GHz band introduced by 802.11b, the 5 GHz band introduced by 802.11a, and the 6 GHz band, introduced by 802.11ax. This study focuses on the first two bands as used the most frequently. In the following I describe those bands in more detail.

Most of the 802.11 amendments, including b,g,n, and partially ax, operate at the unlicensed 2.4 GHz ISM (Industrial, Scientific, Medical) RF band [4], [5]. However, using an unlicensed frequency band introduces multiple challenges: the radio spectrum becomes congested with non-802.11 sources, such as microwave ovens, Bluetooth Personal Area Networks (PAN), cordless phones, etc. [4], [5]. Moreover, 2.4 GHz signals can propagate through solid obstructions such as walls, doors, and windows better than signals operating on higher frequencies [4]. This property can provide better coverage and signal quality for clients, although it can cause interference for neighboring WLANs, which will in turn lead to degradation of signal quality.

The 2.4 GHz ISM band is split into 14 channels. Depending on local regulations, the number of possible channels can vary, but in general channels 1-11 are available in all regions. Assuming a 20 MHz channel width, each channel is characterized by its *center frequency*, ± 10 MHz, with a 5 MHz space between the centers of two adjacent channels, that is, the channels *overlap*. Channel 1 has a central frequency of 2.412 GHz, Channel 14 2.484 GHz. Thus, for channels to be non-overlapping, they must have at least 5 channels or 25 MHz space between the centers. Such non-overlapping channels are 1, 6, 11, with central frequencies 2.412, 2.437, and 2.462 MHz, respectively.

The 5 GHz U-NII (Unlicensed National Information Infrastructure) series of bands is used by the 802.11a/802.11ac/802.11ax amendments. Unlike the 2.4 GHz band, the channels do not overlap. At the end of each band there is a *guard band* as an additional measure to avoid interference. Combined, the bands U-NII-1, U-NII-1, U-NII-2A, U-NII-2C, U-NII-3 provide twenty-five 20 MHz or twelve 40 MHz non-overlapping channels [4]. However, some channels may not be available in different regions, as this band can also be used by military and weather radars. The first channel from the U-NII-1 band has number 36.

C. Signal quality and its metrics

Thus, the presence of physical obstructions, background noise, and interference from other access points urges us to explore possible measurements and metrics for wireless signal quality. In the following, I briefly describe the most widely used quantities.

A measure widely used in RF engineering and employed by Wi-Fi vendors is **Signal-to-Noise Ratio (SNR)**, which is defined as a ratio between the received signal power and the power of background noise:

$$SNR = \frac{P_{signal}}{P_{noise}} \tag{2.1}$$

Since SNR is essentially a difference in power, which is measured in Watts, in practice it is measured in a relative unit on a logarithmic scale called **decibel (dB)** [1,2]:

$$SNR_{dB} = 10 \log_{10} \frac{P_{signal}}{P_{noise}} \tag{2.2}$$

In recent years, **Signal-to-Interference-Plus-Noise ratio (SINR)** measurement have become a more widespread measurement of wireless networksśignal quality. Similarly, it is defined as:

$$SINR = \frac{P_{signal}}{P_{noise} + P_{interf}} \tag{2.3}$$

where P_{signal} is the power of the signal of interest, P_{interf} is the power of interference signals, and P_{noise} is the power of background noise. By considering interference from other 802.11 devices, which is typically a dynamic quantity that changes rapidly over time unlike background noise, SINR describes the EM spectrum situation more accurately.

Received Signal Strength Indicator (RSSI) is a relative measure of signal strength in the range from 0 to 255, where 0 is the weakest signal a receiver can sense. The exact correspondence between RSSI and received signal power is implementation-specific and is left to the hardware manufacturers [4].

D. Radio Resource Management

The scarcity of available frequency bands in the time of growing demand for wireless connectivity has led to the development of methods called *spectrum management* or *radio resource management (RRM)*. Most of the research on RRM is focused on cellular networks, where the coverage area of base stations spans several kilometers, and the number of clients for one station can reach several thousands, so proper spectrum management is vital for operation of cells. However, from a physical layer perspective, the radio situation in 802.11 networks is similar. As described in [6], given a wireless network with a set of access points $\boldsymbol{B}$, which can communicate over a set of channels $\boldsymbol{C}$, with a maximum transmit power of P_{max}, establishing a radio link between a client device and an access point requires the wireless infrastructure to assign:

1. An access point $b \in \boldsymbol{B}$;
2. A frequency channel $c \in \boldsymbol{C}$;
3. A transmission power level $p \leq P_{max}$.

Obviously, channel and transmission power are *global* for a given access point in a sense that all its clients will have to adjust their parameters correspondingly: switch the operating channel and deal with the new received signal strength from their AP. In Wi-Fi, the first requirement is usually managed by the client itself: the user chooses the SSID they wish to use, and in the case if multiple APs have the same SSID, a client device associates with AP having the strongest signal available. Later, a client can switch to another access point within the same extended service set via *roaming* methods, such as *Fast Basic Service Set (BSS) Transition* defined in 802.11r [7]. The roaming decision is ultimately made by a client device, which sends a reassociation request to start the roaming process [4]. However, the access point can force a client to find another access point by sending a deauthentication frame or moving to another channel without notifying. The second and third requirements are part of the current AP configuration and subject to change. A client discovers the current operating channel of the APs by tuning each available channel in succession, while transmission power can only be estimated by measuring the received signal strength. Thus, **the goal of a radio resource allocation algorithm is to optimize spectrum usage within a WLAN by assigning an operating channel and a transmission power level to each access point in a way that maximizes the overall performance of the network.**

As will be shown in Section 2.2, the 802.11 standard does not provide any algorithms for channel and transmission power assignment; however, some amendments introduce methods for measurement, signaling and radio adjustment that can be used for RRM purposes.

Note that related research and commercial solutions introduce many similar terms for the same channel change procedure that can use different algorithms and slightly vary according to the specifics of their application: *Frequency Selection*, *Frequency Planning*, *Channel Selection*, *Channel Planning*, *Channel Assignment*, etc. In this study, we use these terms interchangeably.

II Radio Resource Management in IEEE 802.11

As described in Section 2.1.4, an RRM algorithm seeks to improve network capacity by optimizing spectrum usage through the adjustment of two parameters: the channel frequency and the transmission power of each access point. The very need for an RRM algorithm comes from a scenario known as *Overlapping Ba-*

sic Service Sets (OBSSs) in the IEEE 802.11 standard, when there are two basic service sets operating on the same channel within reach of each other. This scenario leads to co-channel interference, which degrades network performance. In this section, I provide an overview of the methods and techniques provided by the IEEE 802.11 standard that can assist in RRM.

A. IEEE 802.11h

To comply with the legal requirements on transmissions at 5 GHz, the 802.11h-2003 [8] amendment (*Spectrum and Transmit Power Management Extensions*) was introduced. The goal of this amendment is to prevent legacy 802.11a 5GHz APs from interfering with radars. 802.11h introduces [9] spectrum management methods, namely, *Dynamic Frequency Selection (DFS)*, which facilitates automatic change of AP's operating frequency, and *Transmit Power Control (TPC)*, which adjusts AP's transmission power. However, these methods are not designed to implement RRM for the purpose of enhancing network capacity; instead, they focus on avoiding interference with radars functioning in the 5 GHz spectrum.

802.11h describes procedures for: quieting the channel to detect presence of a radar, switching to another channel if a radar is detected, and notifying the client stations (STAs) about channel switch. However, the description of a radar detection procedure is beyond the scope of the 802.11h standard [8]. However, the frame types and measurement reporting mechanisms introduced in 802.11h can be considered as a foundation for further research on RRM algorithms.

B. IEEE 802.11k

The amendment 802.11k [10] (*Radio Resource Measurement*, not to be confused with *Radio Resource Management*) improves the performance of roaming. Roaming is a process of the station moving from one access point to another [4]. The amendment 802.11k improves roaming by allowing stations to request from access points various reports, such as *neighbor reports* for discovering possible access points to which STA can roam, *link measurements* to estimate how well AP can hear the station, etc. This allows stations to reduce power consumption, speed up roaming and decrease power consumption and airtime usage spent on sending probe requests to each channel when trying to find another AP to roam. However, this amendment is also not aimed at optimizing network capacity, but rather at improving roaming performance.

C. IEEE 802.11ax

The amendment 802.11ax [11] (*High Efficiency WLAN*) introduces a means to address the OBSS problem. The amendment introduces *Basic Service Set (BSS) Coloring*, a method to distinguish between different BSSs operating on the same channel. It works as follows: each BSS is assigned with *color*, a 6-bit value spanning from 1 to 63, that resides in the PHY header. Devices belonging to the same BSS check this color when demodulating transmissions (that is, check if this frame is an *intra-BSS*). If the frame has an incorrect color (*inter-BSS* frame), the device stops further demodulation, thus conserving processing resources. [12]. If an inter-BSS frame is found to use the same color, AP switches to another color. One can see that such a solution completely relies on the CSMA/CA media access control (MAC) mechanism and is not able to address the hidden terminal problem.

That is, if the AP serving the overlapping BSS is a hidden terminal, the signal will be disrupted near the destination, so the receiving station will not be able to demodulate the signal to inspect the color tag. Thus, this technique does not solve the problem of co-channel interference and does not provide channel planning, so this cannot be considered as a complete RRM solution.

III Previous Works

Interference from other access points (APs) poses a serious obstacle [13] to provide an acceptable quality of service in large 802.11 WLAN deployments. While the 802.11 carrier-sense MAC protocol is designed to withstand interference, however, interference reduces available airtime and causes frame losses. Thus, network capacity and performance tend to drastically degrade [14].

In Wi-Fi WLANs, interference mostly originates from *rogue access points* (RAPs), which are operated by third parties and, in general, are not under the control of WLAN administrators. According to [13], interference from rogue APs can introduce up to 50% delays in a WLAN. Moreover, the prevalent amount (more than 70%) of rogue APs are stationary [13], so their radio presence can be considered as a constant factor in the WLAN. Note that *rogue* does not imply that these APs are malicious or pose threats other than congesting channels and occuping airtime as a result of their legitimate operation.

In this light, attempts to improve spectrum management through channel assignment and transmit power control adjustment algorithms encompass research on RRM algorithms. As shown in Section 2.2, the IEEE 802.11 standard provides a limited set of tools for RRM, leaving assignment algorithms and policies on the behalf of WLAN equipment vendors. As reported in [13], Cisco's RRM software, shipped with Cisco Aironet APs and Cisco WLAN Controller, was able to improve network performance using Dynamic Channel Selection (DCS) and Transmit Power Control (TPC) so that carrier sense interference was responsible for only 5% of network delays. RRM solutions from Cisco and other vendors will be surveyed in Section 2.4. However, since those technologies are proprietary, their implementation details are not disclosed, so they cannot be properly evaluated in independent research or adopted by third-party vendors. Moreover, such solutions lack interoperability, so it is difficult to use them with networking products from other vendors, which is a major obstacle for large-scale WLAN deployments and leads to vendor lock-in situations. Thus, research on RRM algorithms is important for the industry.

A. *Radio Resource Management Approaches*

Since RRM is a broad topic that does not imply a single methodology, approach, or even a definition, the classification of RRM algorithms is a challenging task. Most of the papers with the "Radio Resource Management" keyword are focused on problems of cellular networks, such as LTE or 5G. However, in some cases problem formalization, optimization objectives and algorithms can also be useful for research in 802.11 networks. Still, band usage, client management, deployment, and operation specifics make most of the approaches from cellular networks inapplicable for 802.11 networks. To my knowledge, no comprehensive survey on RRM in 802.11 WLAN exists. I will refer to [15], which provides a detailed overview of previous research, and [16], which describes the state-of-the-art in RRM. In [15], the authors classify the RRM algorithms into three categories:

- *per-cell* approaches seek to optimize the RF situation within the AP's cell coverage. This means that adjustments of radio parameters are applied on a cell scale and will be in effect for all stations within the cell. Such a classification can be further divided into the following:

- *localized (uncoordinated) per-cell*, where each AP performs RRM decisions independently;
- *centralized per-cell*, where a central entity, such as WLAN Controller, performs RRM decisions for all APs within a WLAN. Some authors refer to this approach as *super-cell* approach [14];
- *coordinated per-cell*, employing cooperation between APs for making coordinated RRM decisions.

- *per-link* approaches, which optimize the transmission power for a given station;
- *per-flow* approaches, which employ frequency and AP transmission power adjusting to optimize the QoS to the granularity of a given traffic flow within a station, for example, to the flow of a VoIP application.

In fact, a simple localized per-cell RRM is already widely implemented: almost every home Wi-Fi router has the option to select the channel automatically. Typically, in this case, the access point surveys each channel, makes an estimate how congested it is, and then switches to the least congested one. This technique is called *least-congested channel scan*, or *least-congested channel search* (LCCS), and the original design uses the number of associated clients as an estimation of channel congestion [17]. As analyzed in [18], LCCS has several limitations:

1. LCCS is unable to accurately identify interference scenarios where clients connected to different Access Points (APs) interfere with each other without

the APs themselves causing interference. This issue is particularly prevalent in real-world setups where APs are strategically placed to ensure wide coverage while overlapping minimally to avoid coverage gaps;

2. LCCS also falls short in optimizing channel reuse based on the distribution of clients. It fails to account for the interference experienced by clients, thus missing the opportunity for channel reuse strategies based on client locations and densities.

In general, uncoordinated decision-making like LCCS tends to yield suboptimal results. Consider an extreme case where a number of APs can sense that channel C_i is not congested and make a decision to switch to that channel. As a result, C_i becomes congested, so APs will seek to switch to another channel C_j, where the problem will reoccur. This situation suggests that using a coordinated RRM policy between APs can improve overall WLAN capacity and thus achieve a global optimum.

Research on Transmit Power Control (TPC) methods, which adjust transmission power P_{Tx} to maintain an acceptable Signal-to-Noise-plus-Interference Ratio (SINR), shows potential for enhancing bandwidth. However, concerns regarding these studies' simplified models, unrealistic experimental setups, and statistically uncertain outcomes suggest the need for further investigation [19], [20]. The effect of IEEE 802.11 roaming on TPC is underexplored, especially in scenarios where APs are part of the same extended service set (ESS), which is more common in enterprise WLANs.

As shown in [21], per-link TPC considerably improves WLAN performance, achieves more spatial reuse, increases throughput, and able to avoid channel access asymmetry and receiver-side interference (also known as hidden-node problem). However, such an approach has certain hardware requirements, namely *per-packet transmit power control*, a feature available only for a small selection of 802.11 chipsets. In turn, implementing per-flow RRM in standard 802.11 networks requires an advanced framework to identify specific traffic flows and assess their Quality of Service (QoS) demands [15]. Therefore, this thesis will focus on solutions that are more practical and applicable to the hardware and software currently available on the market.

B. Mathematical Models for Radio Resource Management

Building on the definition of the Radio Resource Management (RRM) problem introduced in 1, we consider a network composed of a set of base stations (access points) denoted as $\boldsymbol{B}$, capable of operating over a collection of channels $\boldsymbol{C}$, each with a maximum transmission power limit P_{max}. The core objective of RRM is to establish a radio link between a client device and an access point by assigning a triplet (b, c, p), where $b \in \boldsymbol{B}$ represents the base station, $c \in \boldsymbol{C}$ the channel and $p \leq P_{max}$ the transmission power, so that the network capacity is maximized.

The RRM problem, thus, decomposes into three crucial tasks:

- *Client Assignment* — allocating a base station (access point) to a client;
- *Adaptive Channel Selection* — determining the optimal frequency (channel) for client communication;
- *Transmit Power Control* — setting the appropriate transmission power for client communication.

Client Assignment is typically handled by the 802.11 client through roaming decisions, with amendments like 802.11k/r/v designed to enhance and expedite the process of switching to an access point that offers superior service quality. This aspect, therefore, lies beyond the scope of this study.

Channel allocation and transmit power selection, though extensively studied, are often addressed as separate entities in the literature. Combining these factors introduces complexity, as their objectives can conflict. For example, if the objective is to minimize interference, it can be achieved with minimizing transmission power. However, such an approach probably does not satisfy the coverage and quality-of-service requirements. However, considering channel allocation and transmit power control together can also be troublesome, since a change in one variable would change the overall RF situation and the algorithm would not converge.

Another aspect is the choice of metrics. Although the most intuitive and desired metrics are high-level metrics such as network throughput and capacity, actual values of such metrics cannot be used at the time of RRM computations: a significant time of monitoring is required to estimate how throughput changed, so only past records can be used. Another, simpler approach, followed by many works, is to assume that network throughput or capacity is a function of one or more physical layer metrics, such as interference, Received Signal Strength Indicator (RSSI), received signal power, etc. Indeed, a low interference level and low power signal from other APs implies that less transmission errors tend to happen, and more frames can be transmitted with CSMA/CA MAC mechanism.

Reliance on physical layer metrics allows for more immediate adjustments in the RF configuration of a WLAN, ensuring the adaptability of the network to immediate environmental changes. However, in practice, RRM adjustments can lead to disruptions. Most client devices lack support for the Channel Switch

Announcement feature from 802.11h, interpreting a channel switch as if the AP has become unavailable. Therefore, utilizing historical data becomes instrumental in making informed, albeit infrequent, and periodic RRM decisions.

Channel and transmit power settings for each AP should yield optimal values of some given metric for the whole WLAN, such as: interference level, WLAN throughput, WLAN capacity, etc. Thus, most of the works consider RRM as an optimization problem, such as integer linear programming (ILP) [22] [23] or binary quadratic programming (BQP) [16]. Since both ILP and BQP are proven to be NP-hard problems, researchers propose heuristics to reduce search space [14], or apply meta-heuristic methods such as genetic algorithms [24] or deep neural networks [16]. Other approaches, while they do not solve the optimization problem explicitly, aim to keep some target metric, such as Signal-to-Interference-plus-Noise Ratio (SINR), within pre-defined acceptable boundaries [19]. In [18], authors employ a graph model, where APs are represented as nodes and edges connect APs that can potentially interfere. Using such a model, each node can be assigned a color that represents its channel.

It is important to note that in 802.11 WLANs, all clients connected to a specific Access Point (AP) use the same frequency and transmission power settings. Given the limited applicability of per-link (per-client) Transmit Power Control (TPC), as previously discussed, it is assumed that both frequency and transmission power are configured for the entire cell. This means that all clients of a given AP operate on the same frequency and the AP maintains a consistent transmit power level for communication with all its clients.

IV Proprietary RRM Solutions

This section reviews proprietary RRM solutions offered by leading vendors in the enterprise WLAN market. I focus on Cisco, Juniper Networks, and Ruckus Networks, since they are the most popular vendors in the enterprise WLAN market [25]. To the best of my knowledge, peer-reviewed evaluations of these proprietary RRM efficiency are very limited and scarce, so one could only rely on the claims made by the vendor themselves.

A. Cisco

Cisco's RRM strategy is integral to its Cisco Centralized Architecture, known as the Unified Wireless Network (UWN) [3]. In the UWN framework, a single or multiple Wireless LAN Controllers (WLCs) manage up to several thousand Access Points (APs). These WLCs act as the core of the WLAN architecture, enabling centralized control and collection of telemetry from all APs within the network. A WLC can be specialized hardware or a virtual machine hosted in the cloud [26]. Effectively, in UWN, APs can be thought of as Wi-Fi network interface cards for the WLC, providing minimal real-time functionality from the 802.11 standard that cannot be carried out by WLC due to propagation and transmission delays. This architectural model has become the de facto standard for large-scale enterprise WLANs and is used by most major vendors in the industry.

Cisco offers several RRM solutions. First, CleanAir is a flagship technology from Cisco [27] to optimize network performance, avoid jamming, and detect interference sources, including non-802.11 ones. Cisco states that it outperforms competitors through several features:

- It utilizes specialized hardware for RF analytics. For example, Cisco Catalyst 9100 Series APs contain a dedicated radio for background RF scanning. This functionality allows for continuous service provision to clients without disrupting the main AP radio transceivers. Additionally, the Cisco RF ASIC, a dedicated chip, enables advanced wireless network analytics and spectrum analysis that are unavailable to conventional Wi-Fi modules;

- It classifies and visualizes interference sources thanks to dedicated RF hardware;

- Comprehensive WLAN-wide radio resource management, supplying both real-time and historical data at varying levels of granularity;

- CleanAir is event-driven, that means it can adapt to changing RF environment and adjust radio parameters in a matter of few minutes, drastically reducing downtime.

However, CleanAir is only available for the higher-end models in the Cisco product line, posing limitations for its large-scale deployment. Furthermore, the lack of compatible radio analytics hardware from other vendors and undisclosed implementation details restrict the utility of this technology for integration with non-Cisco equipment. On the other hand, the Cisco Catalyst product line of WLAN Controllers also provides "regular" RRM functionality that only requires regular Wi-Fi chipset and can be used with all Cisco APs [1]. The trade-off for this convenience is access to less detailed information about the RF environment and the necessity for APs to temporarily switch off their current channel to conduct scanning. In this case, APs collect statistics on their current channel whenever they are not transmitting data. In addition, APs periodically scan other channels to collect statistics [26]. During scanning, the AP becomes unavailable to its clients, resulting in increased latency for them.

Cisco RRM employs a super-cell concept. In such a scheme, a group of geographically close APs (forming an *RF Group*) is managed by a designated WLAN Controller (*RF Group Leader*). A subgroup unit within an RF group is called *RF Neighborhood*, and consists of AP that can hear each other at signal strength ≥ -80 *dBm* [26]. Each AP is associated with two lists: RX neighbors, that is, the list of APs that a given AP can hear, and TX neighbors, a list of APs that can hear the given AP. Cisco RRM maintains a *cost metric* for every channel, which is an assessment of the channel's quality derived from RSSI, co-channel interference, and non-WiFi interference.

B. Juniper Networks

Juniper Networks offers Mist AI RRM technology to improve network performance. The notable features are [28], [29]:

- Automatic dual-band radio management — if RRM system finds 2.4-GHz radio transmitter to be unused on a given AP, it disables the radio to free airspace for other access points;
- Juniper Mist APs incorporate the Predictive Analytics and Correlation Engine (PACE) "to monitor conditions and make out-of-band adjustments" [29];
- Telemetry is sent to the Juniper Mist Cloud, so that the cloud can periodically fine-tune APs based on historical data and usage statistics;
- Employing a Reinforcement Learning (RL) methodology for the strategic

planning of channel selection and power settings across APs in a WLAN, aiming for optimal network performance [28].

C. Ruckus Networks

Ruckus Networks offers ChannelFly RRM technology, which provides automatic channel selection. ChannelFly estimates the capacity of each channel by continuously monitoring the activity of each channel across the 2.4 and 5 GHz bands. Based on this information, ChannelFly develops a statistical model to predict which channel will offer the highest capacity for clients, as detailed in [30]. A key benefit of ChannelFly is its ability to avoid "dead time", defined as the period an AP spends scanning different channels during which it cannot communicate with clients. This capability implies the inclusion of dedicated scanning radios in Ruckus APs, allowing continuous communication with clients while performing channel assessments.

Additionally, Ruckus offers "smart adaptive antenna array" technology. This feature enhances the beamforming of signals from Ruckus APs, focusing the transmission towards clients to improve the Signal-to-Noise Ratio (SNR).

D. Aruba Networks

The Adaptive Radio Management (ARM) technology by Aruba represents an earlier approach to Radio Resource Management (RRM), utilizing Adaptive Channel Selection and Transmit Power Control to enhance the RF environment in WLANs. ARM stands out for its algorithmic simplicity and the thoroughness of its documentation provided by Aruba, in contrast to other vendors [31].

The key features of the ARM include [32]:

- **Application Awareness**: Addressing the "dead time" caused by APs during channel scanning, ARM throttles the frequency of background scans based on current traffic load, reducing scans under heavy traffic and resuming normal scanning rates when traffic diminishes [2].
- **Mode Awareness**: To mitigate interference in environments with densely installed APs, ARM can switch excessive APs to Air Monitor mode, where they continuously collect and send RRM telemetry to the controller.
- **Band Steering**: Promotes the use 5 GHz band to clients, instead of more congested and higher-range 2.4 GHz band.
- **802.11n HT Mode Support**: ARM can utilize a 40 MHz channel pair for 802.11n networks, selecting the best primary and secondary operating channels automatically.
- **Noise and Error Monitoring**: Distinguishes between 802.11 and non-802.11 noise sources, improving network reliability.
- **Spectrum Load Balancing**: Analyzes client distribution across neighboring APs to direct new connections to less burdened APs, though clients may reconnect to their original choice upon a subsequent attempt.
- **Noise Interference Immunity**: Adjusts the receiver sensitivity threshold to ignore weak and non-802.11 signals, reducing unnecessary decoding efforts and improving network performance.

However, reports from system administrators suggest that RRM decisions in Aruba ARM are made by APs rather than by the controller. Some other user complains include unnecessary disabling of 2.4GHz radios and erroneous TPC that leads to coverage holes [33].

However, ARM is a legacy technology. Its successor, Aruba AirMatch, introduced in recent ArubaOS versions, is a more sophisticated RRM technology, which is based on AI and machine learning and is able to perform channel and power planning on a WLAN-wide scale, suggesting ARM was implemented in a per-cell way and AirMatch is a super-cell solution.

Notable AirMatch features:

- Channel width adjustment based on device density - the more devices are connected to an AP, the narrower channel width is used to allow channel reuse and reduce interference;
- APs measure RF environment for 5 minutes every 30 minutes;
- Decisions based on a 24-hour period analytics unlike instant RF situation snapshots in ARM;
- Elimination of coverage holes based on TPC.
- Configurable thresholds in channel quality improvements to trigger channel and EIRP planning, default threshold is 15%.
- ClientMatch technology that manages clients: performs load balancing between APs, encourages clients to switch to APs providing better signal strength and using higher bands (5 GHz or event 6 GHz in 802.11ax)

Similarly to ARM, Aruba provides more information about AirMatch operating logic than other vendors about their RRM solutions.

According to [31], AirMatch blacklists the channel for channel selection if a radar was detected on it (in the 5 GHz case) or in the case if a high noise level was detected on it (for all bands). In those cases, AirMatch will select the channel with a minimum interference index.

It is not clear whether AirMatch uses the same metrics as ARM, but only ARM metrics are described in the documentation. To make RRM decisions, ARM uses two metrics:

- **Coverage Index** - comprises two components, x and y:
  - x is weighted calculation of Signal-to-Noise Ratio (SNR) for all APs on a given channel;
  - y is the weighted sum of the SNRs that neighboring APs within the group observe on the same channel.
- **Interference Index** - measures co-channel and adjacent-channel interference, calculated as a sum of four quantities a, b, c, d:
  - c is the channel interference the AP neighbors see on the selected channel;
  - d is the interference the AP neighbors see on the adjacent channel.

Furthermore, Aruba APs collect several other metrics, including L2 metrics [31]:

- Number of Retry frames, measured in %
- Number of Low-speed frames, measured in %
- Number of Non-unicast frames, measured in %

- Number of Fragmented frames, measured in %
- Number of Bandwidth seen on the channel, measured in kbps
- Number of PHY errors seen on the channel measured in %
- Number of MAC errors seen on the channel measured in %
- Value of noise floor on the specified AP

Aruba documentation indicates that these metrics offer a "snapshot of the current RF health state" [34], suggesting that they are informational tools for network administrators rather than being actively used in RRM decision-making.

V Conclusion

Summarizing the insights from previous sections, I can conclude that the problem of radio resource management in 802.11 WLANs is still relevant, since the IEEE 802.11 standard provides only limited tools for RRM, while existing commercial solutions are proprietary and lack interoperability. Thus, there is a need for a novel RRM algorithm that can be implemented in existing enterprise WLAN infrastructure and improve overall network performance.

I find super-cell approach most fitting for a modern RRM algorithm that can be applied in real-world WLAN deployments. Super-cell algorithms, while being practical and having less obstacles in hardware and current device drivers compared to other approaches, still have the potential to vastly improve the RF situation and, thus, WLAN performance.

Centralized management that is typically utilized in super-cell RRM is the standard approach when building modern WLANs, allowing to gather more information about RF environment and come up with more optimal allocations compared with local RRM decision-making. Moreover, in the long term, the presence of WLC, as a centralized entity with orders of magnitude higher computation power and ability to collect and store statistics from all APs throughout the WLAN, releases the burden of RRM from APs and potentially improves the overall network efficiency.

Despite the promising capabilities of per-link and per-flow radio resource management approaches for optimizing wireless networks in a more fine-grained and application-aware manner, they currently have considerable limitations that prevent them from being implemented in production wireless networking solutions.

As a summary of this survey, we can identify the research gap: the problem of radio resource management in 802.11 WLANs is still relevant, since IEEE 802.11 standard does not provide fully-fledged RRM, while existing commercial solutions are proprietary and lack interoperability. Thus, there is a need for an RRM algorithm addressing key issues, including:

- Design for centralized management of enterprise WLAN, working as a part of Wireless LAN Controller;
- Applicability with current hardware and software, namely:
  - Effortless integration with OpenWRT-based APs;
  - Requires data like physical and link-layer statistics that can be obtained using only regular Linux Wi-Fi drivers like `nl80211`, and standard Linux networking tools;
- Performing not worse than existing RRM algorithm *RRMGreedy*, analyzed

in Chapter 3.2;

- Able to combine both channel selection and transmit power adjustment to improve RF environment and network performance.

The following chapters will focus on analyzing the limitations of current algorithms and developing a new one.

Chapter 3

Baseline Solution Analysis and Algorithm Design

The chapter is organized as follows. Section 3.1 defines the problem of RRM and the objectives of an RRM algorithm. In Section 3.2, I describe the RRM algorithm called *RRMGreedy*, developed by a Russian telecommunications vendor, analyze the flow and asymptotic complexity of the algorithm and identify its flaws and limitations. The objective of this study is to create an algorithm outperforming *RRMGreedy* in terms of network capacity and throughput; in Section 3.3 I propose a new RRM algorithm, called *RRMGreedy++*, that addresses the flaws of *RRMGreedy*.

I Problem statement

Using the definition of the RRM problem from 2.3.2, I define the problem as follows: given a set of access points $AP = \{AP_1, AP_2, \ldots, AP_n\}$, where each access point AP_i has a set of wireless interfaces $W_i = \{w_{i1}, w_{i2}, \ldots, w_{im}\}$, where each interface w_{ij} can operate on a set of channels $C = \{c_1, c_2, \ldots, c_k\}$, the goal is to find a channel assignment $f : W \to C$ and transmit power assignment $g : W \to P$ that maximizes network throughput and capacity. The problem can be formulated as a combinatorial optimization problem, where the goal is to find a configuration of channels and transmit powers that maximizes the objective function, which is a function of network throughput and capacity.

II Baseline solution analysis — RRMGreedy

The greedy RRM algorithm, used by a major telecommunications vendor, which will be further referred to as *RRMGreedy*, is based on background scanning by access points. The goal of this study is to create a new RRM algorithm that complies with the requirements and addresses the research gap formulated in Section 2.5. In particular, the novel algorithm should result in network throughput and capacity greater or equal than *RRMGreedy* in corner cases. In this light, I start with describing *RRMGreedy*.

A. *RRMGreedy: flow description*

This section describes *RRMGreedy*. As the name suggests, the algorithm tries to achieve optimal radio resource allocation in a greedy way, taking the local optimum for each device in an RRM group. The algorithm operates on the granularity of interfaces rather than access points. I consider it to be reasonable, since practically different interfaces operate on completely different frequency bands, so when are dealing with, for example, the 5 GHz band, the whole access point can be considered as a single interface.

In other words, I consider **RRM group** G as a set of **wireless interfaces** w

(which will be further referred to as simply **interfaces**), where several interfaces can belong to a single access point. On a high level, the algorithm consists of two main steps — **Channel Selection** and **Transmit Power Adjustment**:

1. **Channel Selection**:

   (a) compute **group interference** I_G — sum of interference scores for each interface in group:

   $$I_G = \sum_{i=0}^{|G|} OnIfaceInterference(w_i) \tag{3.1}$$

   Initial group interference score is denoted as I_0.

   (b) for each w_i, compute its updated interference score for every channel c_i^k available for that interface:

   $$interf_i = OnIfaceInterference(w_i, c_i^k) \tag{3.2}$$

   If there is a channel c_i^k that reduced $interf_i$, update interface settings to issue change to that channel later.

   (c) **continue while there is an improvement**, i.e., previous group interference score is larger than the latest one: $I_{j-1} > I_j$ for $j > 1$

2. **Transmit Power Adjustment**:

   (a) Compute group interference as in step 1a

   (b) Identify the *worst interface* w_m causing the *worst interference* (i.e., whose from-interference score is the largest):

$$w_m = \underset{i}{\operatorname{argmax}}\, FromIfaceInterference(w_i) \tag{3.3}$$

(c) Gradually reduce transmit power of w_m with exponential backoff until it stops being the worst interface or reaches minimum Tx power.

After providing a high-level overview of the algorithm, I will analyze the used functions, since they contain the core logic of the algorithm. These functions will be discussed in the order they are executed within the algorithm.

Function *PreprocessRRMData()* does not perform actual RRM logic and is used to preprocess the data for further calculations. Namely, it reorganizes scan results for each interface in a way that for every channel every interface maintains a list of

- Signals originating from access points within the RRM group $\boldsymbol{G}$. In RRMGreedy, such signals are called *inner* signals;
- Signals originating from access points outside of the RRM group (i.e., coming from *rogue APs*). In RRMGreedy terms, such signals are called *outer* signals.

The pseudocode for *PreprocessRRMData()* is shown in Algorithm 2.

ALGORITHM 1: RRMGreedy

Input: ***scandata***: scanning data from access points
G: set of wireless interfaces within the RRM group and its settings
Output: ***resultConfig***: final configuration for access points

```
groupState ← PreprocessRRMData(G, scandata)
// Step 1: Start with max Tx power for each interface
for each interface in G do
    groupState[interface].TxDiff ← (interface.MaxTxPower − interface.TxPower)
// Step 2: Greedy Channel Search
groupInterference ← GroupInterference(groupState)
repeat
    for each interface in G do
        C_i ← available channels for interface
        newChannel ← argmin_{ch∈C_i} OnIfaceInterference(groupState, interface, ch)
        groupState[interface].channel ← newChannel
    groupInterference ← GroupInterference(scandata)
until no improvement in groupInterference
// Step 3: Greedy Transmission Power Adjustment (if enabled)
repeat
    // Identify the "worst interface" interface_worst causing the greatest
       interference for other APs and the maximum RSSI of interface_worst on
       some other interface
    interface_worst, RSSI_worst ← argmax_{w_i∈G} FromIfaceInterference(groupState, w_i)
    // RSSI_min is a predefined threshold for the minimum RSSI, by default
       set to -100 (dBm)
    groupState[interface_worst].TxDiff ← (RSSI_min − RSSI_worst) / 2
    groupInterference ← GroupInterference(scandata)
until no improvement in groupInterference
// Step 4: Send configuration changes to APs
resultConfig ← getUpdatedConfiguration(groupState)
return resultConfig
```

ALGORITHM 2: PreprocessRRMData

```
foreach interface in G do
    scanFromIface ← scandata[interface]
    foreach BSSID, signalData in scanFromIface do
        channel ← signalData.channel
        if bssid in RRMGroup then
            interface.InnerBSS[channel].append(signalData)
        else
            interface.OuterBSS[channel].append(signalData)
```

The result of PreprocessRRMData is saved in a table called *groupState*. This structure maintains updated channel and transmit power settings for each interface in the RRM group as RRMGreedy progresses.

Function *OnIfaceInterference(groupState, interface, channel)* (Algorithm 3) calculates interference experienced by given access point (and thus by its wireless interface) from outer and inner signals. Essentially, it breaks down to calculating interference as a sum of outer and inner interference scores for a given interface w_i on a given channel c_i:

$$interf_i = OnIfaceInterference(w_i, channel) = interf_i^{outer} + interf_i^{inner} \tag{3.4}$$

$$interf_i^{outer} = \sum_{j=0}^{|S_i^{outer}|} ChannelInterference(channel, s_i^j.channel) \cdot scale(s_i^j) \tag{3.5}$$

$$interf_i^{inner} = \sum_{j=0}^{|S_i^{inner}|} ChannelInterference(channel, s_i^j.channel) \cdot scale(s_i^j + w_i.TxDiff) \tag{3.6}$$

where:

S_i^{outer}	set of signals sensed by w on from APs $\notin G$;
S_i^{inner}	set of signals sensed by w_i from APs $\in G$;
s_i^j	j-th signal received by w_i, in S_i^{outer} or S_i^{inner};
$w_i.TxDiff$	transmit power adjustment for w_i;
$ChannelInterference(ch_1, ch_2)$	interference between ch_1 and ch_2 (Algorithm 3);
$scale(x)$	normalization function 3.7.

Transmit power is normalized to $[0, 1]$ with min-max normalization using pre-defined RSSI (Received Signal Strength Indicator) thresholds $RSSI_{min}$ and $RSSI_{max}$ (3.7):

$$scale(x) = \frac{x - RSSI_{min}}{RSSI_{max} - RSSI_{min}} \tag{3.7}$$

The interference between channels is estimated with *ChannelInterference()* function(Algorithm 3). The distance between adjacent channels is assumed to be 5 MHz. The interval of length n should have $n - 1$ channels in between, which is the reason for the factor *add*.

ALGORITHM 3: ChannelInterference

```
Function ChannelInterference(ch_1, ch_2, width):
    add ← 0
    if ch_1 < 36 then
        add ← 1
    if |ch_1 − ch_2| ≥ width/5 + add then
        return 0
    else
        return 1
```

ALGORITHM 4: OnIfaceInterference

```
Input:  groupState: table containing state of G
        interface: interface to calculate interference for channel: channel to calculate
        interference on
Output: Estimated cumulative interference score experienced by interface
ifaceScanData ← scandata[interface]
cumOuterInterf ← 0
// Calculate interference from outer APs
foreach otherChannel, otherChannelSignals in ifaceScanData.OuterBSS do
    channelInterfScore ←
      ChannelInterference(otherChannel, interface.Channel, interface.ChannelWidth)
    channelOuterInterf ← 0
    foreach signal in otherChannelSignals do
        sigRating ← signal.RxPower (in dBm) normalized to [0, 1] scale
        channelOuterInterf ← channelOuterInter + finterferenceScore × sigRating
    cumOuterInterf ← cumOuterInterf + channelOuterInterf
cumInnerInterf ← 0
// Calculate interference from inner APs
foreach otherInterface, signalsFromOtherInterface in ifaceScanData.InnerBSS do
    channelInterfScore ←
      ChannelInterference(oth.Settings.Central, interfaceChannel, interfaceChannelWidth)
    if channelInterfScore == 0 then
        continue
    channelOuterInterference ← 0
    foreach signal in signalsFromOtherInterface do
        adjustedSourceSignal ← signal + otherInterface.TxDiff
        sigRating ← adjustedSourceSignal power (in dBm) normalized to [0, 1] scale
        channelOuterInterference ← channelInterfScore × sigRating
    cumInnerInterf ← cumInnerInterf + channelOuterInterf
return cumOuterInterf + cumInnerInterf
```

The function *FromIfaceInterference()* (Algorithm 5) calculates the amount of interference originating from w_i and experienced by other interfaces in the group $\boldsymbol{G}$. The function returns two values: the cumulative interference quantity, that is, how much w_i affects the RRM group, and the maximum RSSI of w_i on some other interface w_j. Using *FromIfaceInterference()*, RRMGreedy finds the *worst interference* and gradually reduces its transmit power to improve the group interference score.

ALGORITHM 5: FromIfaceInterference

Input: *groupState*: table containing state of $\boldsymbol{G}$
interface: interface to calculate interference from

Output: *cumInterfFromInterface*: cumulative interference quantity estimating how much *interface* affects the RRM group
maxSignal: maximum signal strength from *interface* experienced by other AP

```
cumInterfFromInterface ← 0
maxSignal ← MinSignal
foreach otherInterface in scandata do
    if otherInterface == interface then
        continue
    if otherInterface has not received signals from interface then
        continue
    interfScore ←
      ChannelInterference(interface.Channel, otherInterface.channel, otherInterface.channelWidth)
    if interfScore == 0 then
        continue
    interfaceInterfSum ← 0
    foreach sig in thisIface do
        adjustedSignalPower ← interface.Signal + interface.TxDiff
        maxsignal ← max(signal, maxsignal)
        sigRating ← adjustedSignalPower normalized to [0, 1]
        interfaceInterfSum ← interfaceInterfSum + (interfScore × sigRating)
    cumInterfFromInterface ← cumInterfFromInterface + interfaceInterfSum
return cumInterfFromInterface, maxsignal
```

B. Evaluating asymptotic complexity of RRMGreedy

In this section, I evaluate the asymptotic complexity of *RRMGreedy* algorithm. First, I consider the complexity of used auxiliary functions:

1. $ChannelInterference(ch_1, ch_2, width)$ has constant complexity $O(1)$;
2. $OnIfaceInterference(G, w_i)$ has complexity linear w.r.t. the number of signals sensed by w_i, i.e., $O(|S_i|) = O(|S_i^{outer}| + |S_i^{inner}|)$;
3. $FromIfaceInterference(G, w_i)$ has complexity proportional to the product of number of interfaces in G and amount of signal samples from w_i heard by other interfaces, which is limited by some constant C, reasonably small in practice ($C < 10$), i.e., $O(|G| \cdot C) = O(|G|)$.

Next, I analyze the time complexity of the main phases of the algorithm:

1. **ProcessScanDataForGroup** has complexity $O(|G| \cdot |S|)$, where $|S|$ is the total number of signals in all scans from all interfaces in G;
2. **Set Maximum Transmission Power** has complexity $O(|G|)$;
3. **Optimize Channel Selection** is an iterative algorithm, going for each $w_i \in G$ and for each channel $c \in w_i.channels$ running $OnIfaceInterference(G, w_i)$ until there is no improvement, so its worst-case time complexity can be estimated as $O(|G| \cdot C \cdot K \cdot |S_i|)$, where C is the number of channels and K is the number of iterations;
4. **Adjust Transmission Power** is an iterative algorithm, going for each $w_i \in G$ until there is no improvement yielded by $FromIfaceInterference()$, so its

worst-case time complexity can be estimated as $O(|G| \cdot K \cdot |G|) = O(|G|^2 \cdot K)$, where K is the number of iterations;

5. **Final Data Assembly and Event Generation** has complexity $O(|G|)$.

C. RRMGreedy: limitations and conclusions

Based on the analysis carried out above in this chapter, we can identify the following issues with the *RRMGreedy* algorithm:

1. *ChannelInterference()*, calculating how much two channels interfere, only yields binary values 1 or 0. Due to the presence of cross-channel interference, the result of *ChannelInterference()* should be continuous in range $[0;\ 1]$, not discrete;

2. Noise floor should be included into the interference metric, since it can drastically affect the performance on the channel;

3. Channel utilization is not taken into account. Consider the case when 3 APs with no associated stations operate on channel A, and 1 AP with 10 associated stations operates on channel B. The algorithm does not keep track of the number of stations or other channel utilization data, and will favor B, even though A is less utilized, since the only traffic APs on A are likely to produce is rare and periodic beacon frames;

4. Assuming maximum transmit power for each in-group AP in the Step 1 of Algorithm 1 is not reasonable, since it tends to exaggerate the interference scores and thus lead to unnecessary channel switches;

5. When calculating the interference score, if there are several signals from the same APs, they will be summed as they were different APs, again leading to exaggerated interference scores;

6. Even if next iteration yields worse interference score, there is no rollback to the previous, better state;

7. Change with no improvements will be staged anyway, so disruptive channel switch will occur even if it is not necessary;

8. Finally, selecting channel that minimizes the interference for an interface in a greedy way could not lead to the global optimum.

III RRMGreedy++ design

Based on the limitations identified in Section 3.2.3, I introduce the following improvements:

1. **Continuous interference estimation**: I propose a slightly updated *ChannelInterference* implementation, referred to as *ChannelInterference++*, that estimates interference score based on how much do channels overlap (Algorithm 6);

2. **Updated cumulative metrics approach**:

   - **Noise floor** for *OnIfaceInterference()* cumulative estimation. This affects channel selection decision, discouraging channels with high noise floor level;

- **Channel utilization**: cumulative interference metric now takes into account channel utilization in its simplest form: number of clients of the AP. This will discourage APs from using channels with many active clients that can be heard.

These improvements are used in *OnIfaceInterference++* (Algorithm 7). I also refactored the program to make it more simpler while preserving the behaviour. Namely, splitting signals into "inner" and "outer" ones is not reasonable. The only reason why such division can take place is *TxDiff* adjustment for inner devices, and this is trivial to implement within one loop.

3. **RRMGreedy Flow Improvements**:

   - The "*TxDiff* $\rightarrow$ max" phase of RRMGreedy (Step 1 of Algorithm 1) is moved after the Channel Selection part and before the Transmission Power Adjustment part;
   - The *GroupState* structure, containing the current state of $\boldsymbol{G}$, is saved for the previous iteration of the algorithm. If the interference score increases, *RRMGreedy++* aborts and returns to the previous *GroupState*;
   - Iterations that yield a new group configuration with *GroupInterference* $< \epsilon$, where ϵ is a configurable threshold pre-defined to $\epsilon = 0.005$, are discarded.

ALGORITHM 6: ChannelInterference++

```
Function ChannelInterferencePlusPlus(ch_1, ch_2, width = 20):
    if ch_1 < 36 and ch_2 < 36 then
        freqDiff ← |ch_1 − ch_2| × 5
        maxDistance ← width + 5
    else if ch_1 ≥ 36 and ch_2 ≥ 36 then
        freqDiff ← |ch_1 − ch_2| × 20
        maxDistance ← width
    if freqDiff > maxDistance then
        return 0.0
    overlap ← 1 − freqDiff / maxDistance
    return overlap
```

ALGORITHM 7: OnIfaceInterference++

Input: *groupState*: table containing state of ***G***

interface: interface to calculate interference for *channel*: channel to calculate interference on

Output: Estimated cumulative interference score experienced by *interface*

```
ifaceScanData ← scandata[interface]
cumInterf ← 0
// Calculate interference from both inner and outer APs
foreach otherBssid, otherSignal in groupState[interface].receivedSignals do
    ciScore ← ChannelInterference(otherSignal.RSSI, channel, interface.ChannelWidth)
    foreach signal in otherChannelSignals do
        sigRating ← otherSignal.RSSI
        if otherBssid ∈ groupState then
            // inner interface
            sigRating ← sigRating + groupState[otherBssid].TxDiff
        // sigRating (in dBm) is normalized to [0, 1] scale
        sigRating ← scale(sigRating)
        cumInterf ←
          ciScore × (sigRating + clientsWeight × clients[bssid].len − otherSignal.noiseFloor)
return cumInterf
```

ALGORITHM 8: RRMGreedy++

Input: ***scandata***: scanning data from access points

G: set of wireless interfaces within the RRM group and its settings

Output: ***resultConfig***: final configuration for access points

groupState ← PreprocessRRMData(***G***, scandata)

`// Step 1: Greedy Channel Search`

initialGroupInterference ← nil

groupInterference ← GroupInterference(groupState)

repeat

  for *each interface in* ***G*** **do**

    $\boldsymbol{C_i}$ ← available channels for interface

    newChannel ← $\operatorname{argmin}_{ch \in \boldsymbol{C_i}} OnIfaceInterferencePlusPlus(groupState, interface, ch)$

    groupState[interface].channel ← newChannel

  groupInterference ← GroupInterference(scandata)

until *no improvement in groupInterference*

`// Step 2: Assume max Tx power for each interface`

for *each interface in* ***G*** **do**

  groupState[interface].TxDiff ← $(interface.MaxTxPower - interface.TxPower)$

`// Step 3: Greedy Transmission Power Adjustment (if enabled)`

repeat

  `// Identify the "worst interface"` $interface_{worst}$ `causing the greatest interference for other APs`

  $interface_{worst} \leftarrow \operatorname{argmax}_{w_i \in G} FromIfaceInterference(groupState, w_i)$

  $groupState[interface_{worst}].TxDiff \leftarrow \frac{interface_{worst}.TxPower - interface_{worst}.MinTxPower}{2}$

  `groupInterference ← GroupInterference(scandata)`

until *no improvement in groupInterference*

`// Step 4: Send configuration changes to APs`

resultConfig ← getUpdatedConfiguration(groupState)

return ***resultConfig***

IV Conclusion

In this chapter, I have performed an analysis of a production RRM algorithm called *RRMGreedy*, used by a leading Russian telecommunications vendor. By analyzing the source code of *RRMGreedy*, I have estimated the asymptotic complexity and found several issues potentially affecting the efficiency of the algorithm. To address those issues, I propose a set of enhancements for *RRMGreedy*, together called *RRMGreedy*++. In Chapter 5, I proceed with experimental evaluation of the RRM algorithms, including *RRMGreedy*++, by introducing a new framework for ns-3 network simulator I have created to facilitate RRM assessment.

Chapter 4

Implementation and Evaluation

This chapter is dedicated to implementing the *RRMGreedy++* algorithm described in Section 3.3, as well as other algorithms described in Section 2 and comparing their performance. In this study, I use the ns-3 discrete-event simulation framework to implement and evaluate different RRM algorithms.

The chapter is organized as follows:

- Section 4.1 describes the simulation methodology used to evaluate the performance of RRM algorithms, and the process of adjusting ns-3 to needs of running and evaluating RRM algorithms;
- Section 4.2 describes my implementation of *RRMGreedy* and *RRMGreedy++*;
- Section 4.3 presents the results of the simulation study, comparing RRM algorithms: *RRMGreedy* and *RRMGreedy++*;

- Finally, Section 4.4 concludes the chapter with a summary of the results and a discussion of the implications of the study.

I Simulation methodology

In this study, I implement RRMGreedy and RRMGreedy++ algorithms in ns-3 simulator to have a scalable, flexible (from dozens to hundreds of APs), and reproducible (unlike real-world testbeds) way to evaluate performance of different RRM algorithms under different conditions.

A. RRM support in ns-3

The ns-3 simulator is a comprehensive tool for research in computer networks, modeling each layer of the OSI model and providing simulations for wired and wireless communication technologies and protocols, including LTE, Wi-Fi, and Ethernet. However, despite its extensive capabilities, there is limited support for simulating RRM algorithms.

To address this gap, [35] introduced an RRM framework that employs an additional radio interface, termed a "scan interface". Unlike the standard "data interface" used for communication, the scan interface periodically surveys a specified set of channels. It operates in *promiscuous mode*, allowing it to capture all frames and physical layer headers intercepted by the data interface. Thus, the scan interface is able to record Received Signal Strength Indicator (RSSI) values from transmissions between devices with which the data interface is engaged. This resembles the scanning radios found in high-end Cisco APs. However, unlike the Cisco solution, this scanning interface can be used with both APs and stations.

However, this model presents several deviations from real-world scenarios.

- Most hardware, apart from high-end Cisco models, lacks a dedicated scanning interface, leading to scanning activities disrupting normal data exchanges. This interruption, referred to as *"dead time,"* denotes periods when scanning supersedes the standard operation of an Access Point (AP);
- For mobile devices, scanning capabilities are significantly restricted to avoid draining the battery;
- The `RriModule` code and its associated examples, designed for ns-3 version 3.20, are incompatible with the current ns-3 version 3.40 API and the legacy *Waf* build system. As a part of my contribution, I have refactored the original source code to align with the latest ns-3 version 3.41 API, specifically updating the `RriModule` to integrate with the latest `WifiMac` ns-3 API and the *CMake* build system. This adaptation ensures that modules and examples from [35] are fully operational on the newest ns-3 version. The modified source code is publicly available on GitHub[1].

B. Providing scanning capabilities in ns-3

In real-world scenarios, wireless APs implement scanning by setting the wireless adapter to monitor mode and sequentially switching to each channel that requires scanning. Although direct support for monitor mode is absent in ns-3, a comparable functionality can be simulated using the `MonitorSnifferRx` event source. Subscribing to such events allows the interception of all frames that the physical layer of the access point can receive and demodulate, even if those frames would normally be discarded as not intended for the current AP [36]. However, this method does not exclude the standard operations of an access point; the AP

[1]`https://github.com/ar7ch/ns-3-dev`

still continues to receive and transmit frames. This is not the case in actual access points, where operating monitor mode is mutually exclusive to AP's normal operation. To achieve nearly the same result, my approach prevents the AP from transmitting frames during the scanning period.

Based on the above consideration, I designed an external `Scanner` class that encloses `WifiNetDev` of access point nodes. The `Scanner` can be used on arbitrary AP node to give it background scanning capabilities. The `Scanner` class is invoked on `MonitorSnifferRx` event, simulating receiving frames in monitor mode. For each received frame, if this frame comes from an AP (either a beacon frame or has `FromDs` flag set to 1), it is added to `knownAps` table. If scheduled, the `Scanner` periodically switches to other channels for specified *channel dwell period*, listening for frames. To reduce disruptions in AP's regular BSS operation, the AP returns to its operating channel after performing scan on one channel, proceeding to scan the next channel after a specified *scan interval*. This behavior is also implemented in Cisco access points [1]. Based on this frame capture, Scanner builds a table of known APs, containing information such as BSSID, operating channel, RSSI and SNR levels, and the number of client stations associated with the AP. This table is then used by the RRM algorithms to make decisions on channel and power adjustments.

Another limitation I have encountered during the implementation is the lack of channel switching logic for unassociated stations that perform scans to discover an AP to associate with. Real-world stations scan all available channels within the operating bands to discover APs; however, as of the latest ns-3 version 3.41, released in February 2024, such functionality is absent in `ns3::StaWifiMac` class, which implements the logic for STA (stations) operation. Consequently, ns-3 Wi-Fi stations can only scan for APs present on the current operating channel. For instance, if both the AP and the stations are on channel 1, and the AP shifts to channel 6, the stations remain unable to scan beyond channel 1 to discover the AP's new location on channel 6. To address this problem, I have implemented a custom scanning logic, allowing stations to scan all available channels in their operating band. This modification is available in the modified source code on GitHub[2].

II Implementation

This section provides an implementation of various Radio Resource Management (RRM) strategies, utilizing the ns-3 discrete-event simulation framework.

A. RRMGreedy

RRMGreedy is a centralized super-cell algorithm used by a leading Russian telecommunications vendor, described and analyzed in Chapter 3.2. Implementing RRMGreedy in ns-3 requires a centralized entity to collect data from all APs and perform computations. I introduce `RRMGreedyAlgo` class, which is responsible for collecting data from all APs and making decisions on channel and power adjustments.

Unlike LCCS, RRMGreedy is not invoked by APs due to its centralized nature. Since the original RRMGreedy does not request scanning data directly from APs, but rather queries the database to which APs send periodic reports, I aimed to achieve similar behavior. Thus, `RRMGreedyAlgo::AddApScandata` is set as a callback for each `Scanner` instance in an RRM group to submit the individual scan results to RRMGreedy after every full scan cycle. Each

[2] `https://github.com/ar7ch/ns-3-dev/commit/84c93b474796ed038e3deaf0d0a9ecfb876eb9c8`

`scandata` item has a timestamp, and RRMGreedy does not accept scanning data older than *scandataStaleTime* parameter (in seconds). Callback method `RRMGreedyAlgo::Decide` is invoked by the simulation scheduler at regular intervals to make RRM decisions. Channel planning is implemented by `Decide()` callback and invokes the `RRMGreedy` algorithm described in 1. It is worth mentioning that implementing TPC was challenging, since ns-3 does not support setting AP's transmit power directly. To address this issue, I manipulated the starting and ending transmit power levels of the PHY layer of a `WifiNetDevice` at an AP node.

B. RRMGreedy++

I implemented *RRMGreedy*++ in a class `RRMGreedyPlusPlusAlgo` derived from `RRMGreedyAlgo`. Methods `Decide()`, triggering RRM start, `OnIfaceInterference()`, and `ChannelInterference` were overriden and updated according to the algorithm explained in Section 3.3.

III Evaluation

To evaluate the performance of RRM algorithms, I have conducted a series of test cases for simulations in ns-3. These test case share several common parameters that are fixed (Table I); each test case is also characterized by a set of unique parameters (Table II)

A. Simulation settings

TABLE I
Fixed Simulation Parameters

Parameter	Value
Frequency band	2.4 GHz
IEEE 802.11 amendment	802.11n
Data rate	54 Mbit/s
Channel width	20 MHz
Random seed	2

TABLE II
Test Case Configurations

Test Case 1	
Parameter	**Value**
Number of APs	3
Number of STAs	7
Number of STA allocated for each AP	$\{3, 2, 2\}$
Initial channel allocations for APs	$\{1, 1, 1\}$
Initial Tx power allocations for APs (dBm)	$\{20, 20, 20\}$
Initial channel allocations for STAs	All on channel 1
Traffic model	each AP has at least one pair of STAs, where first STA in the pair generates UDP traffic as client, and second echoing it back to client as UDP server

Simulation time (seconds)	10
Offered load per flow (Mbit/s)	3.28
Test Case 2	
Parameter	**Value**
Number of APs	4
Number of STAs	7
Number of STA allocated for each AP	$\{3, 2, 2, 2\}$
Initial channel allocations for APs	$\{1, 1, 1, 1\}$
Initial Tx power allocations for APs (dBm)	$\{20, 20, 20, 20\}$
Initial channel allocations for STAs	All on channel 1
Traffic flows and traffic model	Each AP has at least one pair of STAs, where first STA in the pair generates UDP traffic as client, and second echoing it back to client as UDP server
Simulation time (seconds)	10
Offered load per flow (Mbit/s)	2.73
Test Case 3	
Parameter	**Value**
Number of APs	8
Number of STAs	8
Number of STA allocated for each AP	$\{6, 0, 0, 0, 0, 1, 1, 0\}$

Initial channel allocations for APs	$\{1, 6, 6, 6, 6, 11, 11, 11\}$
Initial Tx power allocations for APs (dBm)	$\{20, 20, 20, 20, 20, 20, 20, 20\}$
Initial channel allocations for STAs	All on channel 1
Traffic flows and traffic model	If an AP has more than 1 clients, then clients are split in pairs, where the first STA in the pair generates UDP traffic as client, and the second echoing it back to the client as an UDP server
Simulation time (seconds)	10
Offered load per flow (Mbit/s)	8.192

B. Metrics

For evaluation, I collect various L3 and L2 metrics. Table III contains collected metrics, their definitions, and methods to capture them using ns-3.

TABLE III
Gathered Metrics

Metric	**Definition**	**Means of capture**
Average Total WLAN Throughput	Number of L3 IPv4 Rx bytes in WLAN divided by Tx time for all nodes	ns-3 `FlowMonitor` class
Average L3 Delay	Average time delta between L3 (IPv4) Tx and Rx events	ns-3 `FlowMonitor` class

Average Total WLAN SNR (Signal-to-Noise Ratio)	Average SNR of frames received by every node in WLAN	ns-3 `MonitorSnifferRx` event
Average Medium Busy Time	Total time when Wi-Fi device could not transmit due to occupied medium	ns-3 `YansWifiPhy/State` trace source, notifies when Wifi PHY enters `CCA_BUSY` state
Average Medium Idle Time	Total time for Wi-Fi device both not receiving and transmitting	ns-3 `YansWifiPhy/State` trace source, notifies when Wifi PHY enters `IDLE` state

C. Results

The simulation results are presented below in tabular (Table IV) form.

TABLE IV
Simulation Results

Test Case 1					
Algorithm	**Throughput (Mbit/s)**	**Average Delay (sec.)**	**Average Medium Busy Time (sec.)**	**Average Medium Idle Time (sec.)**	**Average SNR (dBm)**

No RRM	8.7147	0.1204	0.6409	0.6846	64.4698
RRMGreedy	24.2725	0.0046	0.2227	1.0998	73.7358
RRMGreedy++	24.9315	0.0184	0.2274	1.0939	73.4875
Test Case 2					
Algorithm	**Throughput (Mbit/s)**	**Average Delay (sec.)**	**Average Medium Busy Time (sec.)**	**Average Medium Idle Time (sec.)**	**Average SNR (dBm)**
No RRM	8.7147	0.1204	0.6409	0.6846	64.4698
RRMGreedy	24.2725	0.0046	0.2227	1.0998	73.7358
RRMGreedy++	24.9315	0.0184	0.2274	1.0939	73.4875
Test Case 3					
Algorithm	**Throughput (Mbit/s)**	**Average Delay (sec.)**	**Average Medium Busy Time (sec.)**	**Average Medium Idle Time (sec.)**	**Average SNR (dBm)**
No RRM	18.3974	0.0082	0.5268	0.8559	51.7041
RRMGreedy	26.6212	0.0007	0.1756	1.2072	56.3327
RRMGreedy++	26.6417	0.0006	0.1715	1.2127	59.7644

IV Conclusion

Based on the results, I can suggest that RRM considerably improves the performance of the WLAN. One can see the tremendous improvement of throughput when APs were moved to another channel in Test Case 2. Since the allocation is trivial in this case — 3 APs, 3 orthogonal channels available — the metrics for *RRMGreedy* and *RRMGreedy++* are almost the same.

That is especially evident when dealing with multiple APs on a single channel and high offered load, as in Test Case 3. The lower medium busy time and the higher medium idle time can be interpreted as the consequence of the medium becoming less congested.

Moreover, RRMGreedy and RRMGreedy++ increase the total SNR of the WLAN as a result of adjusting the transmission power and switching APs to orthogonal channels.

Chapter 5

Discussion and Conclusion

This chapter discusses the results of the study. Section 5.1 summarizes the results and contribution of this study, and outlines the direction for further studies. Section 5.2 discusses the limitations of the study. Section 5.3 outlines the further work in the direction of this research. Section 5.4 concludes the chapter.

I Simulation Results

RRMGreedy++ showed better results compared to *RRMGreedy* in terms of radio resource utilization optimization: the updated algorithm leads to a bandwidth improvement of up to 12% and an increase in the signal-to-noise ratio by 10%.

II Limitations

Using the simulated environment provides a controlled and reproducible environment for evaluation, however, networking and physical signal propagation models used pose their constraints. For example, `YansWifiPhy` propagation model does not consider cross-channel interference between adjacent channel, which can significantly impact channel planning, especially on 2.4 GHz band.

III Further Work

As comes from the limitations (Section 5.2), I can identify two directions for the following research:

1. Improving the simulation model: the next steps would be to extend the simulation environment to consider more realistic scenarios. This includes employing cross-channel interference model SpectrumWifiPhy, considering the impact of hidden nodes, and introducing a wireless LAN controller (WLC) as a part of the simulated network, since current implementation works "above" the nodes, being a part of the script logic;

2. Experimenting with another channel selection approaches: namely, I consider channel planning using genetic algorithms [24] to be a prospective solution with performance on-par with exact optimization-based RRM algorithms [24];

3. Implementing the RRMGreedy algorithm in a real-world test bed: since the algorithm aligns with constraints described in Section 2.5, metrics required for its operation can be obtained from the majority of OpenWRT-based Wi-Fi access points available on the market.

IV Conclusion

In this thesis, I derived an improved version of *RRMGreedy* Radio Resource Management (RRM) algorithm, *RRMGreedy*, and implemented evaluation of these algorithms using ns-3 network simulator. To achieve this, I performed analysis of *RRMGreedy*, and identified its flaws. Moreover, since ns-3 does not provide facilities for implementing RRM algorithms out of the box, and since Wi-Fi station channel switching logic was not implemented as of current ns-3 release, I have introduced a novel RRM framework for ns-3 and a set of enhancements into the existing codebase. Finally, the algorithms were evaluated using various simulation settings. The results of the simulation show that *RRMGreedy*++ outperforms *RRMGreedy* in terms of SNR, throughput, and spectrum management. The further work in this direction would be making the simulation model more accurately represent real-world deployments, experimenting with optimization approaches and heuristics other than greedy search, and evaluating *RRMGreedy* on a real-world test bed.

Bibliography Cited

[1] Cisco. "Radio Resource Management White Paper," Cisco. (), [Online]. Available: https://www.cisco.com/c/en/us/td/docs/wireless/controller/technotes/8-3/b_RRM_White_Paper.html (visited on 09/23/2023).

[2] "Understanding ARM." (), [Online]. Available: https://www.arubanetworks.com/techdocs/ArubaOS_62_Web_Help/Content/ArubaFrameStyles/ARM/ARM_Overview.htm (visited on 01/16/2024).

[3] "Cisco Unified Wireless Network Solution Overview," Cisco. (), [Online]. Available: http://www.cisco.com/en/US/docs/solutions/Enterprise/Mobility/emob30dg/Overview.html (visited on 03/19/2024).

[4] D. D. Coleman and D. A. Westcott, *CWNA Certified Wireless Network Administrator Study Guide: Exam CWNA-108*, 6th edition. Sybex, Mar. 9, 2021, 1088 pp., ISBN: 978-1-119-73450-5.

[5] A. S. Tanenbaum, *Computer Networks*, 6th edition. Pearson, Apr. 24, 2020, 922 pp., ISBN: 978-0-13-676405-2.

[6] J. Zander, "Radio resource management in future wireless networks: Requirements and limitations," *IEEE Communications Magazine*, vol. 35,

no. 8, pp. 30–36, Aug. 1997, ISSN: 1558-1896. DOI: 10.1109/35.606024.

[7] "802.11r-2008 - 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 2: Fast Basic Service Set (BSS) Transition | IEEE Standard | IEEE Xplore." (), [Online]. Available: https://ieeexplore.ieee.org/document/4573292 (visited on 03/19/2024).

[8] "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 - Spectrum and Transmit Power Management Extensions in the 5 GHz Band in Europe," *IEEE Std 802.11h-2003 (Amendment to IEEE Std 802.11, 1999 Edn. (Reaff 2003))*, pp. 1–75, Oct. 2003. DOI: 10.1109/IEEESTD.2003.94393.

[9] A. Könsgen, "Spectrum Management Algorithms," in *Design and Simulation of Spectrum Management Methods for Wireless Local Area Networks*, A. Könsgen, Ed., Wiesbaden: Vieweg+Teubner Verlag, 2010, pp. 103–132, ISBN: 978-3-8348-9738-1. DOI: 10.1007/978-3-8348-9738-1_4.

[10] "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 1: Radio Resource Measurement of Wireless LANs," *IEEE Std 802.11k-2008 (Amendment to IEEE Std 802.11-2007)*, pp. 1–244, Jun. 2008. DOI: 10.1109/IEEESTD.2008.4544755.

[11] "802.11ax-2021 - IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Enhancements for High-Efficiency WLAN | IEEE Standard | IEEE Xplore." (), [Online]. Available: `https://ieeexplore.ieee.org/document/9442429` (visited on 06/05/2024).

[12] "Cisco Catalyst 9800 Series Wireless Controller Software Configuration Guide, Cisco IOS XE Bengaluru 17.4.x - BSS Coloring [Cisco Catalyst 9800 Series Wireless Controllers]," Cisco. (), [Online]. Available: `https://www.cisco.com/c/en/us/td/docs/wireless/controller/9800/17-4/config-guide/b_wl_17_4_cg/m_bss_coloring_ewlc.html` (visited on 03/20/2024).

[13] K. Sui, Y. Zhao, D. Pei, and L. Zimu, "How bad are the rogues' impact on enterprise 802.11 network performance?" *2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 361–369, Apr. 2015. DOI: `10.1109/INFOCOM.2015.7218401`.

[14] A. Levanti, F. Giordano, and I. Tinnirello, "A CAPWAP-Compliant Solution for Radio Resource Management in Large-Scale 802.11 WLAN," in *IEEE GLOBECOM 2007-2007 IEEE Global Telecommunications Conference*, Washington, DC, USA: IEEE, Nov. 2007, pp. 3645–3650, ISBN: 978-1-4244-1042-2 978-1-4244-1043-9. DOI: `10.1109/GLOCOM.2007.692`.

[15] F. Bouhafs, M. Seyedebrahimi, A. Raschella, M. Mackay, and Q. Shi, "Per-Flow Radio Resource Management to Mitigate Interference in Dense IEEE 802.11 Wireless LANs," *IEEE Trans. on Mobile Comput.*, vol. 19, no. 5,

pp. 1170–1183, May 1, 2020, ISSN: 1536-1233, 1558-0660, 2161-9875. DOI: 10.1109/TMC.2019.2903465.

[16] W. Lee and J.-B. Seo, "Deep Learning-Aided Channel Allocation Scheme for WLAN," *IEEE Wireless Commun. Lett.*, vol. 12, no. 6, pp. 1007–1011, Jun. 2023, ISSN: 2162-2337, 2162-2345. DOI: 10.1109/LWC.2023.3257128.

[17] M. Achanta, "Method and apparatus for least congested channel scan for wireless access points," U.S. Patent 20060072602A1, Apr. 6, 2006. [Online]. Available: https://patents.google.com/patent/US20060072602A1/en (visited on 11/16/2023).

[18] Arunesh Mishra, A. Mishra, Suman Banerjee, *et al.*, "Weighted coloring based channel assignment for WLANs," *Mobile Computing and Communications Review*, vol. 9, no. 3, pp. 19–31, Jul. 1, 2005. DOI: 10.1145/1094549.1094554.

[19] M. Michalski and K. Staniec, "A simple performance-boosting algorithm for transmit power control in WLAN access points," *2016 21st International Conference on Microwave, Radar and Wireless Communications (MIKON)*, pp. 1–4, May 9, 2016. DOI: 10.1109/mikon.2016.7492075.

[20] .. and.., ",", vol. 7, no. 3, pp. 25–37, 2021, ISSN: 1813-324X, 2712-8830. DOI: 10.31854/1813-324X-2021-7-3-25-37.

[21] K. Ramachandran, R. Kokku, H. Zhang, and M. Gruteser, "Symphony: Synchronous two-phase rate and power control in 802.11 wlans," *IEEE/ACM Transactions on Networking*, pp. 132–145, Jun. 17, 2008. DOI: 10.1145/1378600.1378616.

[22] Y. Lee, K. Kim, and Y. Choi, "Optimization of AP placement and channel assignment in wireless LANs," in *27th Annual IEEE Conference on Local Computer Networks, 2002. Proceedings. LCN 2002.*, Nov. 2002, pp. 831–836. DOI: `10.1109/LCN.2002.1181869`.

[23] R. Rodrigues, G. Mateus, and A. Loureiro, "On the design and capacity planning of a wireless local area network," in *NOMS 2000. 2000 IEEE/IFIP Network Operations and Management Symposium 'The Networked Planet: Management Beyond 2000' (Cat. No.00CB37074)*, Apr. 2000, pp. 335–348. DOI: `10.1109/NOMS.2000.830394`.

[24] A. Raschellà, M. Mackay, F. Bouhafs, and B. I. Teigen, "Evaluation of Channel Assignment Algorithms in a Dense Real World WLAN," in *2019 4th International Conference on Computing, Communications and Security (ICCCS)*, Oct. 2019, pp. 1–5. DOI: `10.1109/CCCS.2019.8888082`.

[25] "WiFi Market - Size, Share & Trends." (), [Online]. Available: `https://www.mordorintelligence.com/industry-reports/wi-fi-market` (visited on 11/16/2023).

[26] S. Arena, F. S. Crippa, N. Darchis, and S. Katgeri, *Understanding and Troubleshooting Cisco Catalyst 9800 Series Wireless Controllers*, 1st edition. Cisco Press, Jul. 25, 2022, 656 pp., ISBN: 978-0-13-749232-9.

[27] "Cisco CleanAir Technology At-a-Glance," 2014.

[28] JuniperNetworks, director, *Understanding Radio Resource Management RRM*, Jul. 19, 2023. [Online]. Available: `https://www.youtube.com/watch?v=BUmhVytJfXI` (visited on 10/31/2023).

[29] "Radio Management (technology) | Mist | Juniper Networks." (), [Online]. Available: https://www.juniper.net/documentation/us/en/software/mist/mist-wireless/topics/concept/wireless-rrm.html (visited on 11/16/2023).

[30] "Ruckus ChannelFly Feature Sheet." (Jan. 24, 2023), [Online]. Available: https://web.archive.org/web/20230124150501/https://webresources.ruckuswireless.com/pdf/feature-sheets/fs-channelfly.pdf (visited on 11/16/2023).

[31] "ArubaOS 8.6.0.0 User Guide," PSNow. (), [Online]. Available: https://www.hpe.com/psnow/doc/a00092459en_us (visited on 01/16/2024).

[32] "ARM Overview." (), [Online]. Available: https://www.arubanetworks.com/techdocs/ArubaOS_60/UserGuide/ARM.php (visited on 01/16/2024).

[33] "Taming Aruba ARM | Wifizoo." (Nov. 9, 2012), [Online]. Available: https://wifizoo.org/2018/01/11/taming-aruba-arm/ (visited on 01/16/2024).

[34] "ARM Metrics." (), [Online]. Available: https://www.arubanetworks.com/techdocs/ArubaOS_62_Web_Help/Content/ArubaFrameStyles/ARM/ARM_Metrics.htm (visited on 11/01/2023).

[35] K. Bharadwaj, S. Kalpalatha, and V. Ramaiyan, "A simulation framework for radio resource management in WiFi networks," in *2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Bhubaneswar: IEEE, Dec. 2017, pp. 1–6, ISBN: 978-1-5386-2347-3. DOI: 10.1109/ANTS.2017.8384106.

[36] "Ns-3: All TraceSources." (), [Online]. Available: https://www.nsnam.org/docs/release/3.41/doxygen/de/de5/_trace_source_list.html (visited on 05/06/2024).

[37] "The Authoritative Dictionary of IEEE Standards Terms, Seventh Edition," *IEEE Std 100-2000*, pp. 1–1362, Dec. 2000. DOI: 10.1109/IEEESTD.2000.322230.

Appendix A

Used Terms and Definitions

TABLE V
Used Terms and Definitions

Term	**Definition**
Signal	Airborne RF energy [4]
Channel	A band of frequencies that 802.11 devices can use for communications [37]
Inteference	Destructive influence of another signal leading to degradation of signal quality and loss of frames
Noise	RF signal that cannot be demodulated as 802.11 signal
SNR (Signal-to-Noise Ratio)	Signal quality metric, defined as ratio of signal power to the noise
SINR (Signal-to-Interference-plus-Noise Ratio)	Signal quality metric, defined as ratio of signal power to the sum of noise power and power of in-terefering signals

RSSI (Received Signal Strength Indicator)	A metric of a wireless signal quality defined in 802.11, varying from 0 to 255, where exact mappings to Rx signal power are vendor-specific.
Radio cell	Geographical area covered by a radio transmitter [5]
Basic Service Set (BSS)	Set of stations belonging to the same radio cell and exchanging information [9]
Distribution System (DS)	Network that interconnects multiple BSSs and provides connectivity to a wired network [9]
Extended Service Set (ESS)	Set of Basic Service Sets connected by a DS [9]
Station (STA)	Client device that can connect to a wireless LAN
Access Point (AP)	Device providing stations wireless access to a distribution system
Infrastructure mode	Centralized mode of 802.11 WLAN operation using a star topology, where AP serves as a central entity for managing WLAN and switching traffic
Network capacity	Maximum transmission rate that any station can achieve in a given WLAN
Tx	Short notation for "Transmission" or "Transmit"
Rx	Short notation for "Receipt" or "Receive"