UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

José Joaquim de Andrade Neto

Formulações e Heurísticas Para Duas Variantes do Problema do Escalonamento em Redes Sem Fio

Belo Horizonte
2022-04

José Joaquim de Andrade Neto

Formulações e Heurísticas Para Duas Variantes do Problema do Escalonamento em Redes Sem Fio

Versão Final

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Thiago Ferreira de Noronha
Coorientador: Marcos Augusto Menezes Vieira

Belo Horizonte
2022-04

José Joaquim de Andrade Neto

Formulations and Heuristics for two variants of the Wireless Scheduling Problem

Final Version

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Thiago Ferreira de Noronha
Co-Advisor: Marcos Augusto Menezes Vieira

Belo Horizonte
2022-04

© 2022, José Joaquim de Andrade Neto.
. Todos os direitos reservados

Andrade Neto, José Joaquim de.

A554f Formulações e heurísticas para duas variantes do problema do escalonamento em redes sem fio [manuscrito] /José Joaquim de Andrade Neto — Belo Horizonte, 2022.
60 f. il.; 29 cm.

Orientador: Thiago Ferreira de Noronha.
Coorientador: Marcos Augusto Menezes Vieira.
Dissertação (mestrado) - Universidade Federal de Minas Gerais – Departamento de Ciência da Computação
Referências: f. 57-60.

1. Computação – Teses. 2. Programação (Matemática) – Teses. 3. Algoritmos – Teses. I.Noronha, Thiago Ferreira de. II. Vieira, Marcos Augusto Menezes. III . Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Computação. IV.Título.

CDU 519.6*52 (043)

Ficha Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa - CRB 6/1510 - Universidade Federal de Minas Gerais - ICEx

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

FORMULAÇÕES E HEURÍSTICAS PARA DUAS VARIANTES DO PROBLEMA DE ESCALONAMENTO EM REDES SEM FIO

JOSÉ JOAQUIM DE ANDRADE NETO

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

Prof. Thiago Ferreira de Noronha - Orientador
Departamento de Ciência da Computação - UFMG

Prof. Marcos Augusto Menezes Vieira - Coorientador
Departamento de Ciência da Computação - UFMG

Profa. Olga Niklaevna Goussevskaia
Departamento de Ciência da Computação - UFMG

Prof. Martín Goméz Ravetti
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 4 de abril de 2022.

---

sei! assinatura eletrônica

Documento assinado eletronicamente por **Thiago Ferreira de Noronha**, **Professor do Magistério Superior**, em 18/04/2022, às 13:49, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Marcos Augusto Menezes Vieira**, **Professor do Magistério Superior**, em 18/04/2022, às 14:52, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Olga Nikolaevna Goussevskaia**, **Professora do Magistério Superior**, em 18/04/2022, às 16:49, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Martin Gomez Ravetti**, **Professor do Magistério Superior**, em 20/04/2022, às 08:48, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1390544** e o código CRC **3AAB6D91**.

For everyone who supported me in this cycle.

Acknowledgments

The closure of this master thesis represents the ending of a long journey that took more time than was expected. The world has its strange way of challenging us a little bit more every day. Since the beginning of this journey, I have faced up with several doubts about the capacity and limitations of my progress. Nevertheless, my love for computer science and my curiosity for learning has always driven me to new adventures (such as the non-linear optimization course!). At the same time, I am one hundred percent sure that this "series finale" is actually happening only because of the support of some group of people that shall be enumerated.

My dear friends at the *Laboratório de Pesquisa Operacional*, that accepted me with a true friendship. So many insightful ideas, exchanged knowledge, tips for homework, honest hugs, and smiles. I am very grateful for the times we spent together. I am already missing the coffee break!

My teachers: Vinícius, Alexandre, Gabriel, Sebastián, and Robson. I appreciate all the patience, wisdom, and the passion you all have for teaching. Every single lecture was outstanding and motivated me to seek more.

The Federal University of Minas Gerais community, especially the Department of Computer Science and its staff, and the CAPES agency, which granted me with a scholarship during two years.

My co-advisor, Marcos Vieira, that patiently helped me and gave valuable advices that helped me to build a work that has its purpose for the computer network field.

Last but not least, to the person that made all this possible, my advisor Thiago Noronha. It takes courage to accept a stranger as a master student, and yet thanks to your empathy, patience, wisdom, and friendship above all, we reached the end. I must say that this journey was brilliant and full of rewards.

Thank you.

“*As long you are learning, you are not failing.*”

(Bob Ross)

Resumo

Novos protocolos de rede sem fio, a exemplo do IEEE 802.11ax (Wi-Fi 6), permitem que o espectro eletromagnético seja particionado em um ou mais canais de comunicação cuja largura de banda pode variar de 20 MHz a 160 MHz. Esse particionamento permite conexões com interferências menores e velocidades de transmissão maiores. Entretanto, particionar o espectro de forma ótima ou até mesmo de forma factível, em casos extremos, é visto como um desafio no campo de projeto de algoritmos, uma vez que pode existir uma quantidade exponencial de maneiras que tal particionamento pode ser realizado.

Tendo em vista o contexto de algoritmos de escalonamento, esse presente trabalho aborda dois problemas considerados $\mathcal{NP}$-difícil que visam otimizar o escalonamento de conexões Wi-Fi. O primeiro, intitulado de *Variable Rate Variable Scheduling Problem* (VRBSP), é o problema de selecionar um subconjunto de conexões, um subconjunto de canais de comunicações e escalonar os links nos respectivos canais tal que o *throughput* da rede seja o maior possível. O segundo problema, intitulado de *Minimum-Delay Variable Rate Variable Scheduling Problem* (MD-VRBSP), busca associar um conjunto de conexões em um subconjunto de canais de comunicação para que todas conexões sejam satisfeitas utilizando o menor tempo possível. Técnicas exatas, tais como formulações de programação misto-inteira, e meta-heurísticas como *Variable Neighborhood Search* (VNS), foram utilizadas para buscar soluções ótimas ou quase ótimas para instâncias de tamanho médio e grande, respectivamente.

Esse presente trabalho adotou metodologias já conhecidas na literatura para criar um conjunto de instâncias de testes para a análise dos algoritmos propostos. Tais instâncias simulam redes sem fio que podem possuir até 2048 conexões a serem escalonadas. Os experimentos computacionais sugerem que, observando os tempos de execução de cada algoritmo, a formulação mista-inteira para o VRBSP encontrou *gaps* de otimalidade 19% menores, em média, quando comparada com outros algoritmos existentes na literatura. A heurítisica VNS, que teve uma performance superior à heurísticas da literatura, atingiu limites inferiores 250,63% melhores, em média, na maior instância, quando comparada com formulações misto-inteira da literatura. Além disso, os algoritmos exatos para o MD-VRBSP foram capazes de encontrar escalonamento ótimo para redes com até 64 conexões. Finalmente, os resultados observados para as heurísticas do MD-VRBSP sugerem que elas não foram efetivas quando comparadas com os algoritmos exatos, produzindo resultados 17,17% piores, em média.

Palavras-chave: Programação Matemática, Projeto de Algoritmos, Computação.

Abstract

Newer wireless protocols, such as the IEEE 802.11ax (Wi-Fi 6), allow for the electromagnetic spectrum to be partitioned into several communication channels with bandwidths varying from 20 MHz to 160 MHz. This feature allows for lower interferences and higher data-rates. However, it is now more difficult to cope with scheduling algorithms that consider this protocol, as there might exist an exponential number of ways to partition the Wi-Fi spectrum.

This present work tackles two related $\mathcal{NP}$-hard problems that attempt to optimize the schedule of Wi-Fi links. The first problem, the Variable Rate Variable Scheduling Problem (VRBSP), aims to schedule a subset of links and assign them to a subset of communication channels so that the transmission can occur with the highest possible throughput within a single time-slot. The second problem, the Minimum-Delay Variable Rate Variable Scheduling Problem (MD-VRBSP), aims to assign a set of links into a subset of communication channels so that all transmissions can occur using the lowest number of time-slots possible while respecting the minimum data-rate specifications of each link. This work adopts the use of Mixed-Integer Linear Programming (MILP) formulations to seek optimal solutions for small-sized instances and Variable Neighborhood Search (VNS) heuristics to find near-optimal solutions for all medium and large-sized instances.

This work adopted methodologies from the literature to create instance sets that represent wireless network environments with up to 2048 links. The computational experiments with these instances suggest that, given the respective execution times, the proposed MILP formulation for the VRBSP found optimality gaps of 19% below, on average, when compared with the existing formulation in the literature. The VNS heuristic, which outperformed baseline heuristics from the literature, achieved lower-bounds 250.63% higher, on average, in the largest instances, compared with the baseline MILP formulations. Besides, the MILP formulations for the MD-VRBSP were able to find optimal schedules for networks with up to 64 links. Moreover, the results observed by the MD-VRBRSP heuristics suggest that they generate solutions 17.17% worse, on average, in the largest instances, compared with MILP formulations.

Keywords: Mathematical Programming, Algorithm Design, Computing.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

The first chapter of this work attempts to motivate the reader about two problems that arise in the context of computer wireless networks. In what follows, this work introduces elements of wireless transmissions such as signal interference and wireless network protocols. Together, they form a basis for a precise formalization and definition of the two problems. Later on, this chapter also describes the general objective of this work, along with its research questions. Besides, the authors expose the expected contributions and the adopted methodology.

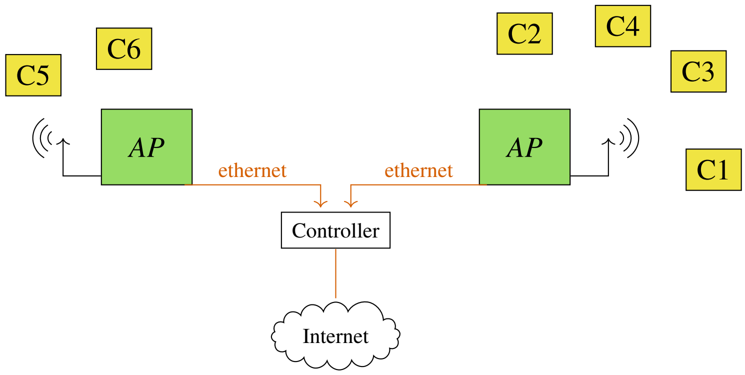
1.1 Motivation and Problem Overview

A connected world and an ever-increasing demand for online services lead to a run for better technologies and applications that yield a better user experience. Besides, conflicting requirements and the limited capacity of attending all requests force devices or applications to decide which related decisions should be taken. It is desired that such decisions be optimal or at least satisfy certain requirements. Therefore, operational research is an exciting approach to tackle such problems. Online meetings, streaming, or multiplayer gaming are examples of common applications that must meet certain requirements to work correctly. This present work takes further steps in this theme by proposing techniques to minimize the existent delay within wireless transmissions and to maximize the amount of data transferred, i.e., the throughput, in a given time window.

Most wireless Local Area Networks (WLAN) are networks in an infrastructure environment. This type of network became popular in certain types of environments such as hotels, malls, universities, industries, etc. WLAN networks consist of a set of Access Points (AP), devices responsible for transmitting and receiving wireless messages. Although connected to the Internet using a wired cable, APs can connect the clients to the Internet without using wires in both endpoints using a wireless protocol. The most common protocols in these networks belong to the 802.11 family of protocols. Figure 1.1 illustrates an example of a wireless network containing several APs and clients.

The APs can be interconnected, via cable, through a centralized entity called controller [Moura et al., 2015]. This controller is represented in Figure 1.1. Controllers are very common, for instance, in Software Defined Networks [Masoudi and Ghaffari, 2016, Chaudet and Haddad, 2013]. In this case, they represent a logical centralization, enabling the configuration of the APs and the management of the connections in the network. Moreover, it is possible to use this centralized organization to configure the APs to operate on a single-hop configuration, i.e., without the need for intermediary devices between them. Therefore, this approach motivates the design of algorithms to configure a set of access points.

Figure 1.1: An example of wireless network with many access points.



Source: own authorship.

A *link* in wireless networks can be modeled as a connection between a sender and a receiver device. The *signal quality* of a link is a measure of how strong is the signal that reaches the receiver of this link. The interference of the medium can degrade the signal quality when links are transmitting concurrently in overlapping frequencies. If the interference is high, the message might not be decoded by the receiver of the link. Moreover, the cleaner (i.e., less interference) is a transmission, the higher can be the data-rate as more symbols can be distinguished.

The Wireless Scheduling Problem (WSP) is a classical problem in the wireless network field, having intersections with the combinatorial optimization research field. Known initially as the *Link Scheduling Problem* [Baker et al., 1982], a large number of authors devoted their works to either study this problem or to introduce new variants. It was first stated as follows:

Definition 1.1 (Wireless Scheduling Problem). *Given is a set L of links, where each element $(i, j) \in L$ is a pair of sender and receiver. Using a single communication channel, select a subset of links $S \subseteq L$ that are allowed to transmit concurrently, given interference constraints. The objective is to minimize the number of time-slots needed for all links to transmit.*

Two important keys should be addressed regarding the WSP and its variants, that is, the interference among devices and the place where transmissions occur, namely the communication channel. The practical relevance of the model or its computational complexity varies according to these two choices. This work starts with the interference modeling discussion and then proceeds to communication channels.

The computation of the interference on a link is still an open challenge in the WSP literature, as it is hard to measure with precision the amount within an environment. According to Blough et al. [2010], existing interference models fall into two categories: primary and secondary. The former assumes that two links will only interfere with each other if and only if they share a common endpoint. The latter considers that two links can interfere with each other even with the lack of common endpoints because of the frequency spectrum that is shared between all devices, which leads to a better representation of what happens in reality.

This work considers WSP variants under the secondary interference model, called the Signal-to-interference-plus-noise Ratio (SINR). This is the ratio between the strength of the signal and the sum of all other signal strengths plus ambient noise [Gupta and Kumar, 2000]. As pointed by Voelker et al. [2009], this model is physically motivated and is believed to be reasonably realistic since it considers that the interference at the receiver can have other sources that are also emitting signals within the same environment. Indeed, the SINR model is the default interference model in the WSP literature. Under this scheme, a transmission is only considered successful if the SINR value, measured in dB, is above a determined threshold. Besides, higher data-rates can be achieved when having higher SINR values.

While the SINR model allows for a more realistic network modeling, the WSP under this interference scheme is a $\mathcal{NP}$-hard problem [Goussevskaia et al., 2007]. With no known polynomial technique to design algorithms for $\mathcal{NP}$-hard problems, most of the WSP works and its variants propose approximation algorithms [Goussevskaia et al., 2009, Halldórsson and Mitra, 2011b], heuristics [Vieira et al., 2016], and mixed-integer formulations [Costa et al., 2019, Kompella et al., 2007, Bjorklund et al., 2003].

The IEEE 802.11ax protocol, also known as the Wi-Fi 6 protocol, is considered in this work. This version introduces features that enable a more intelligent use of the Wi-Fi spectrum. Some of these features are well detailed in [Coleman, 2020]. Most of them are focused on reducing the interference of the network and reducing the delay between transmissions. Among several improvements, a more flexible partitioning of the frequency spectrum is the most relevant to this work.

This work addresses two WSP variants. They adopt the SINR model and simulate networks using the Wi-Fi 6, though any Wi-Fi version can be used. Namely, the Variable Rate and Variable Bandwidth Scheduling Problem (VRBSP) is a problem already studied in Costa et al. [2017, 2019], while the Minimum-Delay Variable Rate and Variable Bandwidth Scheduling Problem (MD-VRBSP) is first seen in this present work. They are can be summarized as follows:

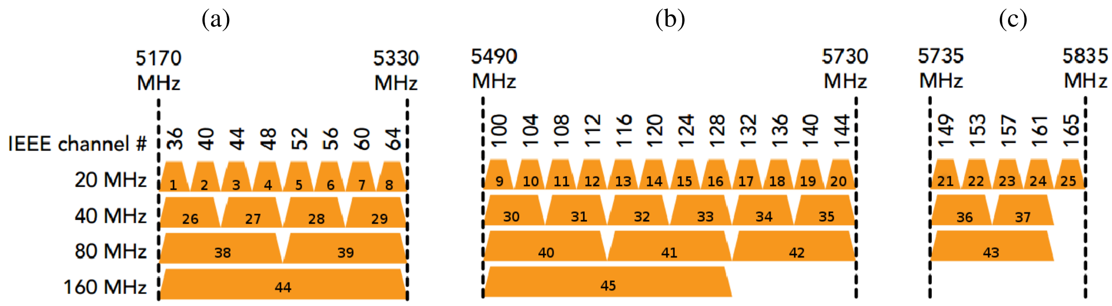
- The VRBSP is the problem of, given a single time-slot, to select a subset of links and assign them to a subset of communication channels, such that the resultant interference at each link is small enough so that transmissions occur successfully. Moreover, the selected schedule must be the one having the maximum throughput.

- The MD-VRBSP is the problem of scheduling all the links using the minimum number of time-slots, subject to each link to transmit using a data-rate that must be not less than the one specified for each link.

1.2 Problem Definition

In the Wi-Fi 6, transmissions occur within communication channels whose bandwidth can vary between 20 MHz, 40 MHz, 80 MHz, and 160 MHz. Figure 1.2 describes precisely all allowed partitions. The available spectrum is subdivided into three bands with 160, 240, and 100 MHz, respectively. The allowed partitions for each of these bands are given by Fig. 1.2a, 1.2b, and 1.2c, respectively. For example, the first band could be partitioned into the channels $\{1,2,27,39\}$ (i.e., two channels of 20 MHz, one of 40 MHz, and another of 80 MHz) or into the channels $\{26,27,28,29\}$ (i.e., four channels of 40 MHz). Similarly, the second band could be divided into the channels $\{42,45\}$ (i.e., one channel of 160 MHz and another of 80 MHz), or into the channels $\{40,41,42\}$ (i.e., three channels of 80 MHz). Likewise, the third band could be divided into the channels $\{25,36,37\}$ (i.e., one channel of 20 MHz and two of 40 MHz) or into the channels $\{24,45\}$ (i.e., one channel of 20 MHz and another of 80 MHz).

Figure 1.2: The subdivisions of the electromagnetic spectrum into communication channels defined by the Wi-Fi 6.



Source: Costa *et al.* (2019)

Let $V = \{1,2,\ldots,n\}$ be the set of devices, and let $L \subset V \times V$ be the set of links, such that a link $(i,j) \in L$ has a sender device i and a receiver device j. The set of admissible partitions of the frequency bands is defined as the power set of a set C of predefined channels. In the case of the Wi-Fi 6, $C = \{1,2,\ldots,45\}$ (see Fig. 1.2). These channels might overlap with each other. Therefore, consider a set $O^c \subseteq C$ with the channels that overlap with $c \in C$ (including itself).

For example, $O^{32} = \{13, 14, 32, 41, 45\}$ because channel 32 overlaps with channels 13, 14, 41, 45, and itself. The different bandwidths (in MHz) are defined by the set B, which in the case of the Wi-Fi 6 is equal to $B = \{20, 40, 80, 160\}$. In addition, let $C^b \subseteq C$ be the subset of channels whose bandwidth is $b \in B$. In the case of the Wi-Fi 6, $C^{20} = \{1, \ldots, 25\}$, $C^{40} = \{26, \ldots, 37\}$, $C^{80} = \{38, \ldots, 43\}$, and $C^{160} = \{44, 45\}$. Moreover, define $B^c \in B$ as the bandwidth of the channel $c \in C$. For example $B^1 = 20$, $B^{26} = 40$, $B^{38} = 80$, and $B^{44} = 160$ (see Fig. 1.2).

Consider $S \subseteq L \times C$ a schedule, where an element is in the form $\langle (i, j), c \rangle$ and means that link $(i, j) \in L$ is using channel $c \in C$ to transmit. Moreover, the resultant interference at link $\langle (i, j), c \rangle \in S$ is caused only by another link $\langle (u, v), c' \rangle \in S \setminus \{\langle (i, j), c \rangle\}$ that is simultaneously transmitting using some overlapping channel c'. This definition allows for the interference of a scheduled link to be measured using the definition of *affectance* [Halldórsson and Wattenhofer, 2009]:

Definition 1.2 (Affectance)**.** *The affectance a_{uj} of link $\langle (i, j), c \rangle \in S$ caused the sender u of link $\langle (u, v), c' \rangle \in S \setminus \{\langle (i, j), c \rangle\}$, with a given transmission power P, is the interference of (u, v) on receiver j:*

$$a_{ij} = \frac{P}{(d_{ij})^{\alpha}}, \tag{1.1}$$

where d_{ij} is the euclidean distance between $i \in V$ and $j \in V$ (in meters), and $\alpha \in \mathbb{R}$ is the path-loss exponent [Rappaport, 2002], with typical values in the range $2 \leq \alpha \leq 6$ [Goussevskaia et al., 2012, Gupta and Kumar, 2000, Goussevskaia et al., 2009].

Therefore, the total interference at the receiver j of scheduled link $\langle (i, j), c \rangle \in S$ is equal to:

$$I_{ij} = \sum_{\langle (u,v), c' \rangle \in S \setminus \{\langle (i,j), c \rangle\} : c' \in O^c} a_{uj} \tag{1.2}$$

Having the value of a_{ij} and I_{ij}, the signal quality $SINR_{ij}$ of the scheduled link $\langle (i, j), c \rangle \in S$ is given by Equation (1.3), where N is the ambient noise (in Watts), with typical values in the range $0 \leq N \leq 8 \cdot 10^{-14}$ [Halldórsson and Mitra, 2012, Vieira et al., 2016].

$$SINR_{ij} = \frac{a_{ij}}{I_{ij} + N} \tag{1.3}$$

The data transmission rate of a scheduled link $\langle (i, j), c \rangle \in S$, denoted by r_{ij}, is computed by mapping $SINR_{ij}$ and B^c to the *Modulation and Coding Scheme* (MCS) that results in the largest possible data transmission rate. The Wi-Fi 6 standard has 12 MCS. Let $M = \{0, \ldots, 11\}$ be the set of MCS identifiers, and denote by $\bar{q}^{bm}$ the minimum SINR that a link must have to be able to transmit with a data-rate of r^{bm} using a channel with bandwidth $b \in B$ and the MCS $m \in M$. The values of q^b_m and $\bar{r}^b_m$, for all $b \in B$ and $m \in M$, are given by Table 1.1.[1]

[1]This table was constructed mapping the values reported in the MCS index webpage and experimental results in the Qualcomm Inc. laboratories. They are available through the URL `https://mcsindex.net` and `https://bit.ly/2GoUSoc`, respectively. It was considered a noise of -84 dBm.

Table 1.1: SINR(dB) and Data-rate (Mbps) for each channel bandwidth used in the Wi-Fi 6.

MCS Index	20 MHz		40 MHz		80 MHz		160 MHz	
	SINR q_m^{20}	Data rate $\bar{r}_m^{20}$	SINR q_m^{40}	Data rate $\bar{r}_m^{40}$	SINR q_m^{80}	Data rate $\bar{r}_m^{80}$	SINR q_m^{160}	Data rate $\bar{r}_m^{160}$
0	2	8.6	5	17.2	8	36	11	72.1
1	5	17.2	8	34.4	11	72.1	14	144.1
2	7	25.8	10	51.6	13	108.1	16	216.2
3	10	34.4	13	68.8	16	144.1	19	288.2
4	14	51.6	17	103.2	20	216.2	23	432.4
5	18	68.8	21	137.6	24	288.2	27	576.5
6	19	77.4	22	154.9	25	324.3	28	648.5
7	20	86	23	172.1	26	360.3	29	720.6
8	25	103.2	19	206.5	31	432.4	34	864.7
9	27	114.7	30	229.4	33	480.4	36	960.8
10	30	129	33	258.1	36	540.4	39	1080.9
11	32	143.4	35	286.8	38	600.5	41	1201

Column 1 identifies the MCS, and Column 2 gives the minimum value of SINR $\bar{q}^{bm}$ necessary to transmit with a date rate of $\bar{r}_m^b$ (shown in Column 3) using a channel of bandwidth $b = 20$ and the MCS $m \in M$. The same data is reported for channels of bandwidth 40 MHz in columns 4 and 5, of bandwidth 80 MHz in columns 6 and 7, and of bandwidth 160 MHz in columns 8 and 9, respectively. For example, according to Table 1.1, a link $\langle (i, j), c \rangle \in S$ that is assigned to a channel of 80 MHz and that has $SINR_{ij} = 20.0$ dB transmits at a data-rate of 216.2 Mbps. If $SINR_{ij} \geq 36.0$ dB the same link would transmit at a data-rate of 540.0 Mbps. However, if $SINR_{ij} < 8.0$ dB the receiver of this link would not be able to decode the message, and this link could not be scheduled. Thus, given $SINR_{ij}$ and B^c, the transmission data-rate of $\langle (i, j), c \rangle \in S$ is given by Equation (1.4).

$$r_{ij} = \max_{m \in M : SINR_{ij} \geq q_m^{B^c}} \bar{r}_m^{B^c} \tag{1.4}$$

We can now formally define both VRBSP and MD-VRBSP:

Definition 1.3 (Variable Rate Variable Bandwidth Scheduling Problem)**.** *The VRBSP consists in selecting a subset of links in L and a subset of channels in C, and assigning each of the chosen links to one of the selected channels, such that (i) no two links that share a device are simultaneously chosen, and (ii) the interference at the receiver of each selected link is small enough so that the transmitted message can be decoded. A solution for this problem is represented by a schedule $S \subseteq L \times C$. The objective of this problem is to find the schedule that has the maximum throughput over all possible schedules.*

To put in another words, consider the set $\Delta \subset 2^{L \times C}$ of all *all feasible VRBSP schedules*, i.e., all schedules $S \in \Delta$ that respect conditions (i) and (ii) in Definition 1.3. The VRBSP is formally defined as in Equation 1.5.

$$S^* = \underset{S_{VR} \in \Delta_{VR}}{\text{argmax}} \sum_{\langle (i,j),c \rangle \in S} r_{ij} \tag{1.5}$$

This problem was first described as a Mixed-Integer Non-Linear Problem (MINLP) by Costa et al. [2019]. Moreover, the authors used linearization techniques to describe the same problem as two different Mixed-Integer Linear Problems (MILP). A new mixed-integer formulation is proposed and depicted in Chapter 3. It is believed that the new formulation can improve the LP bounds in relation to the original formulations.

The representation that is being used so far must be slightly extended to define the MD-VRBSP. Henceforth, consider S_{VR} as a solution for the VRBSP and Δ_{VR} the set of all feasible solutions as well. Besides, consider S_{MD} as a solution for the MD-VRBSP, where an element $\langle (i,j),(c,t) \rangle \in S_{MD}$ means that link $(i,j) \in L$ is scheduled to transmit at channel $c \in C$ and time-slot $t \in T$. The definition is as follows:

Definition 1.4 (Minimum-Delay Variable Rate Variable Bandwidth Scheduling Problem)**.** *The MD-VRBSP consists of selecting a subset of channels in C, a subset of time-slots T, and assigning each link $(i,j) \in L$ to one of the selected channels and one of the selected time-slot, such that (i) no two links that share a device are simultaneously chosen, (ii) the throughput of each link $(i,j) \in L$ must be greater or equal than the minimum throughput γ_{ij}. A solution for this problem is represented by a schedule $S_{MD} \subseteq L \times C \times T$. An element $\langle (i,j),c,t \rangle \in S_{MD}$ means that link $(i,j) \in L$ is assigned to channel $c \in C$ and transmits at time-slot $t \in T$. The objective of this problem is to find the schedule that uses the minimum number of time-slots over all possible schedules.*

Similarly to the VRBSP, consider the set $\Delta_{MD} \subset 2^{L \times C \times T}$ of all *all feasible MD-VRBSP schedules*, i.e., all schedules $S_{MD} \in \Delta_{MD}$ that respect conditions (i) and (ii) in Definition 1.4. Besides, let $T(S_{MD})$ be the set of time-slots used in S_{MD}, then the MD-VRBSP is formally defined as in Equation (1.6).

$$S^*_{MD} = \underset{S_{MD} \in \Delta_{MD}}{\text{argmin}} |T(S_{MD})| \tag{1.6}$$

The MD-VRBSP is a generalization of the original WSP (Definition 1.1), as when $|T| = |B| = |C| = 1$ MD-VRBSP reduces to WSP. To the best of the author's knowledge, this is the first work to tackle this problem. Table 1.2 gathers all symbols used so far to define the VRBSP and the MD-VRBSP.

Table 1.2: Symbols used in this chapter to define VRBSP and MD-VRBSP.

Symbol	Description
V	Set of devices.
L	Set of links.
C	Set of channels.
B	Set of bandwidths.
T	Set of time-slots.
O^c	Set of channels that overlap with channel c.
B^c	Bandwidth value of channel c.
C^b	Set o channels whose bandwidth is b.
M	Set of indexes for the Modulation and Coding Scheme.
a_{ij}	Affectance value of link (i,j).
I_{ij}	Interference value at the receiver of link (i,j).
$\bar{r}_m^b$	Data transmission rate using MCS m and bandwidth b.
q_m^b	Minimum SINR necessary to transmit using MCS m and bandwidth b.
γ_{ij}	Minimum data-rate requested by link (i,j)
β_{ij}^c	Minimum SINR necessary for link (i,j) to achieve γ_{ij} in channel c.
P	Transmission power used in the transmission.
N	Environment noise.
d_{ij}	Euclidean distance between devices $i \in V$ and $j \in V$.
S_{VR}	A VRBSP solution.
S_{MD}	A MD-VRBSP solution.

1.3 Objectives and Contributions

The goal of this work is to study and propose new algorithms to solve problems in the computer networks field (VRBSP and MD-VRBSP). Therefore, part of the goal is to evaluate the proposed algorithms and to make conclusions on whether they are or not better algorithms than the ones existing in the literature. To that end, the following computational techniques are used:

- **Integer Programming**: this work proposes one mixed-integer programming formulation for the VRBSP, and two mixed-integer programming formulations for the MD-VRBSP;
- **Heuristics**: a Variable Neighborhood Search (VNS) algorithm is proposed for each problem.

The performance of each algorithm will be measured according to the value of the objective function computed within the established time limits. This work will execute empirical experiments with baseline algorithms (e.g., [Costa et al., 2017, 2019]) from the literature and the ones proposed in later chapters. Besides, this work will use the warm-start technique [Wolsey, 2020] with the MD-VRBSP experiments. This technique will provide insights into the quality of the solutions generated by the heuristics. Besides, it may improve the results of the MILP formulations, as it can improve optimality gaps. It is expected that, at the end of this work, the following Research Questions (RQ) are completely or (at least) partially answered:

(RQ1): Does the new mixed-integer formulation for the VRBSP find as good or as lower optimality gaps as the one proposed in [Costa et al., 2019]?

(RQ2): Which heuristic for the VRBSP achieves better results? Can these results be considered satisfactory?

(RQ3): Which mixed-integer formulation is the best for the MD-VRBSP, i.e., which one gives lower optimality gaps?

(RQ4): Which heuristic for the MD-VRBSP achieves better results? Can these results be considered satisfactory?

(RQ5): Does the warm-start technique succeed in improving the results of the MD-VRBSP MILP formulations?

1.4 Methodology

The objectives of this work will be evaluated through computational experiments. The experiments with the MILP formulation will be carried out in commercial solvers, such as the Gurobi[2] solver. The authors coded the heuristics using the same computational language. All source codes written for the experiments will be available online in public repositories. The experiments were executed using a set of instances builded using the same methodology found in related work.

1.5 Outline

The remainder of this work is organized as follows. Chapter 2 presents related work of the WSP literature. In Chapters 3 and 4, solutions strategies such as integer programming models and heuristics are proposed for the VRBSP and the MD-VRBSP, respectively. The

[2]Available in: https://www.gurobi.com

details of the computational experiments and answers for the RQs are the subject of Chapter 5. Finally, Chapter 6 depicts the conclusions of this work and future work.

Chapter 2

Related Work

This chapter is devoted to discussing some of the works in the WSP literature. Particularly, it focuses only on related work that use the SINR as the interference model. This choice was based not only on filtering the works but because the SINR model is the default model in the WSP literature since it was formalized in [Gupta and Kumar, 2000].

Two aspects are of relevant interest in the theoretical study and algorithmic design of the WSP: the interference model and the power controlling adopted. A detailed survey about these two aspects is presented in Goussevskaia et al. [2010]. The following two sections revisit some works related to the theme. The chapter proceeds with two additional sections by presenting works separated between those who consider single and multiple data-rates, respectively. In this work, the SINR model is considered. Moreover, it is assumed that all links have the same power transmission level, and that links can achieve different data-rates according to the bandwidth of the communication channel that it is transmitting.

2.1 Interference modelling in the WSP

Throughout the years, graph-based representations and the SINR model were largely used. The former relies on graphs and its properties. Generally, devices are represented as nodes, and links as edges that connect these nodes. Using this type of representation has its advantages: scientists are familiar with graph theory and wireless networks are easily represented by graphs. Indeed, there are polynomial algorithms for variants of the WSP using graph representation ([Ephremides and Truong, 1990], for instance). Besides, some more realistic variants lead to matching [Borbash and Ephremides, 2006] or coloring problems [Goussevskaia et al., 2007, Chafekar et al., 2008]. However, interference models are far away from the physical reality of wireless networks when graphs are considered. The major causes are that the interference caused by different senders may accumulate and it is not binary, i.e., does not stop at any specific border.

The SINR model, formalized in Gupta and Kumar [2000], is believed to be more realistic

[Blough et al., 2010]. For instance, it captures the accumulated interference that may affect a link. It was on [Moscibroda and Wattenhofer, 2006] that the algorithm work with this model started, increasing the interest for the theme. The original work provided an approximation scheduling algorithm that schedules links into a polynomial number of time-slots in networks with arbitrary topologies. A tighter result was presented in Moscibroda et al. [2006a]. Later, other works explored the scheduling in networks with different aspects. For instance, there are works considering sensors networks [Maheshwari et al., 2008] and networks in multi-hop fashion [Cruz and Santhanam, 2003].

Goussevskaia et al. [2007] presented a proof that the WSP under the SINR model is a NP-hard problem, leaving the question of whether it was possible to create better algorithms. In the same work, an approximation algorithm was provided. Approximation algorithms were indeed one of the main techniques used in the WSP literature.

2.2 Power Controlling in the WSP

Power controlling in wireless transmissions is relevant for several reasons. For example, reducing the power used in a transmission leads to energy saving. On the other hand, increasing the power helps to diminish the effects of high interference. These conflicting aspects motivated the discussion in [Moscibroda and Wattenhofer, 2006, Moscibroda et al., 2006a,b]. Besides, [Voelker et al., 2009] showed, based on the work of Goussevskaia et al. [2007], that scheduling with power control is also a NP-hard problem. Mainly, there are two types of setting the power control: with a *uniform* power assignment (when all links have the same power transmission level), or with a *linear* power assignment (when the power value is proportional to a path-loss exponent). Some authors (e.g, [Halldórsson, 2009]) refer to them as *oblivious* power settings, as they depend only on the length of the given link. Moscibroda et al. [2007] argue that the former setting yields poor worst-case schedules. Fortunately, the scenarios where such a result happened is rare and in practice an uniform setting is commonly used, as pointed out by [Goussevskaia et al., 2009].

Just as with the interference case, approximation algorithms were the most used technique to find feasible schedules. Normally, the approximation factors depend upon the maximum and minimum distance between the links. For instance, the reader can refer to [Goussevskaia et al., 2007, 2009, 2012, Blough et al., 2010] for the uniform case. For the linear case, [Halldórsson, 2009, Halldórsson and Mitra, 2011a,b, Kesselheim and Vöcking, 2010] are some of the relevant works. Nevertheless, there are also other approaches, such as exact methods. In [Bjorklund et al., 2003, Kompella et al., 2007], the column generation technique is used to solve the WSP with uniform and linear power setting, respectively. Andrews and Dinitz [2009] proposed a game theory algorithm to solve the linear power assignment problem. Based on this work, [Ásgeirsson and Mitra, 2011] achieved better results both for uniform and linear power assignment.

All works cited so far in this chapter belong to the class of single rate variants. This type of variant has the characteristic that only a single, usually fixed, data-rate is considered. The data-rate is given as part of the input or computed once in the algorithm. The use of a single data-rate was compatible with earlier technologies that have a simpler usage of the frequency spectrum. However, later versions of the Wi-Fi protocol, for instance, allows for each link to have its own data-rate value.

2.3 Variable Rate Variants

In Variable Rate Variants, the throughput is part of a solution, meaning that it can assume different values. Naturally, these values are within a lower and an upper bound, as in the case of Table 1.1. So far, there are few WSP works considering variable data-rates, but it seems that it will become a trend in the WSP literature. Santi et al. [2009] is the first work using variable data-rates, motivated by earlier works [Zuniga and Krishnamachari, 2004, Maheshwari et al., 2008, 2009] that contributed to a better understanding on how the SINR levels affect the data-rate. It was noted that the curve that represents this relationship has what is now called the *gray region*. Links transmitting within this region might have a successful transmission, but with a lower average data-rate. Modern APs can actually adjust the data-rate in order to correspond with the SINR levels.

Kesselheim [2012] used the SINR model to define the Variable Rate Scheduling Problem (VRSP) as follows. Select a subset $S \subseteq L$ of links to transmit concurrently and a data-rate $r_{ij} \in R$ for each $(i, j) \in S$ such that $SINR_{ij} \geq \beta(r_{ij})$. The objective function is to find the optimal throughput of the network (i.e., the maximum possible sum of each data-rates). Note, however, that when $|R| = 1$, VRSP reduces to the SSSP. Therefore, VRSP is a $\mathcal{NP}$-Complete problem. In that way, Kesselheim [2012] proposed an algorithm with an approximation factor of $\mathcal{O}(logn)$.

The VRSP is also the subject of the work of Goussevskaia et al. [2016]. They modeled this problem as a conflict graph and proposed a polynomial-time approximation scheme (PTAS). This algorithm has an approximation factor of $(1 + \frac{1}{K-1})^2$, where K is a constant. Hence, the algorithm achieves a constant approximation factor as K gets larger. Besides, to assess the performance of the proposed algorithm, experimental tests were run with adaptations of the algorithms presented in [Goussevskaia et al., 2007, 2009, 2012]. In these experiments, [Goussevskaia et al., 2016] achieved better results than all previous adaptations. However, this algorithm has a complexity of $n^{\mathcal{O}(K^2)}$, being impractical for larger networks.

More recently, Costa et al. [2017] proposed the VRBSP variant that was defined in Chapter 1 and introduced a BRKGA-based heuristic to find VRBSP schedules. Besides, Costa et al. [2019] proposed two mixed-integer formulations to find feasible VRBSP schedules. They performed computational experiments for both VRSP and VRBSP. In these experiments, they achieved results that outperformed other algorithms from the literature. Nevertheless, their MILP formulations require prohibitive computational running time in instances with more than 1024 links.

In this work, mixed-integer formulations and heuristics are proposed for the VRBSP and the MD-VRBSP. For the VRBSP, the hypothesis is that the new formulation might improve the LP bounds, by eliminating, for instance, auxiliary variables and coupling constraints. Regarding the MD-VRBSP, these are the first formulations for this problem and will be compared against each other in order to identify which is better. Moreover, a VNS-based heuristic is proposed for this problem.

Chapter 3

Algorithms for the VRBSP

This chapter focuses on presenting algorithms to find solutions for the VRBSP. To that end, it starts by reviewing a mixed-integer formulation proposed by Costa et al. [2019]. The chapter then proposes a different but equivalent mixed-integer formulation that also uses the big-M approach. At last, a Constructive Algorithm (CH) and a VNS-based heuristic are also proposed.

3.1 Mixed-integer Formulations for the VRBSP

Consider the tuple $\langle V, L, B, C, O^c, C^b, M, q_s^b, \bar{r}_s^b, P, d_{ij}, \alpha, N \rangle$ that characterizes an instance of the VRBSP (see Table 1.2 for a summary of each symbol). Costa et al. [2019] proposes the following: define the decision variables $x_{ij}^c \in \{0,1\}$, where $x_{ij}^c = 1$ if and only if link $(i,j) \in L$ is scheduled and assigned to channel $c \in C$, and $x_{ij}^c = 0$ otherwise. Moreover, consider the set of auxiliary variables $y_{ij}^{bm} \in \{0,1\}$, where $y_{ij}^{bm} = 1$ means that link $(i,j) \in L$ is transmitting using channel $c \in C$ and MCS index $m \in M$, and $y_{ij}^{bm} = 0$ otherwise. Lastly, the auxliary variables $I_{ij} \in \mathbb{R}$ represent the interference that reaches the receiver j of link all links $(i,j) \in L$. These variables are used in formulation VR1 (equations (3.12)-(3.19)) to describe the VRBSP.

The objective function (3.12) maximizes the throughput of the transmission. Constraint (3.13) guarantees that every link is assigned to at most one channel, and that two links that share the same device are not scheduled together. Constraints (3.14) and (3.15) compute the right values for the variables z_{uv}^c and I_{ij}^c, while constraints (3.16) and (3.17) compute the interference I_{ij} of link (i,j). Besides, constraint (3.18) determines the minimum SINR value required in order to link (i,j) transmit in channel $c \in C$ using $m \in M$. Constraint (3.19) guarantees the domain of the variables. In this work, the value of the big-M M_{ij} constant equals to the sum of the affectance of all links into another one, as stated in Equation (3.1).

$$M_{ij} = \sum_{(u,v) \in L} a_{uj}, \forall (i,j) \in L \tag{3.1}$$

(VR1)

$$\max \sum_{(i,j)\in L}\sum_{b\in B}\sum_{m\in M} \bar{r}_m^b \cdot y_{ij}^{bm} \tag{3.2}$$

$$\text{s.t.} \sum_{(i,j)\in L}\sum_{c\in C} x_{ij}^c + \sum_{(j,i)\in L}\sum_{c\in C} x_{ji}^c \leq 1, \qquad \forall i \in V, \tag{3.3}$$

$$\sum_{m\in M} y_{ij}^{bm} \leq \sum_{c\in C^b} x_{ij}^c, \qquad \forall (i,j) \in L, b \in B, \tag{3.4}$$

$$\sum_{b\in B}\sum_{m\in M}\left(\frac{\frac{P}{(d_{ij})^\alpha}}{q_m^b} - N\right)\cdot y_{ij}^{bm} \geq I_{ij}, \qquad \forall (i,j) \in L, \tag{3.5}$$

$$z_{uv}^c = \sum_{\bar{c}\in O^c} x_{uv}^{\bar{c}}, \qquad \forall (u,v) \in L, \forall c \in C, \tag{3.6}$$

$$I_{ij}^c = \sum_{(u,v)\in L\setminus\{(i,j)\}} \frac{P}{(d_{uj})^\alpha}\cdot z_{uv}^c, \qquad \forall (u,v) \in L, \forall c \in C, \tag{3.7}$$

$$I_{ij} \geq I_{ij}^c - M_{ij}\cdot\left(1 - x_{ij}^c\right), \qquad \forall (i,j) \in L, \forall c \in C, \tag{3.8}$$

$$I_{ij} \leq I_{ij}^c + M_{ij}\cdot\left(1 - x_{ij}^c\right), \qquad \forall (i,j) \in L, \forall c \in C, \tag{3.9}$$

$$y_{ij}^{bm} \in \{0,1\}, \qquad \forall (i,j) \in L, \forall b \in B, \forall m \in M, \tag{3.10}$$

$$x_{ij}^c \in \{0,1\}, \qquad \forall (i,j) \in L, \forall c \in C. \tag{3.11}$$

The same problem can be modeled in an alternative formulation. To that end, consider the same VRBSP instance characterized by the same VRBSP tuple. However, let the decision variables be represented by $x_{ij}^{cm} \in \{0,1\}$, where $x_{ij}^{cm} = 1$ if and only if link $(i,j) \in L$ is scheduled and assigned to channel $c \in C$ and uses MCS m, and $x_{ij}^{cm} = 0$ otherwise. The variables $I_{ij} \in \mathbb{R}$ have the same definition as before. These variables are used in formulation VR2 (equations (3.12)-(3.19)) to describe the VRBSP.

The objective function (3.12) maximizes the throughput of the transmission. Constraint (3.13) guarantees that every link is assigned to at most one channel, and that two links that share the same device are not scheduled together. Constraints (3.14) and (3.15) compute the right values for the variables z_{uv}^c and I_{ij}^c, while constraints (3.16) and (3.17) compute the interference I_{ij} of link (i,j). Besides, constraint (3.18) determines the minimum SINR value required in order to link (i,j) transmit in channel $c \in C$ using $m \in M$. Constraint (3.19) guarantees the domain of the variables. The big-M constants are also defined using equation (3.1).

$$\text{(VR2)} \quad \begin{aligned}
\max \quad & \sum_{(i,j)\in L}\sum_{b\in B}\sum_{m\in M} \bar{r}_m^b \cdot x_{ij}^{cm} && && (3.12)\\
\text{s.t.} \quad & \sum_{(i,j)\in L}\sum_{c\in C}\sum_{m\in M} x_{ij}^{cm} + \sum_{(j,i)\in L}\sum_{c\in C}\sum_{m\in M} x_{ji}^{cm} \leq 1, && \forall i \in V, && (3.13)\\
& z_{uv}^c = \sum_{\bar{c}\in O^c} x_{uv}^{\bar{c}m}, && \forall (u,v) \in L, \forall c \in C, && (3.14)\\
& I_{ij}^c = \sum_{(u,v)\in L\setminus\{(i,j)\}} a_{uj} \cdot z_{uv}^c, && \forall (i,j) \in L, \forall c \in C, && (3.15)\\
& I_{ij} \geq I_{ij}^c - M_{ij} \cdot (1 - x_{ij}^{cm}), && \forall (i,j) \in L, \forall c \in C, \forall m \in M, && (3.16)\\
& I_{ij} \leq I_{ij}^c + M_{ij} \cdot (1 - x_{ij}^{cm}), && \forall (i,j) \in L, \forall c \in C, \forall m \in M, && (3.17)\\
& \sum_{c\in C}\sum_{m\in M} \left(\frac{a_{ij}}{q_m^c} - N\right) \cdot x_{ij}^{cm} \geq I_{ij}, && \forall (i,j) \in L, && (3.18)\\
& x_{ij}^{cm} \in \{0,1\}, && \forall (i,j) \in L, \forall c \in C, \forall m \in M. && (3.19)
\end{aligned}$$

While VR1 and VR2 are equivalent formulations, VR2 does not have coupling constraints as in equation (3.4), thus strengthening formulation VR1. Therefore, it is believed that VR2 can achieve better optimality gaps. As a consequence of this observation, VR2 saves a cubic quantity of memory as it removes the y_{ij}^{bm} variables. Note that, depending on the size of C and M, VR2 would lead to a model with more variables than VR1, as it would result in a model with up to $\mathcal{O}(|V|^2 \cdot |C| \cdot |M|)$ variables.

3.2 Constructive Heuristic

The formalization of the CH requires the introduction of two procedures that are capable of manipulating Wi-Fi channels. The split and merge procedures have Wi-Fi channels as input and output and conserve the total bandwidth and the scheduled links in the target channels. Nevertheless, they operate in opposite directions. For the sake of clarity, let two hypothetical channels, c_1 and c_2, having a bandwidth of $b_{c_1} = b_{c_2} = 40$ MHz. Besides, let four links $\{l_1, l_2, l_3, l_4\}$ where $L(c_1) = \{l_1, l_2\}$ and $L(c_2) = \{l_3, l_4\}$. Given the resultant interference at each link, the throughputs of channels c_1 and c_2 are respectively 360 Mbps and 324 Mpbs. This configuration is illustrated in Figure 3.1.

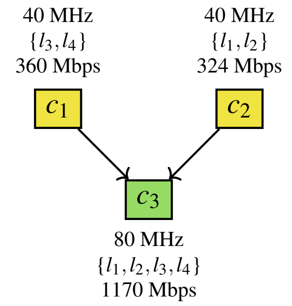
The operation of merging of c_1 and c_2 results in a new channel c_3, which will have bandwidth equal as $b_3 = b_1 + b_2$ and scheduled links $L(c_3) = \{L(c_1) \cup L(c_2)\}$. In the example, c_3, receives all links originally scheduled in c_1 and c_2, but with a bandwidth 80 MHz and a new throughput, as depicted by Figure 3.2.

Suppose now, as depicted in Figure 3.3, that channel c_4 has bandwidth $b_{c_4} = 40$ MHz

Figure 3.1: Two hypothetical Wi-Fi 6 channels.

Source: Own authorship

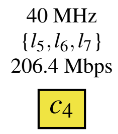
Figure 3.2: An example of the merge operation.



Source: Own authorship

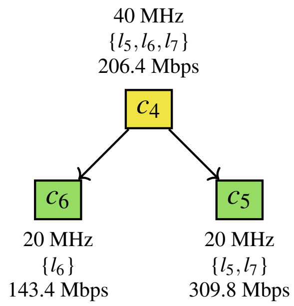
and $L(c_4) = \{l_5, l_6, l_7\}$, and a throughput that is equal to 206.4 Mbps. The splitting operation in channel c_4 results in two new channels, c_5 and c_6, as displayed in Figure 3.4. Each new channel has half of the original bandwidth, i.e., $b_{c_5} = b_{c_6} = b_{c_4}/2$. The rearrangement of the links $L(c_4)$ depends on the user-implementation. In this work, each link is randomly scheduled into the new channels.

Figure 3.3: A hypothetical Wi-Fi 6 channel.



Source: Own authorship

Figure 3.4: An example of the splitting operation.



Source: Own authorship

Consider now $\Psi(S_{VR}) = \text{argmin}_{Z \in G: C(S_{VR}) \subseteq Z} |Z|$. In practice, $\Psi(S_{VR})$ can be translated to the set that contains two different subset of channels: (i) the channels used in S_{VR} and (ii) the other channels that have the largest bandwidth and that “fill” the available wireless. Moreover, consider the definition of G, H, and $C(S_{VR})$ as follows:

- $G = \{Y \in H : \sum_{c \in Y} B^c = 2*160+2*80+20\}$: Subset of another subset of channels of H whose bandwidth summation equals to the available bandwidth in the wireless spectrum;
- $H = \{X \in 2^C : |O^c \cap X| = 0 \forall c \in X\}$: Subset of another subset of channels that does not collide among themselves;
- $C(S_{VR}) = \{c \in C : \exists \langle (i,j), c \rangle \in S\}$ contains all channels used in a solution S_{VR}.

For instance, let a hypothetical solution S_{VR} for which $C(S_{VR}) = \{3,4,26,39\}$. In this case, $\Psi(S_{VR}) = \{3,4,25,26,39,43,45\}$. Besides, if $C(S_{VR}) = \{9,10,11,12\}$, then $\Psi(S_{VR}) = \{9,10,11,12,25,41,42,43,44\}$. Note that $\Psi(\emptyset) = \{25,42,43,44,45\}$ is the subset of channels that “fill” the Wi-Fi spectrum and cannot be merged.

The CH for the VRBSP tries to schedule as many links as possible. In Algorithm 1, this is represented in the **foreach** loop of line 2. A link $(i,j) \in L$ is only inserted if it can cause an improvement in the throughput of the incumbent solution S_{VR}. Otherwise, this link is left unscheduled. The insertion procedure tests the insertion of link $(i,j) \in L$ for each channel $c \in \Psi(S_{VR})$ as follows. Three new solutions are created S^1, S^2, S^3. S^1 is the result of the insertion of (i,j) into c (line 5). If $B^c > 20$, then c is splitted into two channels c_1 and c_2. The algorithm then creates S^2 as the result of the insertion of (i,j) into c_1 and S^3 as the result of the insertion into c_2 (line 6). The last step (line 7) compares which solution has the highest throughput, and updates the current optimum solution if an improvement is found.

Algorithm 1: CH-VRBSP

---

input : I_{VR}
output: $S_{VR} \in \Delta_{VR}$

1 $S_{VR} \leftarrow \emptyset$
2 **foreach** $(i,j) \in L$ **do**
3     $S' \leftarrow \emptyset$
4   **foreach** $c \in \Psi(S_{VR})$ **do**
5         $S^1 \leftarrow S_{VR} \cup \{\langle (i,j), c \rangle\}$
6         $S^2, S^3 \leftarrow \text{SplitInsert}(S_{VR}, (i,j), c)$
7         $S' \leftarrow \text{Best}(S', S^1, S^2, S^3)$
8   **end**
9     $S_{VR} \leftarrow \text{Best}(S, S')$
10 **end**
11 **return** S_{VR}

3.3 VNS-based heuristic

A VNS-based heuristic for the VRBSP aims to improve a feasible solution by exploring its neighbors. A neighbor of a solution S_{VR} is another solution S' such that $S_{VR} \setminus S' \neq 0$. A k-neighbor of S_{VR} is a solution S' such that $S \setminus S' = k$. Consider $S \in \Delta_{VR}$, then $\mu(S_{VR}, \tau) \in \Delta_{VR}$ is the set of solutions in the neighborhood of S_{VR} that is obtained using a procedure τ. When applied to a scheduling $\langle (i,j), c \rangle \in S_{VR}$, this procedure can change channel c by some other channel $c' \in \Psi(S_{VR})$ or remove this link from the solution. Formally, $\mu(S_{VR}, \tau) = \{S' \in \Delta_{VR} : |S_{VR} \setminus S'| = k\}$, where k is the k-th neighborhood.

The VNS descent variant, introduced by [Mladenović and Hansen, 1997], was chosen for this work and is detailed in Algorithm 2. The procedure starts with an incumbent solution Γ that is equal to the solution returned by the CH. Then, until the stopping criterion is reached, this VNS variant applies perturbation and local-search procedures in Γ in an attempt to find an improved solution Γ''. In this case, Γ is updated with Γ'', and the search is resumed from its 1-neighborhood. Otherwise, the search is continued in the next neighborhood $(k+1)$, until reaching the $(k + K^{MAX} - 1)$-neighborhood. Note that the search always returns to the 1-neighborhood after visiting all possible neighborhoods.

A careful analysis would reveal that a trivial approach that searches for the best solution in a neighborhood might need to evaluate an exponential number of solutions. Nevertheless, there exists an approach that requires a polynomial time to compute the configuration of the channels that gives the optimal objective value for a neighborhood search. This approach is expressed in Algorithm 2 by two adaptations, namely the *Reshape* and the *DP* procedures. The former is a procedure that reshapes a solution $S_{VR} \in \Delta_{VR}$ into an equivalent solution that has only 20 MHz channels. The latter is another procedure that computes the local best partitioning of the channels of a particular solution and returns its throughput. The *DP* procedure is presented in details first, followed by the *Reshape* procedure.

Consider the example illustrated in Figure 3.5, already used earlier in Chapter 1. Note that channel c_3 has a higher throughput (1170 Mbps) than the sum of the throughput of channels c_1 and c_2 (684 Mbps). A decision algorithm that asks "does the channel resulting from the merging of two other channels has a higher throughput than the sum of its children?"[1] would choose channel c_3 over c_1 and c_2 as it contributes more to the objective function. By executing this decision algorithm in all channels of any solution, the result is the optimal configuration of the channels. Therefore, the final solution is always no worse than the original solution. This work named this algorithm as *DP* because of the similarities with a top-down dynamic programming algorithm. Algorithm 3 presents the details. It recursively decides whether it is best to merge two channels or to leave them separated based on the resultant throughput.

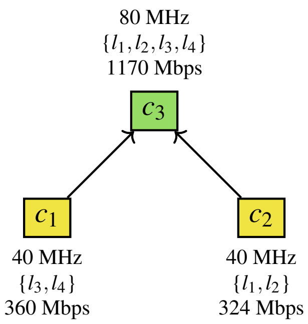
[1] Another way to formulate the question is: "it is better to merge two channels or to leave it separated?"

Algorithm 2: VNS-VRBSP

input : I_{VR}, S^0_{VR}
output: S^*_{VR}

$\Gamma \leftarrow \text{Reshape}(S^0_{VR})$
while *Stopping criteria not met* **do**
  $k \leftarrow 1$
  while $k \leq K^{max}$ **do**
    $\Gamma' \leftarrow \text{Perturbation}(\Gamma)$
    $\Gamma'' \leftarrow \text{LocalSearch}(\Gamma')$
    if $f(\text{DP}(\Gamma'')) > f(\text{DP}(\Gamma))$ **then**
      $\Gamma \leftarrow \Gamma''$
      $k \leftarrow 1$
    else
      $k \leftarrow k + 1$
    end
  end
end
return $S^*_{VR} = \text{DP}(\Gamma)$

Figure 3.5: An example of when the merge of two channels can improve the throughput.



Source: Own authorship

Given that the *DP* procedure decides the optimal partitioning of the channels, it is not necessary to search solutions over the entire set of feasible solutions Δ_{VR}. Instead, it is only necessary to search within a subset $\Delta^{20}_{VR} \subseteq \Delta_{VR}$ of solutions where all channels have a bandwidth of 20 MHz. Given that the algorithm receives a solution $S_{VR} \in \Delta_{VR}$, a procedure, named *Reshape*, was designed to perform a conversion of a solution S_{VR} into a S^{20}_{VR}.

Algorithm 4 depicts how the *Reshape* procedure works. It starts by initializing an empty solution S'_{VR}. The next step executes the *BuildDictionary* auxiliary procedure, that receives S_{VR} as input and returns a dictionary D. A element of D is in the form of a pair $(c, L(S_{VR}, c))$, where c is a channel used in S_{VR} and $L(S_{VR}, c) \subseteq S_{VR}$ is a list of tuples $\langle (i, j), c \rangle \in S_{VR}$[2], i.e., a subset of links that are transmitting using $c \in C$. The loop of lines 3-11 is responsible for constructing S'. It does so by iterating over each element of D. Whenever the loop reaches an element with

[2]Note that $L(S_{VR}, c)$ is also a valid solution for the VRBSP.

Algorithm 3: DP

input : An instance $I = \langle S_{VR}, c \rangle$
output: Best local partitioning S'_{VR} of channel c

$S'_{VR} \leftarrow L(S,c)$
if $B^c = 20$ **then**
  return S'
end
$s_1, s_2 \leftarrow \text{Split}(c)$
$s'_1 \leftarrow \text{DP}(s_1)$
$s'_2 \leftarrow \text{DP}(s_2)$
if $f(s'_1) + f(s'_2) \geq f(S'_{VR})$ **then**
  $S'_{VR} \leftarrow s'_1 \cup s'_2$
end
return S'_{VR}

$B^c = 20$, it appends $L(S_{VR}, c)$ to S'_{VR}. Otherwise, $L(S_{VR}, c)$ is splited onto s_1 and s_2 using the auxiliary procedure *Split*. Finally, the results of the recursive calls *Reshape(s_1)* and *Reshape(s_2)* are appended to S', and the algorithm proceeds to the next element, if any, or returns S'.

Algorithm 4: Reshape

input : S_{VR}
output: $S'_{VR} \in \Delta^{20}$

$S'_{VR} \leftarrow \emptyset$
$D \leftarrow \text{BuildDictionary}(S_{VR})$
foreach $(c, L(S_{VR}, c)) \in D$ **do**
  if $B^c = 20$ **then**
    $S'_{VR} \leftarrow S' \cup L(S_{VR}, c)$
  else
    $s_1, s_2 \leftarrow \text{Split}(L(S_{VR}, c))$
    $S'_{VR} \leftarrow S'_{VR} \cup \text{Reshape}(s_1)$
    $S'_{VR} \leftarrow S'_{VR} \cup \text{Reshape}(s_2)$
  end
end
return S'_{VR}

3.3.1 Perturbation

The perturbation procedure, detailed in Algorithm 5, modifies a solution S_{VR} by changing a schedule $\langle (i,j), c \rangle$ to $\langle (i,j), \hat{c} \rangle, c \neq \hat{c}$, or by removing a scheduling, i.e., $S_{VR} \setminus \langle (i,j), c \rangle$.

Such a changing is executed in the reshaped solution $\Gamma \in \Delta_{20}$. Next, k links will be removed from Γ within a probability of σ (line 2). In this case, k links are selected at random and then removed from Γ (lines 3 - 6). Nevertheless, the perturbation procedure always inserts k links that, at this point (lines 8 - 12), are not part of Γ. This selection is also performed by selecting random links. Therefore, note that the resultant interference of some scheduling $\langle (i,j),c \rangle$ may be high enough so that there might exist a link that is not transmitting. The *RemoveInfeasibleLinks* then remove such links from Γ. The procedure then finishes returning the associated $S \in \Delta_{VR}$ solution with Γ.

Algorithm 5: VR-PER

input : I_{VR}, S^0, k
output: $S_{VR} \in \Delta_{VR}$

1 $\Gamma \leftarrow$ Reshape(S^0)
2 **if** *Random(*$[0,1]$*)* $\leq \sigma$ **then**
3   **for** *k iterations* **do**
4         $\langle (i,j),c \rangle \leftarrow$ Random(Γ)
5         $\Gamma \leftarrow \Gamma \setminus \{\langle (i,j),c \rangle\}$
6   **end**
7 **end**
8 **for** *k iterations* **do**
9     $(i,j) \leftarrow$ Random($L \setminus L(\Gamma)$)
10     $c \leftarrow$ Random($\Psi(\Gamma)$)
11     $\Gamma \leftarrow \Gamma \cup \{\langle (i,j),c \rangle\}$
12 **end**
13 RemoveInfeasibleLinks(Γ)
14 **return** $S_{VR} =$ DP(Γ)

3.3.2 Local Search

The local search designed for this problem uses a first improvement strategy to determine the best neighbor of a given solution. Its goal is to decide between scheduling a link or leaving it out of the solution. In the former case, the algorithm also determines the best channel to schedule, i.e., the tuple $\langle (i,j),c \rangle$ that will increase the objective function value the most. The latter case can be more profitable if this link causes a high interference, thus decreasing the throughput of the schedule.

Algorithm 6 illustrates how this local search works. After reshaping a solution S^0 into a $\Gamma \in \Delta_{VR}^{20}$, the algorithm starts the search for the best neighbor and only stops when no improved solution is found (lines 2 - 12). The *Best()* procedure is a procedure that receives a pair of solutions, e.g., Γ' and Γ^*, and returns the best of them, based on the value computed by DP(Γ') and DP(Γ^*).

Algorithm 6: VR-LS

input : I_{VR}, S^0
output: $S_{VR} \in \Delta_{VR}$

$\Gamma \leftarrow \text{Reshape}(S^0_{VR})$
do
  $\hat{\Gamma} \leftarrow \Gamma$
  foreach $(i,j) \in L$ **do**
    $\Gamma^{\emptyset}, \Gamma^* \leftarrow \Gamma \setminus \{\langle (i,j), c\rangle \in L \times C : \langle (i,j), c\rangle \in \Gamma\}$
    foreach $c' \in C^{20}$ **do**
      $\Gamma' \leftarrow \Gamma^{\emptyset} \cup \{\langle (i,j), c'\rangle\}$
      $\Gamma^* \leftarrow \text{Best}(\Gamma', \Gamma^*)$
    end
    $\Gamma \leftarrow \text{Best}(\Gamma, \Gamma^*)$
  end
while $\hat{\Gamma} \neq \Gamma$;
return $S = \text{DP}(\Gamma)$

3.4 Summary of the Chapter

This chapter presented solution strategies for the VRBSP. It reviews one mixed-integer program proposed in Costa et al. [2019]. Moreover, this chapter introduces one new mixed-integer program and one new heuristic to find feasible solutions for this problem. In the following, the main symbols (e.g., variables and constants) used in this chapter are depicted in Table 3.1.

Table 3.1: Symbols used to propose exact formulations and heuristics for the VRBSP.

Symbol	Type	Meaning
I_{VR}	Set	Valid instance for the VRBSP
$C(S)$	Set	Channels used in solution S_{VR}
$\Psi(S_{VR})$	Set	Contains all channels of S_{VR} and the channels with the largest bandwidth
Random($[0,1]$)	Func.	Returns a random number between $[0,1]$
Reshape(S_{VR})	Func.	Reshapes a solution $S \in \Delta_{VR}$ to a solution $S^{20} \in \Delta^{20}_{VR}$
Child(c)	Func.	Returns the "childs" of channel c
Best(Γ, Γ')	Func.	Returns the best solution according the $DP(\Gamma)$ and $DP(\Gamma')$

Chapter 4

Algorithms for the MD-VRBSP

This chapter tackles the Minimum-Delay Variable Rate Variable Bandwidth Problem by providing, in order of presentation, one mixed-integer nonlinear program, two mixed-integer linear programs and a VNS-based heuristic.

4.1 Mixed-Integer Formulations for the MD-VRBSP

4.1.1 A Mixed-Integer Non-linear Formulation

Given an instance of the MD-VRBSP that is characterized by the tuple $I_{MD} = \langle T, V, L, B, C, O^c, C^b, M, q_s^b, \bar{r}_s^b, P, d_{ij}, \alpha, N, \gamma_{ij} \rangle$ (see Table 1.2 for a summary of each symbol), consider the variable $z^t \in \{0,1\}$ such that $z^t = 1$ if and only if time-slot $t \in T$ is being used in the transmission and $z^t = 0$ otherwise. Besides, let $x_{ij}^{ct} \in \{0,1\}$ such that $x_{ij}^{ct} = 1$ if link $(i,j) \in L$ is set to transmit using channel $c \in C$, MCS $m \in M$, and at time-slot $t \in T$, and $x_{ij}^{ct} = 0$ otherwise. Consider also the use of auxiliary variables $I_{ij} \in \mathbb{R}$, for all $(i,j) \in L$, that give the interference at j.

The objective function (4.2) minimizes the number of time-slots used in the transmission. Constraint (4.3) indicates that the time-slots are used in an ordered manner. Constraint (4.4) enforces every link $(i,j) \in L$ to be scheduled at some time slot. Moreover, constraint (4.5) guarantees that every link is assigned to at most one channel and one time-slot, and that two links that share the same device are not scheduled together. The interference of each link (i,j) is computed in constraint (4.6). Besides, constraint (4.7) ensure that the minimum throughput γ_{ij} for each link will be satisfied through the usage of the β_{ij}^c variables. These variables represent the minimum SINR value necessary that a link (i,j) must have in order to transmit using channel c so that it can achieve a throughput at least as great as γ_{ij}. This value can be computed using Equation (4.1).

$$\beta_{ij}^{c} = \min_{\forall (m,B^c)\in R\,:\, \bar{r}_m^b \geq \gamma_{ij}} q_m^{B^c} \tag{4.1}$$

Moreover, Constraint (4.8) forces the program to schedule links $(i,j) \in L$ only in channels whose bandwidth allows data-rates that can go up to γ_{ij}. Finally, constraints (4.9) and (4.10) guarantees the domain of the z^t and x_{ij}^{ct} variables.

(NMD)

$$
\begin{aligned}
\min \quad & \sum_{t\in T} z^t && && (4.2)\\
\text{s.t.} \quad & z_{t+1} \leq z_t, && \forall\, t \in 1,2,\ldots,|T|-1, && (4.3)\\
& \sum_{c\in C}\sum_{t\in T} x_{ij}^{ct} = 1, && \forall\, (i,j) \in L, && (4.4)\\
& \sum_{(i,j)\in L}\sum_{c\in C} x_{ij}^{ct} + \sum_{(j,i)\in L}\sum_{c\in C} x_{ji}^{ct} \leq z^t, && \forall\, i \in V, \forall\, t \in T, && (4.5)\\
& I_{ij} = \sum_{(u,v)\in L\setminus\{(i,j)\}} a_{uj} \left(\sum_{c\in C}\sum_{\bar{c}\in O^c}\sum_{t\in T} x_{ij}^{ct} x_{uv}^{\bar{c}t} \right), && \forall\, (i,j) \in L, && (4.6)\\
& SINR_{ij} \geq \sum_{c\in C}\sum_{t\in T} \beta_{ij}^{c} x_{ij}^{ct}, && \forall\, (i,j) \in L, && (4.7)\\
& x_{ij}^{ct} = 0, && \forall \langle (i,j),(c,t)\rangle \in \phi && (4.8)\\
& z^t \in \{0,1\}, && \forall\, t \in T, && (4.9)\\
& x_{ij}^{ct} \in \{0,1\}, && \forall\, (i,j) \in L, \forall\, c \in C. && (4.10)
\end{aligned}
$$

This work provides a valid upper bound on the number of time-slots needed in the optimal solution. A trivial approach is to use a time-slot for each link, resulting in $|T| = |L|$. But the size of the linear models would turn prohibitive as it depend on the values of $|T|$ and $|L|$. Besides, this approach is clearly over estimated. Therefore, all computational experiments that do not use the warm-start technique considers that $|T| = (\sum_{(i,j)\in L} \beta_{ij}^{45})/500$, where 500 is all the bandwidth available in the Wi-Fi 6 spectrum.

4.1.2 Using a Big-m Approach

In order to define a linearization approach to formulation (4.2)-(4.10), consider the addition of the following variables. The variable $q_{uv}^{ct} \in \mathbb{R}$, defined for all $(u,v) \in L$, $c \in C$, $t \in T$, assume the value $q_{uv}^{ct} = 1$ if link $(u,v) \in L$ is assigned to an overlapping channel $\bar{c} \in O^c$, in time-slot $t \in T$, or $q_{uv}^{ct} = 0$ otherwise. Besides, let $I_{ij}^{ct} \in \mathbb{R}$ be the variable that gives which would be the value of I_{ij} if link $(u,v) \in L$ were assigned to channel $c \in C$ and time-slot $t \in T$. The correct value for these variables are ensured by constraints 4.11 and 4.12.

$$q_{uv}^{ct} = \sum_{\bar{c} \in O^c} x_{uv}^{\bar{c}t}, \qquad \forall (u,v) \in L, \forall c \in C, \forall t \in T, \tag{4.11}$$

$$I_{ij}^{ct} = \sum_{(u,v) \in L \setminus \{(i,j)\}} a_{uj} \cdot q_{uv}^{ct}, \qquad \forall (i,j) \in L, \forall c \in C, \forall t \in T. \tag{4.12}$$

Constraints (4.13) and (4.14) are used to tighten the value of the variable I_{ij}, where $M_{ij} = \sum_{(u,v) \in L \setminus \{(i,j)\}} a_{uj}$ is an upper limit to the value of I_{ij}^{ct}. Altogether, the first mixed-integer formulation for the MD-VRBSP, named of MD1, is given by equations (4.2)-(4.5), (4.7)-(4.10), and (4.11)-(4.14).

$$I_{ij} \geq I_{ij}^{ct} - M_{ij} \cdot (1 - x_{ij}^{ct}), \qquad \forall (i,j) \in L, \forall c \in C, \forall t \in T, \tag{4.13}$$

$$I_{ij} \leq I_{ij}^{ct} + M_{ij} \cdot (1 - x_{ij}^{ct}), \qquad \forall (i,j) \in L, \forall c \in C, \forall t \in T. \tag{4.14}$$

4.1.3 Using Glover and Woolsey [1974]'s Approach

The second linearization approach also uses of the variables z_{uv}^{ct}. In addition, let the variables $w_{ij}^{uv} \in \mathbb{R}$ assume the value $w_{ij}^{uv} = 1$ if and only if links (i,j) and (u,v) are assigned to the same channel in the same time-slot, and $w_{ij}^{uv} = 0$ otherwise. The correct values of variables w_{ij}^{uv} are ensured by constraints (4.15), and (4.16).

$$w_{ij}^{uv} \geq x_{ij}^{ct} + z_{uv}^{ct} - 1, \qquad \forall (i,j) \in L, \forall (u,v) \in L \setminus \{(i,j)\}, \forall c \in C, \forall t \in T, \tag{4.15}$$

$$0 \leq w_{ij}^{uv} \leq 1, \qquad \forall (i,j) \in L, \forall (u,v) \in L \setminus \{(i,j)\}. \tag{4.16}$$

It is possible to compute the right value of I_{ij} by using the variables w_{ij}^{uv}, as depicted in constraint (4.17). Therefore, the second mixed-integer formulation for the MD-VRBSP, named MD2, is given by equations (4.2)-(4.5), (4.7)-(4.10), (4.15)-(4.16), and (4.17).

$$I_{ij} = \sum_{(u,v) \in L \setminus \{(i,j)\}} a_{uj} \cdot w_{ij}^{uv}, \forall (i,j) \in L. \tag{4.17}$$

4.2 Constructive Heuristic

The goal of the proposed CH is to schedule as many links as possible using the available time-slots. The steps are depicted in Algorithm 7. It begins by initializing an empty solution and set the counter of used time-slots as 1. The rest of the algorithm aims to insert each link $(i,j) \in L$ in the following manner. First, if there is no pair $(c,t) \in \Psi(S_{MD})$ such that $\gamma\text{-}feasible(S \cup \{\langle (i,j),(\bar{c},\bar{t})\rangle\})$ returns *true*, then the counter τ is incremented one unit and then the current solution is extended with a tuple $\{\langle (i,j),(44,\tau)\rangle\}$. These steps, described in lines 4-7, can be translated as the creation of a new time-slot and the insertion of the link into a channel that has a bandwidth of 160 MHz. When the function returns *false*, then link (i,j) according to the following steps. Each $(c,t) \in \Psi(S_{MD})$ generates three solutions, S^0, S^1, S^2. The first is the result of the raw insertion of (i,j) into (c,t), while the others are generated by the *SplitInsert* procedure. The CH then selects the first one that $\gamma\text{-}feasible(S')$ returns *true*, if any, updates S_{MD} and continues to the next (i,j) until there is none.

Algorithm 7: MD-CH

input : I_{MD}
output: $S_{MD} \in \Delta_{MD}$

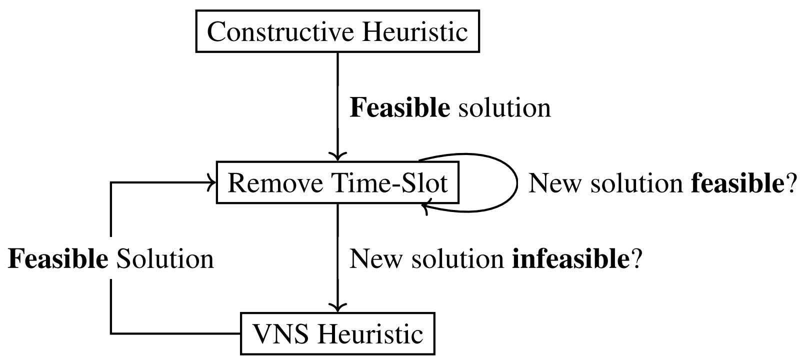
1 $S \leftarrow \emptyset, \tau \leftarrow 1$
2 **foreach** $(i,j) \in L$ **do**
3 $\quad \omega \leftarrow (\bar{c},\bar{t}) \in \Psi(S) : \gamma\text{-}feasible(S \cup \{\langle (i,j),(\bar{c},\bar{t})\rangle\}$
4 $\quad$ **if** $\omega = \emptyset$ **then**
5 $\quad\quad \tau \leftarrow \tau + 1$
6 $\quad\quad S \leftarrow S \cup \{\langle (i,j),(44,\tau)\rangle\}$
7 $\quad$ **else**
8 $\quad\quad (c,t) \leftarrow \text{First}(\omega)$
9 $\quad\quad S^0 \leftarrow S \cup \{\langle (i,j),(c,t)\rangle\}$
10 $\quad\quad S^1, S^2 \leftarrow \text{SplitInsert}(S,(i,j),(c,t))$
11 $\quad\quad$ **foreach** $S' \in \{S^0, S^1, S^2\}$ **do**
12 $\quad\quad\quad$ **if** $\gamma\text{-}feasible(S')$ **then**
13 $\quad\quad\quad\quad S \leftarrow S'$
14 $\quad\quad\quad\quad$ **continue** to line 17
15 $\quad\quad\quad$ **end**
16 $\quad\quad$ **end**
17 $\quad$ **end**
18 **end**
19 **return** S

4.3 VNS-based Heuristic

The VNS heuristic for the MD-VRBSP is based on the one proposed in Section 3.3 except for a few adaptations. The first adaptation enables the algorithm to consider the existence of multiple time-slots. This change is reflected in the code by changing all tuples $\langle (i,j), c\rangle$ by $\langle (i,j), (c,t)\rangle$. The second adaptation was the adoption of a new objective function. In this case, consider *violation* as the difference between the minimum throughput γ_{ij} of a link and the actual throughput of r_{ij}, or 0.0 if $r_{ij} \geq \gamma_{ij}$. Note that a solution is feasible if and only if its violation is equal to 0.0. Hence, the objective of this heuristic is to minimize the highest violation of a solution, i.e., $f(S_{MD}) = \min\{\max(\gamma_{ij} - r_{ij}), \forall (i,j) \in L\}$.

To generate a candidate to incumbent solution as an input to the VNS heuristic, another heuristic algorithm is executed. This algorithm, the Reduction Heuristic, can be executed as many times as a feasible solution is found. Its goal is to keep removing a time-slot from the incumbent solution until an infeasible solution is found. Then, this solution is passed as input for the VNS heuristic to fix such infeasibility. Figure 4.1 illustrates this strategy. In what follows, this work details the designed algorithms for the Reduction Heuristic, a procedure of time-slot removal, the VNS perturbation, and the local search.

Figure 4.1: Illustration of the Reduction Heuristic.



Source: Own authorship

4.3.1 Reduction Heuristic

The Reduction Heuristic (ReH), presented in Algorithm 8, is a heuristic that repeatedly tries to improve the incumbent solution by removing a time-slot from the solution. As such movement might result in an infeasible solution, a VNS algorithm is executed over this solution in order to make it feasible again. If, after the VNS algorithm is done, the incumbent is feasible, clearly this is a better solution and in this case the RH updates the current best.

Algorithm 8: ReH

input : I_{MD}
output: $S^*_{MD} \in \Delta_{MD}$

$S^*_{MD} \leftarrow \text{MD-CH}(I_{MD})$
while *stopping criteria not reached* **do**
  $S' \leftarrow \text{RTS}(S^*_{MD})$
  $S' \leftarrow \text{VNS}(S')$
  if $\gamma\text{-}feasible(S')$ **then**
    $S^*_{MD} \leftarrow S'$
  end
end
return S^*_{MD}

4.3.2 Removing a time-slot

For instances with a larger number of links to schedule, it may be that the CH yields sub-optimal solutions, i.e., solutions using more time-slots than needed. Considering this scenario, the second proposed heuristic, *Remove Time Slot* (RTS), receives a solution S_{MD}, and keeps removing time-slots from S_{MD} until it becomes an optimal solution or becomes an infeasible solution. Algorithm 9 shows the steps of this heuristic. The proposed approach chooses a time-slot at random. All links transmitting in this time-slot are scheduled into the remaining channels and time-slots. These two are also randomly chosen.

Algorithm 9: RTS

input : $I_{MD}, S_{MD} \in \Delta_{MD}$
output: S'_{MD}

$\bar{t} \leftarrow \text{Random}(\{1,\dots,\tau(S)\})$
$\bar{S} \leftarrow \{\langle (i,j),(c,\bar{t})\rangle \in S\}$
$S' \leftarrow S^* \setminus \bar{S}$
foreach $(i,j) : \exists \langle (i,j),(c,\bar{t})\rangle \in \bar{S}$ **do**
  $(c,t) \leftarrow \text{Random}(\Psi(S'))$
  $S' \leftarrow S' \cup \langle (i,j),(c,t)\rangle$
end
return S'_{MD}

4.3.3 Perturbation

The perturbation procedure, exposed in Algorithm 10, has a minor difference from the perturbation of the VNS heuristic for the VRBSP (Algorithm 5). It also randomly selects k scheduled links to perturb but, differently from the previous perturbation, all the selected scheduled links will be rescheduled again into the solution in an existing channel and time-slot, these two also selected at random.

Algorithm 10: MD-PER

input : I_{MD}, S^0, k
output: $S_{MD} \in \Delta_{MD}$

$\Gamma \leftarrow \text{Reshape}(S^0)$
for k *iterations* **do**
    $\langle (i,j),(c',t') \rangle \leftarrow \text{Random}((L \times C^{20} \times T) \setminus \Gamma)$
    $\Gamma \leftarrow \Gamma \setminus \{\langle (i,j),(c,t) \rangle \in \Gamma\}$
    $\Gamma \leftarrow \Gamma \cup \{\langle (i,j),(c',t') \rangle\}$
end
return $S_{MD} = \text{DP}(\Gamma)$

4.3.4 Local Search

The local search procedure of this VNS heuristic is essentially the same as the VNS for the VRBSP, except for a few minor adaptations. For the sake of clearness, Algorithm 6 is rewritten into Algorithm 11 to reflect these changes. Note that the scheduling tuples at lines 5 and 8 consider the existence of multiple time-slots. It was also added a nested *foreach* (line 7-10) to iterate over all available time-slots. Besides, consider that solution Γ^* has, at line 5, its objective function value set to infinity. This will prevent scenarios where it would be better to remove a link from a solution.

Algorithm 11: MD-LS

input : I_{MD}, S^0
output: $S_{MD} \in \Delta_{MD}$

$\Gamma \leftarrow \text{Reshape}(S^0)$
do
  $\hat{\Gamma} \leftarrow \Gamma$
  foreach $(i, j) \in L$ **do**
    $\Gamma^{\emptyset}, \Gamma^* \leftarrow \Gamma \setminus \{\langle (i,j),(c,t)\rangle \in L \times C \times T : \langle (i,j),(c,t)\rangle \in \Gamma\}$
    foreach $c' \in C^{20}$ **do**
      foreach $\tau(\Gamma^{\emptyset})$ **do**
        $\Gamma' \leftarrow \Gamma^{\emptyset} \cup \{\langle (i,j),(c',t)\rangle\}$
        $\Gamma^* \leftarrow \text{Best}(\Gamma', \Gamma^*)$
      end
    end
    $\Gamma \leftarrow \text{Best}(\Gamma, \Gamma^*)$
  end
while $\hat{\Gamma} \neq \Gamma$;
return $S_{MD} = \text{DP}(\Gamma)$

4.4 Summary of the Chapter

This chapter introduces solution strategies for the MD-VRBSP. Namely, it introduces one non-linear mixed-integer program, two new mixed-integer programs and two new heuristics in order to find feasible solutions for this problem. In the following, Table 4.1 summarizes the main symbols (e.g., variables and constants) used in this chapter.

Table 4.1: Symbols used to propose exact formulations and heuristics for the MD-VRBSP.

Symbol	Type	Meaning
I_{MD}	Set	Valid instance for the MD-VRBSP
Γ	Var.	A solution within Δ^{20}_{MD}
β^c_{ij}	Const.	Min. SINR necessary for link (i, j) to transmit in channel $c \in C$ satisfying $r_{ij} \geq \gamma_{ij}$
$\tau(S_{MD})$	Func.	Set of available time-slots
γ-$feasible$	Func.	Returns true if and only if $\forall \langle (i,j),(c,t)\rangle \in S_{MD} \Rightarrow r_{ij} \geq \gamma_{ij}$
Random($\mathscr{A}$)	Func.	Returns a random element from a set $\mathscr{A}$
DP(Γ)	Func.	Determine the best local partitioning of $\Gamma \in \Delta^{20}_{MD}$

Chapter 5

Computational Experiments And Results

The subject of this chapter are the computational experiments. The next sections detail the instances set, the computational resources used for these experiments and its parameters, and the computed results.

5.1 Instance Set

This work adopted the same methodology that has been used in the literature to create an instance set for both problems (e.g., Costa et al. [2019], Goussevskaia et al. [2016]). Each instance consists of euclidean planes with dimensions equal to 250×250 meters. The set L of links is generated as follows: first, $|L|$ receivers are randomly positioned in the plane; then, for each receiver, a sender is randomly positioned within $l_{max} = 6\sqrt{2}$ meters from the receiver. Besides, the values of parameters N, α, and P were set to $0.0, 3.0$, and 1000.0, respectively, while the value of $|L|$ was varied by $8, 16, 32, 64, 128, 256, 512, 1024$, and 2048 links. For each value of L, we randomly generated 30 instances. In the case of MD-VRBSP, each link is associated a random value in the range $[8.6, 1201.0]$ (the minimum and maximum values for the SINR in Table 1.1) that represents the gamma value.

5.2 Computational Experiments

All experiments were run in computers equipped with a processor Intel Core i7, 24GB of RAM, and Ubuntu 18.04 LTS. Besides, this work used the Gurobi v9.1 with default parameters and its python shell interface to build the exact models. All heuristics were coded and executed using the C++ language with the compiler g++ v9.3 with no optimization flags.

The Gurobi solver was set to stop the optimization of an instance after 3600 seconds, while all heuristics can run up to 600 seconds. Besides, the experiments with the MD1 and MD2 programs considered the value of $|T|$ as equal as specified in Section 4.1.1. The experiments with a warm-start solution will use the solutions returned by the CH. In this case, it is considered that $|T| = |T(S_{MD})|$.

5.2.1 Results for the VRBSP

Table 5.1 shows the results for VR1 and VR2. The results for VR1 are depicted as follows. Column 1 refers to the number of devices, $|L|$, in the instance. The number of instances solved to optimality within 3600 seconds is reported in Column 2. Besides, Columns 3 to 7 depict, on average, the optimality gap $\frac{(ub-lb)}{lb}$, the lower and upper bounds, the number of non-zero coefficients in the restriction matrix, and the number of explored nodes in the B&B tree, respectively. Column 8 displays the required time, on average, by the Gurobi solver to solve the instances to optimality. The respective column received the 3600 value when no optimal solution was found. The columns marked with a dash '$-$' means that the Gurobi solver did not find a feasible solution (lb) within the time limit for at least one instance. Moreover, a column with a '$\times$' means that the Gurobi solver could not start the B&B tree for some instance. The same results are presented for VR2 in the last seven columns, respectively.

One can observe from Table 5.1 that both models were capable of finding optimal solutions for all instances where $|L| = 8$. Moreover, VR1 found nine optimal solutions for $|L| = 16$, while VR2 could only find 6. None of the models could find an optimal solution for higher values of $|L|$. For $|L| = \{512, 1024\}$, in particular, the Gurobi solver could not find a feasible solution for at least one instance, and therefore the respective column is marked a '$-$'. Note, however, that VR1 achieved lower upper bounds for these two groups of instance. Besides, the instances where $|L| = 2048$ showed to be the hardest ones so that both models were not capable of starting the B&B tree within 3600 seconds, hence the entire row is marked with a $\times$.

Table 5.2 displays the results of the heuristics for the VRBSP. Column 1 refers to the number of devices $|L|$ in the instance. The results for the CH are reported in Columns 2 and 3, with the resulting throughput and the required computational time to find it, respectively. The values of both columns represent the average value of 30 instances. The VNS and the BRKGA heuristics results are depicted in Columns 4 to 6 and 7 to 9, respectively. They have the same data structure. The first column is the average throughput of 30 instances; the second and third are the percentage difference between its throughput and the one found by the CH and VR1, respectively.

The results suggest that the performance of the VNS heuristic had a better performance than the CH in all instances, outperforming with up to 39.57% when $|L| = 2048$. The BRKGA

Table 5.1: Results of VR1 and VR2 for the VRBSP.

	VR1							VR2						
$\|L\|$	#opt	gap(%)	lb	ub	non-zeros	nodes	t(s)	#opt	gap(%)	lb	ub	non-zeros	nodes	t(s)
8	30	0.00	7863.33	7863.33	8664.00	838.10	1.03	30	0.00	7863.33	7863.33	59024.00	731.30	10.36
16	9	6.56	12811.26	13610.31	23088.00	724083.90	1769.82	6	6.75	12771.72	13592.82	123808.00	65365.13	1855.33
32	0	74.47	18811.92	32623.78	69216.00	214722.77	3600.00	0	66.35	18737.67	31009.14	270656.00	16834.83	3600.00
64	0	162.12	26484.92	68968.63	230592.00	22305.20	3600.00	0	160.38	25688.91	66459.55	633472.00	6609.70	3600.00
128	0	372.26	31678.01	148273.76	829824.00	3387.93	3600.00	0	411.47	28374.12	143213.40	1635584.00	35.97	3600.00
256	0	1634.37	21159.27	301920.14	3134208.00	161.50	3600.00	0	2020.97	16973.18	303040.39	4745728.00	1.00	3600.00
512	0	–	–	609509.39	12166656.00	1.00	3600.00	0	–	–	38714236.59	15389696.00	0.03	3600.00
1024	0	–	–	6221326.12	47926272.00	0.00	3600.00	0	–	–	80056012.80	54372352.00	0.00	3600.00
2048	×	×	×	×	×	×	×	×	×	×	×	×	×	×

heuristic also performed better than the CH in most of the instances, with exception for $|L| = 8$. In these instances, the CH obtained higher lower-bound values. Besides, BRKGA showed to be more suitable than the VNS heuristic when $|L| = \{32, 64\}$. Nevertheless, the values of Columns 5 and 8 reveal that the VNS heuristic remained with the best performance as the number of devices $|L|$ grows. Moreover, when $|L| = 8$, none of the heuristics could perform better than VR1, which found better lower-bounds than the BRKGA also when $|L| = 16$.

Table 5.2: Results for the CH, VNS and BRKGA algorithms for the VRBSP.

	CH		VNS			BRKGA		
$\lvert L\rvert$	lb	t(s)	lb	%CH	%VR1	lb	%CH	%VR1
8	7514.63	0.000	7843.31	4.53%	-0.26%	7114.00	-5.33%	-10.53%
16	12097.86	0.001	12838.17	6.23%	0.22%	12657.22	4.62%	-8.85%
32	18094.48	0.004	19625.57	8.84%	4.38%	19935.78	10.18%	5.63%
64	25645.32	0.012	29616.84	15.69%	11.98%	30015.96	17.04%	11.76%
128	35638.36	0.032	44096.76	23.89%	39.99%	41473.54	16.37%	23.61%
256	49242.34	0.127	61633.04	25.21%	250.63%	58606.54	19.02%	63.89%
512	65065.25	0.539	83255.05	28.07%	–	78058.58	19.97%	–
1024	84804.45	2.203	112259.58	32.47%	–	96914.34	14.28%	–
2048	110399.63	8.716	154015.61	39.57%	–	123751.66	12.09%	–

5.2.2 Results for the MD-VRBSP

Table 5.3 presents the results for MD1 and MD2, respectively. They are displayed in the same fashion as in Table 5.1. One can observe that both models successfully found optimal solutions for all instances with up to 32 devices. However, when $|L| = 64$, MD2 found 28 optimal solutions, while MD1 could only find 3. Moreover, both models failed to find an incumbent solution for at least one instance with $|L| = 128$, and the respective column is marked with '–'. In this case, MD2 achieved higher lower-bounds. Last, MD1 and MD2 failed to initialize the B&B tree for any instance with $|L| = 256$.

The results for the CH and the VNS heuristics are presented in Table 5.4 in the following order. Column 1 refers to the number of devices $|L|$ in the instance. Columns 2 and 3 show the number of time-slots needed to schedule all links and the computational time required to achieve this result. Column 4 shows the percentage difference between the solutions found by CH and MD1. The remaining columns represent the VNS results. Column 5 shows the number of time-slots needed to schedule all links, while Column 6 exposes how many times the algorithm executed the *while* loop of lines $4-13$. Columns 7 and 8 show the percentage difference between the VNS results and the solutions found by the CH and MD1, respectively.

Table 5.3: Results of the MD1 and MD2.

	MD1							MD2						
$\lvert L \rvert$	#opt	gap(%)	ub	lb	non-zeros	nodes	t(s)	#opt	gap(%)	ub	lb	non-zeros	nodes	t(s)
8	30	0.00	1.43	1.43	18490.07	0.10	0.08	30	0.00	1.43	1.43	24014.07	0.07	0.03
16	30	0.00	2.33	2.33	89817.10	11.07	0.69	30	0.00	2.33	2.33	155209.10	0.77	0.32
32	30	0.00	3.70	3.70	499041.20	50695.53	160.82	30	0.00	3.70	3.70	799549.33	0.70	3.41
64	3	27.16	6.10	4.28	3266696.80	44511.77	2876.94	28	0.97	6.07	6.00	8148840.80	3961.83	261.13
128	0	–	–	1.50	23202552.57	0.70	3600.00	0	–	–	9.31	63105157.21	2.66	3600.00
256	0	×	×	×	×	×	×	×	×	×	×	×	×	×
512	0	×	×	×	×	×	×	×	×	×	×	×	×	×
1024	0	×	×	×	×	×	×	×	×	×	×	×	×	×
2048	0	×	×	×	×	×	×	×	×	×	×	×	×	×

All values reported in Columns 2 to 8 are displayed as average values of the 30 instances.

One can see that the VNS heuristic could not improve the results of the CH in nearly half of the instances. Major improvements occurred in the small-size instances. Such a performance may be related to the number of iterations (Column 6) that the algorithm executed within 600 seconds. Less iterations means that the VNS performed less local searches, therefore decreasing the chances of a solution improvement. In fact, with exception for $|L| = 8$, the VNS heuristic had worse results than MD1, as reported in Column 8.

Table 5.4: Results for CH and VNS heuristics for the MD-VRBSP.

	CH			VNS			
$\|L\|$	$\#ts$	$t(s)$	%MD1	$\#ts$	$\#iter$	%CH	%MD1
8	1.76	0.003	-16.11%	1.43	35255.52	18.75%	0.00%
16	2.6	0.004	-9.16%	2.36	44951.96	9.23%	-1.11%
32	4.57	0.006	-17.72%	4.36	22623.30	4.59%	-14.16%
64	7.37	0.010	-17.33%	7.37	13802.50	0.00%	-17.33%
128	13.13	0.024	–	13.13	6873.03	0.00%	–
256	24.10	0.070	–	24.10	3290.40	0.00%	–
512	45.47	0.261	–	45.46	1430.46	0.02%	–
1024	87.03	1.039	–	87.03	557.26	0.00%	–
2048	170.87	4.238	–	170.86	179.33	0.005%	–

Table 5.5 shows the results when "warm-starting" MD1 and MD2 models with the VNS solutions. Models WMD1 and WMD2 represent the MD1 and MD2 programs, respectively. The results are organized in the same manner as in Table 5.3. One can see that, when compared with Table 5.3, there is a worsening of the number of instances solved to optimality. For instance, WMD1 was not capable to find 30 optimal solutions for $|L| = 32$ nor 3 optimal solutions for $|L| = 64$. Instead, it found 29 and 1 optimal solutions, respectively. Meanwhile, WMD2 was capable to find 27 optimal solutions for $|L| = 64$. Originally, it found 28 optimal solutions. But the warm-start solution showed to be an interesting option for $|L| = \{128, 256\}$. In the first case, both models presented non-trivial lower-bounds, and MD2 could improve the upper bound, thus tightening the optimality gap. Besides, for $|L| = 256$, MD1 could also compute lower gaps, while MD2 could not initialize the B&B tree. Last, there was insufficient memory to initialize the B&B tree for instances with more than 256 devices.

Table 5.5: Results of the WMD1 and WMD2.

	WMD1							WMD2						
$\lvert L \rvert$	#opt	gap(%)	ub	lb	non-zeros	nodes	t(s)	#opt	gap(%)	ub	lb	non-zeros	nodes	t(s)
8	30	0.00	1.43	1.43	12238.83	0.40	0.04	30	0.00	1.43	1.43	15914.83	0.27	0.02
16	30	0.00	2.33	2.33	54052.23	8.13	0.41	30	0.00	2.33	2.33	93500.23	0.60	0.18
32	29	0.67	3.70	3.67	370845.33	21863.67	37.10	30	0.00	3.70	3.70	799549.33	0.70	3.41
64	1	29.01	6.13	4.23	2241671.40	57905.87	3322.56	27	1.53	6.07	5.97	5592999.40	3862.77	297.88
128	0	83.10	13.13	1.90	10870246.03	0.97	3600.00	0	24.56	12.57	9.33	29571686.03	120.43	3600.00
256	0	94.92	24.07	1.11	75347896.07	0.07	3600.00	×	×	×	×	×	×	×
512	0	×	×	×	×	×	×	×	×	×	×	×	×	×
1024	0	×	×	×	×	×	×	×	×	×	×	×	×	×
2048	0	×	×	×	×	×	×	×	×	×	×	×	×	×

5.3 Discussing Research Questions

This section focus on discussing the research questions, now in the light of the results presented in the previous sections. In the following, each RQ is enumerated again and discussed.

(RQ1): *Does the new mixed-integer formulation for the VRBSP find as good or as lower optimality gaps as the one proposed in [Costa et al., 2019]?*

Answer: The results depicted in Table 5.1 show that, in 3600 seconds, the formulation from Costa et al. [2019] (VR1) performed better, i.e., generated lower gaps, than the formulation proposed in this work (VR2) in most of the instances.

One possible cause for this observation might be related to the number of explored nodes of the B&B tree by the Gurobi solver. Note that fewer nodes were explored as $|L|$ increases. Such behavior may be related to the higher number of variables, leading to a more challenging branching procedure. Nevertheless, the VR2 formulation performed at least as good as VR1 in formulations where $|L|$ equals 8, 16, and 32.

(RQ2): *Which heuristic for the VRBSP achieves better results? Can these results be considered satisfactory?*

Answer: The results reported in Table 5.2 show that the heuristics for the VRBSP were capable of providing better lower-bounds than the Gurobi solver, given its respective execution times. Nevertheless, this does not mean that their results are optimal. Note that the upper bounds computed in Table 5.1 have higher values than the lower bounds in Table 5.2. Therefore, it is expected that there is room for an improvement in the solutions returned by the heuristics.

Considering the execution time values, one can prefer the heuristics over the MILPs when response time is more important than the quality of the solution. The CH did achieve its purpose, giving feasible solutions using lower computational resources and requiring low execution time. Besides, the VNS performed better than the CH and BRKGA. In the harder instances, e.g., instances with 2048 devices, this heuristic generates solutions with an objective function value 39.51% higher than the ones found by the CH and 27.41% better than the ones found by the BRKGA heuristic.

(RQ3): *Which mixed-integer formulation is the best for the MD-VRBSP, i.e., which one gives lower optimality gaps?*

Answer: Table 5.3 show that for $|L|$ equals to 8, 16, and 32, both formulations find optimal solution for all instances. However, they do not find an equal number of optimal solutions in solutions with a greater number of devices. On the one hand, MD2 finds more optimal solutions for instances with 64 devices, suggesting a better capacity in generating lower gaps. This might be related to the Big-M value used in MD1. It is well known that formulations that use the Big-M technique can yield worse gaps due to weak linear relaxations [Bazaraa et al., 2009]. On the other hand, MD2 has more variables and runs out of memory on smaller instances than MD1.

(RQ4): *Which heuristic for the MD-VRBSP achieves better results? Can these results be considered satisfactory?*

Answer: One can observe from Tables 5.4 and 5.3 that the CH achieved, at least for the small-sized instances, similar results when compared with the optimal results found by the Gurobi solver. The same can not be inferred for the VNS, as this heuristic could not find better solutions in most of the used instances. When doing a cross-analysis of Tables 5.4 and 5.3, one can note how the objective function value from the solutions found by the CH is already near to the optimal solutions. Therefore, it is expected that the VNS heuristic needs to do more work to improve a solution. For that reason, more analysis is needed on the VNS heuristic.

(RQ5): *Does the warm-start technique succeed in improving the results of the MD-VRBSP MILP formulations?*

Answer: The use of a warm-start solution allowed WMD1 to find upper-bounds for all instances with $|L| = 256$, whereas MD1 and MD2 could only find for instances with $|L| = 64$ or below. But at the same time, this technique reduced the number of solutions solved to optimality in instances where $|L| = \{32, 64\}$. Therefore, one should analyze the trade-off imposed by these results. If one seeks solutions with more quality, i.e., lower optimality gaps, then one must choose not to use the warm-start technique. However, when scalability is a more important factor, a more intelligent option would be to initialize the MD-VRBSP models with an initial solution.

Chapter 6

Conclusions

This work dealt with two problems belonging to the computer networks field: the VRBSP and the MD-VRBSP. Both problems seek optimal network schedulings but with different objectives. The former wants the schedule with the highest throughput, while the latter seeks the scheduling having the lowest number of time-slots. Costa et al. [2017] introduced the VRBSP in his work, and it proposes the first BRKGA heuristic for this problem. Later on, [Costa et al., 2019] proposed the first mixed-integer formulations. Besides, an equivalent mixed-integer formulation and a VNS heuristic are proposed in this work. Moreover, this is the first work to address the MD-VRBSP and it follows a similar strategy to previous VRBSP works, by proposing MILP formulations and heuristics.

The computational experiments with the VRBSP suggest that the formulation from Costa et al. [2019] had a better performance, i.e., found lower optimality gaps than the formulation proposed in this work even though the alternative MILP formulation does not have coupling constraints. The results suggest that this behavior was due to the higher number of variables in the proposed formulation, yielding higher computational time in the root node. Meanwhile, the VNS heuristic outperforms with a factor of 250.63%, on average, the lower bounds achieved by VR1. The same cannot be said about the BRKGA heuristic, which outperformed in 63.89% the solutions of VR1, therefore being worse than the VNS heuristic given its time-limits. The difference in the performance of this heuristics is more evident in the harder instances, i.e., instances with higher number of devices. One possible cause is that the BRKGA could not diversify its population, therefore getting stuck in local optima.

This work proposed and presented the first results for the MD-VRBSP. The results suggest that the proposed heuristics find near-optimal solutions for the instances tested, as the generated solutions for the harder instances that could be tested were 17.33% worse than the ones found by the MILP formulations. Such a hypothesis was confirmed by using its solutions as warm-start solutions for the Gurobi solver. This observation explains why the VNS heuristic did not perform well for the instance set used in the experiments. Nevertheless, a deeper investigation is necessary to confirm whether the instance set is “easy” or if the VNS heuristic is not suited for this problem.

Future works can explore new mixed-integer formulations for these problems. In this context, one might develop decomposition techniques, especially when considering formulation

VR2 for the VRBSP and MD2 for the MD-VRBSP, as these two formulations have a larger number of variables (consequently, more complicating variables) for the instance set used in this work. Besides, new meta-heuristics for the MD-VRBSP should be adopted in future works. A possible challenge in all these works is to deal with the potential symmetry that can appear in the solutions.

Bibliography

Matthew Andrews and Michael Dinitz. Maximizing capacity in arbitrary wireless networks in the sinr model: Complexity and game theory. In *IEEE INFOCOM 2009*, pages 1332–1340, 2009.

E. I. Ásgeirsson and Pradipta Mitra. On a game theoretic approach to capacity maximization in wireless networks. In *2011 Proceedings IEEE INFOCOM*, pages 3029–3037, 2011. doi: 10.1109/INFCOM.2011.5935146.

D. J. Baker, J. E. Wieselthier, and Anthony Ephremides. A distributed algorithm for scheduling the activation of links in a self-organizing mobile radio networks. In *IEEE Int. Conference Communications*, pages 2F6.1–2F6.5, 1982.

Mokhtar S. Bazaraa, John J. Jarvis, and Hanif D. Sherali. *Linear Programming and Network Flows*. John Wiley & Sons, 2009.

Patrik Bjorklund, Peter Varbrand, and Di Yuan. Resource optimization of spatial tdma in ad hoc radio networks: A column generation approach. In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)*, volume 2, pages 818–824. IEEE, 2003.

Douglas M Blough, Giovanni Resta, and Paolo Santi. Approximation algorithms for wireless link scheduling with sinr-based interference. *IEEE/ACM Transactions on Networking*, 18(5): 1701–1712, 2010.

Steven A. Borbash and Anthony Ephremides. Wireless link scheduling with power control and sinr constraints. *IEEE Transactions on Information Theory*, 52(11):5106–5111, 2006.

Deepti Chafekar, V. S. Anil Kumar, Madhav V. Marathe, Srinivasan Parthasarathy, and Aravind Srinivasan. Approximation algorithms for computing capacity of wireless networks with sinr constraints. In *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*, pages 1166–1174. IEEE, 2008.

C. Chaudet and Y. Haddad. Wireless software defined networks: Challenges and opportunities. In *2013 IEEE International Conference on Microwaves, Communications, Antennas and Electronic Systems (COMCAS 2013)*, pages 1–5. IEEE, 2013.

David Coleman. *802.11ax for dummies*. Extreme Networks, 2020.

Jose Mauricio Costa, Pedro Pappa Paniago, Thiago Ferreira Noronha, and Marcos Augusto Vieira Menezes. A biased randomkey genetic algorithm for the multi-period, multi-rate and multi-channels with variable bandwidth scheduling problem. In *The 12th edition of the Metaheuristics International Conference (MIC 2017)*, 2017.

José Maurício Costa, Pedro P. Paniago, Joaquim de Andrade, Thiago F. Noronha, and Marcos A. M. Vieira. Integer linear programming formulations for the variable data rate and variable channel bandwidth scheduling problem in wireless networks. *Computer Networks*, 165, 2019.

Rene L. Cruz and Arvind V. Santhanam. Optimal routing, link scheduling and power control in multihop wireless networks. In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)*, volume 1, pages 702–711. IEEE, 2003.

Anthony Ephremides and Thuan V Truong. Scheduling broadcasts in multihop radio networks. *IEEE Transactions on communications*, 38(4):456–460, 1990.

Fred Glover and Eugene Woolsey. Converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations Research*, 22(1):180–182, 1974.

Olga Goussevskaia, Yvonne Anne Oswald, and Rogert Wattenhofer. Complexity in geometric sinr. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 100–109, 2007. doi: 10.1145/1288107.1288122.

Olga Goussevskaia, Roger Wattenhofer, Magnús M Halldórsson, and Emo Welzl. Capacity of arbitrary wireless networks. In *IEEE INFOCOM 2009*, pages 1872–1880. IEEE, 2009.

Olga Goussevskaia, Yvonne Anne Pignolet, and Roger Peter Wattenhofer. *Efficiency of wireless networks: Approximation algorithms for the physical interference model*. Now Publishers Inc, 2010.

Olga Goussevskaia, Luiz FM Vieira, and Marcos AM Vieira. Wireless multi-rate scheduling: From physical interference to disk graphs. In *37th Annual IEEE Conference on Local Computer Networks*, pages 651–658. IEEE, 2012.

Olga Goussevskaia, Luiz FM Vieira, and Marcos AM Vieira. Wireless scheduling with multiple data rates: From physical interference to disk graphs. *Computer Networks*, 106:64–76, 2016.

Piyush Gupta and Panganmala R. Kumar. The capacity of wireless networks. *IEEE Transactions on information theory*, 46(2):388–404, 2000. doi: 10.1109/18.825799.

Magnús M Halldórsson. Wireless scheduling with power control. In *European Symposium on Algorithms*, pages 361–372. Springer, 2009. doi: https://doi.org/10.1007/978-3-642-04128-0_33.

Magnús M Halldórsson and Pradipta Mitra. Wireless capacity with oblivious power in general metrics. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1538–1548. SIAM, 2011a.

Magnús M Halldórsson and Pradipta Mitra. Nearly optimal bounds for distributed wireless scheduling in the sinr model. In *International Colloquium on Automata, Languages, and Programming*, pages 625–636. Springer, 2011b.

Magnús M Halldórsson and Pradipta Mitra. Wireless capacity and admission control in cognitive radio. In *2012 Proceedings IEEE INFOCOM*, pages 855–863. IEEE, 2012.

Magnús M. Halldórsson and Roger Wattenhofer. Wireless communication is in apx. In *International Colloquium on Automata, Languages, and Programming*, pages 525–536. Springer, 2009.

Thomas Kesselheim. Approximation algorithms for wireless link scheduling with flexible data rates. In *European Symposium on Algorithms*, pages 659–670. Springer, 2012.

Thomas Kesselheim and Berthold Vöcking. Distributed contention resolution in wireless networks. In *International Symposium on Distributed Computing*, pages 163–178. Springer, 2010. doi: https://doi.org/10.1007/978-3-642-15763-9_16.

Sastry Kompella, Jeffrey E. Wieselthier, and Anthony Ephremides. A cross-layer approach to optimal wireless link scheduling with sinr constraints. In *MILCOM 2007-IEEE Military Communications Conference*, pages 1–7. IEEE, 2007.

Ritesh Maheshwari, Shweta Jain, and Samir R. Das. A measurement study of interference modeling and scheduling in low-power wireless networks. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 141–154, 2008.

Ritesh Maheshwari, Jing Cao, and Samir R. Das. Physical interference modeling for transmission scheduling on commodity wifi hardware. In *IEEE INFOCOM 2009*, pages 2661–2665. IEEE, 2009.

R. Masoudi and A. Ghaffari. Software defined networks: A survey. *Journal of Network and computer Applications*, 67:1–25, 2016.

Nenad Mladenović and Pierre Hansen. Variable neighborhood search. *Computers & operations research*, 24(11):1097–1100, 1997.

Thomas Moscibroda and Roger Wattenhofer. The complexity of connectivity in wireless networks. In *Infocom*, pages 1–13, April 2006.

Thomas Moscibroda, Roger Wattenhofer, and Yves Weber. Protocol design beyond graph-based models. In *Proc. of the ACM Workshop on Hot Topics in Networks (HotNets-V)*, pages 25–30. Citeseer, November 2006a.

Thomas Moscibroda, Rogert Wattenhofer, and Aaron Zollinger. Topology control meets sinr: The scheduling complexity of arbitrary topologies. In *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 310–321, May 2006b. doi: 10.1145/1132905.1132939.

Thomas Moscibroda, R. Rejaie, and Roger Wattenhofer. How optimal are wireless scheduling protocols? In *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*, pages 1433–1441. IEEE, IEEE, May 2007. doi: 10.1109/INFCOM.2007.169.

H. Moura, G. V. C. Bessa, M. A. M. Vieira, and D. F. Macedo. Ethanol: Software defined networking for 802.11 wireless networks. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015.

Theodore S. Rappaport. *Wireless communications: principles and practice*. Prentice Hall, 2nd edition, 2002.

Paolo Santi, Ritesh Maheshwari, Giovanni Resta, Samir Das, and Douglas M. Blough. Wireless link scheduling under a graded sinr interference model. In *Proceedings of the 2nd ACM international workshop on Foundations of wireless ad hoc and sensor networking and computing*, pages 3–12, 2009.

Fabio RJ Vieira, José F de Rezende, and Valmir C Barbosa. Scheduling wireless links by vertex multicoloring in the physical interference model. *Computer Networks*, 99:125–133, 2016.

Markus Voelker, Bastian Katz, and Dorothea Wagner. *On the complexity of scheduling with power control in geometric SINR*. Univ., Fak. für Informatik, 2009.

L. A. Wolsey. *Integer Programming*. John Wiley & Sons, 2020.

Marco Zuniga and Bhaskar Krishnamachari. Analyzing the transitional region in low power wireless links. In *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.*, pages 517–526. IEEE, 2004.