INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Exploring Graph Coloring Heuristics for Optical Networks Planning

Inês Maria Leandro Duarte

Master in, Telecommunications and Computer Engineering

Supervisor:
Prof. Doctor Luís Gonçalo Lecoq Vences e Costa Cancela, Assistant Professor,
ISCTE-IUL

Co-supervisor:
Prof. Doctor João Lopes Rebola, Assistant Professor,
ISCTE-IUL

October, 2020

TECNOLOGIAS
E ARQUITETURA

Exploring Graph Coloring Heuristics for Optical Networks Planning

Inês Maria Leandro Duarte

Master in, Telecommunications and Computer Engineering

Supervisor:
Prof. Doctor Luís Gonçalo Lecoq Vences e Costa Cancela, Assistant Professor,
ISCTE-IUL

Co-supervisor:
Prof. Doctor João Lopes Rebola, Assistant Professor,
ISCTE-IUL

October, 2020

Ao meu avô

Acknowledgments

I would like to express all my gratitude and appreciation to all those who directly or indirectly contributed to this work being accomplished. To all, I would like to express my sincere thanks.

First of all, I would like to thank professors Luís Cancela and João Rebola for allowing me to do this dissertation and for their continuous and endless guidance, assistance, constructive critiques and constant supervision that made this work possible.

To my friends who have been by my side throughout this period, especially to Bárbara Costa and Bernardo Alves for their interest in listening to my concerns and doubts, but also to the small victories that I have achieved throughout this process.

I cannot forget to thank my family, mother, father and brother, who always supported me and showed interest in my work. They were the first to listen to me in moments of discouragement.

Resumo

As redes óticas são essenciais nas comunicações globais atuais e, o estudo de ferramentas de planeamento que utilizem eficientemente os recursos da rede são cruciais aos operadores de rede. A atribuição de comprimentos de onda, juntamente com o encaminhamento, são funções críticas em todas as ferramentas de planeamento de redes óticas. Esta dissertação foca-se no estudo de algoritmos de atribuição de comprimentos de onda baseados em técnicas de Coloração de Grafos.

Na presente dissertação analisamos o desempenho da heurística Greedy, uma heurística de Coloração de Grafos tipicamente aplicada ao planeamento de redes óticas, assim como as heurísticas Degree of Saturation (DSATUR) and Recursive Largest First (RLF), em diversos cenários de redes reais. Estas duas últimas heurísticas, tanto quanto sabemos, ainda não foram aplicadas no contexto de redes óticas. Foram realizadas inúmeras simulações, utilizando topologias de redes reais, como as redes COST 239, e CONUS considerando uma topologia lógica em malha completa e concluímos que as heurísticas DSATUR e RLF podem superar a heurística Greedy em cenários de rede onde existem vários clusters de rede interligados por apenas uma ou duas ligações. Nestas redes, as heurísticas RLF e DSATUR, proporcionam menos 9 e 5 comprimentos de onda, respetivamente, do que a heurística Greedy. Apesar de gerarem menos comprimentos de onda, verificamos que estas heurísticas necessitam de um tempo de computação superior ao da heurística Greedy. Além de terem sido estudadas estas heurísticas, também foram estudadas as heurísticas tradicionais First Fit e Most-Used e concluímos que têm um desempenho semelhante à heurística Greedy.

Palavras-chave: Coloração de Grafos, DSATUR, Greedy, Redes Óticas, RLF.

Abstract

Optical networks are essential in today's global communications, and the study of planning tools that efficiently allocate network resources is crucial to network providers. The assignment of wavelengths, alongside routing, are critical functions in all optical network planning tools. This dissertation focuses on the study of wavelength assignment algorithms based on Graph Coloring techniques.

In this dissertation, we analyse the performance of the usual Greedy heuristic, a well-known Graph Coloring heuristic applied to optical network planning, as well as the Degree of Saturation (DSATUR) and Recursive Largest First (RLF) heuristics, in several real network scenarios. These last two heuristics, to the best of our knowledge, have not yet been applied in the context of optical networks. Extensive simulations have been performed, using real network topologies, such as COST 239, and CONUS networks, considering a full mesh logical topology, and we conclude that DSATUR and RLF heuristics can outperform Greedy heuristic in network scenarios where there are several network clusters interconnected by only one or two links. In these cases, the RLF and DSATUR heuristics provide less 9 and 5 wavelengths respectively than the Greedy heuristic. Despite generating fewer wavelengths, we have verified that these heuristics need a higher computing time than the Greedy heuristic. Besides these heuristics, the traditional First Fit and Most-Used heuristics were also studied, and lead to performance similar to the Greedy heuristics.

Keywords: DSATUR, Graph Coloring, Greedy, Optical Networks, RLF.

Contents

List of Figures

List of Tables

List of Acronyms

APS	**A**utomatic **P**rotection **S**witching
AWG	**A**rrayed **W**aveguide **G**ratings
CONUS	**Con**tinental **U**nited **S**tates
DSATUR	**D**egree of **SATUR**ation
DWDM	**D**ense **W**avelength - **D**ivision **M**ultiplexing
FEC	**F**orward **E**rror **C**orrection
GT-ITM	**G**eorgia **T**ech **I**nternet **T**opology **M**odeler
ILP	**I**nteger **L**inear **P**rogramming
IP	**I**nternet **P**rotocol
ITU	**I**nternational **T**elecommunication **U**nion
LCoS	**L**iquid **C**rystal on **S**ilicon
MEMS	**M**icro **E**lectro **M**echanical **S**ystems
MPLS	**M**ulti-**P**rotocol **L**abel **S**witching
NSFNET	**N**ational **S**cience **F**oundation **Net**work
OA	**O**ptical **A**mplifier
O-E-O	**O**ptical - **E**lectrical - **O**ptical
OSNR	**O**ptical **S**ignal to **N**oise **R**atio
OTH	**O**ptical **T**ransport **H**ierarchy
OTM	**O**ptical **T**ransport **M**odule
OTN	**O**ptical **T**ransport **N**etwork
OTU	**O**ptical **T**ransport **U**nit
RLF	**R**ecursive **L**argest **F**irst
ROADM	**R**econfigurable **O**ptical **A**dd - **D**rop **M**ultiplexer
RWA	**R**outing and **W**avelength **A**ssignment
SDH	**S**ynchronous **D**igital **H**ierarchy
STM	**S**ynchronous **T**ransport **M**odule
UBN	**U**S **B**ackbone **N**etwork
WA	**W**avelength **A**ssignment
WDM	**W**avelength **D**ivision **M**ultiplexing
WSS	**W**avelength **S**elective **S**witch

CHAPTER 1

Introduction

1.1. Motivation and Background

The evolution of optical networks has undergone significant advances throughout the years as innovative technologies are developed in order to take increasing advantage of the transport capacity of optical fibres. When the optical networks appeared, the optical signal passed through an optical-electrical-optical (O-E-O) conversion and an electronic processing in all intermediate nodes along its path was performed [**1**]. These first networks are called opaque networks. With the development of optical switching technologies such as Micro-Electro-Mechanical Systems (MEMS) and multiplexing/desmultiplexing such as Arrayed Waveguide Gratings (AWG) transparent optical networks emerged [**1**]. In these networks, the optical path is switched only in optical domain, using networks elements, which are nowadays denominated reconfigurable optical add-drop multiplexers (ROADMs), which reduce the network cost, since there is no need for O-E-O conversions or electronic processing at each node. The optical signal in transparent networks is routed and transmitted through the network based on its wavelength, which means that the signal is routed independently of its bit rate, modulation format or protocol [**2**].

Network planning is focused on the details of how to accommodate the traffic that will be carried by the network, which includes selecting how a particular demand should be routed, protected and groomed in the network, and which wavelength(s) in the system spectrum should be assigned to carry that demand [**1**]. The planning of a network is important because it sets the optimal strategy to accommodate a set of demands, denominated network traffic. There are different types of problems related to planning in a transport network, such as: defining the localization of nodes and design a physical topology; traffic routing and dimensioning of links capacity; wavelengths assignment; traffic grooming; protection and rehabilitation; and equipment selection [**3**].

The different planning problems can be solved using integer linear programming (ILP) or heuristics solutions. ILP methodologies require a very long execution time and are impractical for large networks, since to find the optimal solution they consider the whole solution space [**1**]. The main objective of heuristics solutions is to find a solution in a reasonable amount of time that is good enough for the problem in question. The solution may not be optimal to solve a particular problem, but may lead to very satisfactory results. These heuristics are important because finding optimal solutions may require high and unbearable computational times [**1**].

In this work, several heuristics to solve Routing and Wavelength Assignment (RWA) problems in transport optical networks with static traffic will be studied. The goal of

a static RWA scenario is to minimise the number of wavelengths to be assigned in a specific number of optical paths that compose the network. The wavelength assignment heuristics that will be studied with more detail in this work are Graph Coloring heuristics [**4**]. This heuristic assigns wavelengths to a network based on the coloring of the vertices. In the end, the total number of distinct colors used to color a graph corresponds to the total number of wavelengths assigned to the paths of a network. Besides the application to network planning, Graph Coloring heuristics are also applied in different areas, like producing sports schedules or solving Sudoku puzzles.

1.2. Objectives

The goal of this work is to study some routing and wavelength assignment algorithms for network with static traffic. Firstly, the routing algorithm *Yen's* k-shortest path is implemented, in order to route the demands of the network [**1**]. Then, the wavelength assignment algorithms are applied. The wavelength assignment algorithms studied and implemented in this work are the First Fit, Most Used and, with main focus, several Graph Coloring heuristics [**1**]. The Graph Coloring algorithms analysed and developed are the Greedy, degree of saturation (DSATUR) and Recursive Largest First (RLF) algorithms [**5**]. Usually, the most commonly Graph Coloring algorithm used for RWA is the Greedy, because it is simple and easy to deploy. For this reason, it is necessary to analyse whether the DSATUR and RLF algorithms are more effective to solve the wavelength assignment problem than the Greedy algorithm, and which benefits they bring to the network planning.

Initially, a performance study of the three Graph Coloring algorithms namely, the Greedy, DSATUR and RLF, will be carried out as a function of the number of network vertices and their degree, for randomly generated networks, in order to have some insight into their behaviors and their advantages and disadvantages. At a later stage, all these algorithms, from the routing algorithm to the wavelength assignment algorithms, will be applied to real optical networks. The real optical networks analysed in this work are COST239 [**6**], NSFNET [**7**], UBN [**8**], two variations of the CONUS network, one with 30 vertices and other with 60 vertices [**9**] and two variations of the network generated by the GT-ITM tool in [**10**], one with 27 vertices and other with 34 vertices.

1.3. Dissertation Organization

This dissertation is organised in 5 chapters as described in the following:

Chapter 1 presents the background and motivation of this work, as well as the identification of its main objectives. It also shows how this dissertation is organised and highlights the main contributions of this dissertation.

Chapter 2 focuses on the main concepts in the area of optical networks, from topologies, architectures and network elements, to the definition of the adjacent matrix and the traffic matrix of the network. In addition, the RWA problem is also defined, theoretically

describing the routing and wavelength assignment algorithms that will be used in this work, giving more emphasis to the Graph Coloring heuristics.

Chapter 3 presents through a flowchart, the planning tool for optical networks developed in this work, describing with detail, the *Yen's* k-shortest path routing algorithm and the wavelength assignment algorithms (First Fit, Most Used and Graph Coloring heuristics). Since Graph Coloring heuristics are the main study goal of this work, the pseudocodes of the Greedy, DSATUR and RLF algorithms that served as the starting implementation point are presented. After implementation in the software tool, a study is carried out to evaluate the performance of these Graph Coloring algorithms as a function of the number of vertices and their degree. This study is performed with the random graph generation and validated in comparison with other works [**5**].

In Chapter 4, the RWA tool is applied to real networks (COST 239, NSFNET, UBN, two versions of the CONUS network and two networks generated by the GT-ITM tool) and to ring networks of 25, 35 and 45 vertices. From the application of this tool, the number of wavelengths assigned and the computation time of each network are presented, for each wavelength assignment algorithm studied. The solutions obtained by First Fit and Most Used algorithms are also presented and compared with the results obtained by the Graph Coloring algorithms.

Finally, Chapter 5 summarises the main conclusions of this work and outlines possible directions for future work in this area.

1.4. Main Contributions

The main contributions of this dissertation are the following:

- Implementation of three Graph Coloring algorithms, Greedy, DSATUR and RLF, applied to the planning of optical networks, specifically to solve the RWA problem;
- Performance study of these Graph Coloring algorithms as a function of the number of vertices and their degree and also the corresponding computation time;
- Study of the behavior of these algorithms when applied to several real networks and comparison with the results of First Fit and Most Used algorithms;
- The DSATUR and RLF algorithms show a better performance in networks that have a smaller number of links connecting several clusters of nodes.

CHAPTER 2

Fundamental Concepts in Optical Networks

2.1. Introduction

This chapter covers the basic concepts in the area of the optical network, from topologies, architectures and network elements to the planning tool. In section 2.2, the graph concept, which is essential in telecommunication network analysis, its relationship with physical and logical topologies and their definition is outlined. This section also explains the definition of an adjacency matrix and a traffic matrix. Next, in section 2.3, a brief explanation of the Optical Transport Network (OTN) architecture is provided, in order to have a better overview of how information is transported in an optical network. In section 2.4, the Reconfigurable Optical Add-Drop Multiplexer (ROADM) network element is briefly analysed. At last, section 2.5 discusses the main objective of this dissertation, the RWA. The problem of RWA can be defined in two parts separately. First, the routing algorithms (e.g. Dijkstra and *Yen's* k-shortest path) that will be studied throughout this work are described. Then, the wavelength assignment algorithms are presented. In the wavelength assignment, the major focus is on the three Graph Coloring techniques, whose algorithms are Greedy, DSATUR and RLF.

2.2. Network Representation

A telecommunications network can be represented schematically through a graph [**1**]. A graph is a well-defined mathematical model that represents objects and their relationship. In other words, a graph is defined geometrically as a set of points named vertices (or nodes) interconnected by a set of lines, known as edges. The edge represents a direct connection between two adjacent nodes [**11**]. Formally, graphs are represented as $G=(V,E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ represents the set of vertices (nodes) of the network and $E = \{e_1, e_2, \ldots, e_n\}$ the set of edges (links).

Figure 2.1 shows a simple example of a network and the corresponding graph $G = (\{v_1, v_2, \ldots, v_4\}\{e_1, e_2, \ldots, e_6\})$, with 4 vertices (nodes) and 6 edges (links). Each link has an associated cost whose weight value ranges from e_1 to e_6. The cost of the link can be represented, for example, by the distance between nodes or by the link delay. These link costs will be discussed later in this section.

The way network nodes are physically connected to each other defines the physical topology. On the other side, the way how information is distributed between nodes defines the network logical topology [**1**]. Taking Figure 2.2 as an example, it can be assumed that node v_1 functions as a distribution node and that all communications between the different nodes pass through node v_1. As a result, the physical topology in Figure 2.2(a) is a mesh

FIGURE 2.1. Example of a 4 nodes network and its corresponding graph.

topology, and the corresponding logical topology represented in Figure 2.2(b) is a star topology.

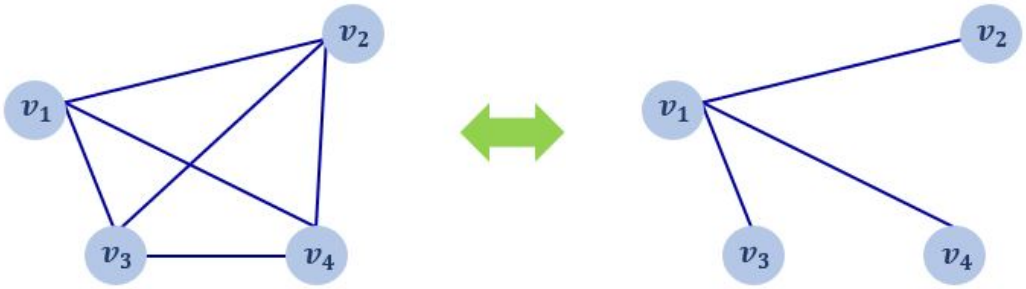

FIGURE 2.2. (a) Physical topology (b) Logical topology.

In telecommunications networks, there is a huge variety of physical topologies. The adequate choice of the topology is an extremely important step in the network planning process since the various existing topologies can influence the type of service that the network can provide [**1**]. Topologies are divided into two different types: non-shared way and shared way [**11**] as shown in Figure 2.3.

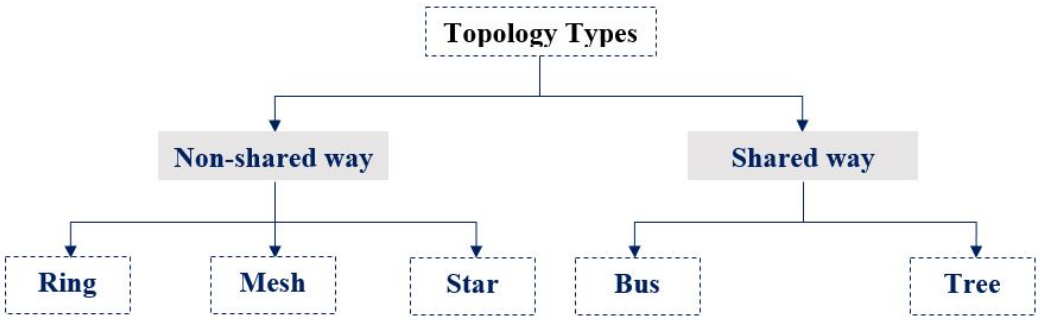

FIGURE 2.3. Different types of topologies.

A link (which is represented by lines in Figure 2.2) is a connection between two nodes (e.g. $e_1 = (v_1, v_2)$), and has an associated cost, which can be the geographical distance, delay, price, number of hops, etc. A link may be considered unidirectional or bidirectional. Typically, a bidirectional link is implemented using two unidirectional links. A network graph with unidirectional links is considered a directed graph (Figure 2.4(b)), whereas a graph with bidirectional links is designated an undirected graph (Figure 2.4(a)). In this work, a bidirectional link is assumed to be implemented by two optical fibres (one pair of fibre), one fibre for each transmission direction. An optical link includes optical fibres and possibly optical amplifiers (OA). The section of the link between an OA and a node or between two OAs is called a span [**3**].

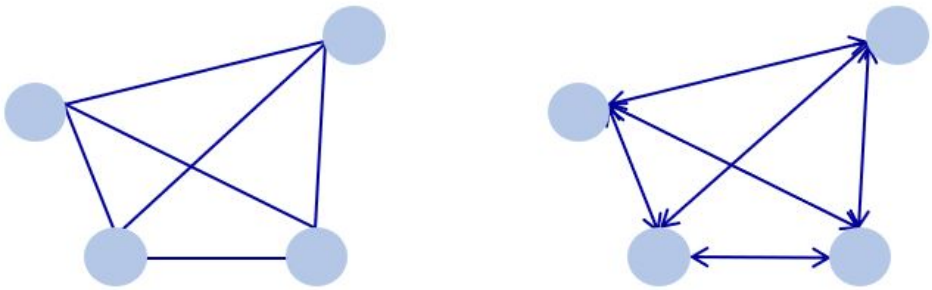

FIGURE 2.4. (a) Undirected graph and (b) Directed graph.

In Figure 2.5, a link between two nodes is represented. The two nodes are located at the beginning and the end of the link and can be optical and/or electrical devices. and in Figure are represented as ROADMs nodes. In Figure 2.5, the connection between the ROADMs nodes is composed of two inline OA, a post and a pre-OA and three fibre spans.

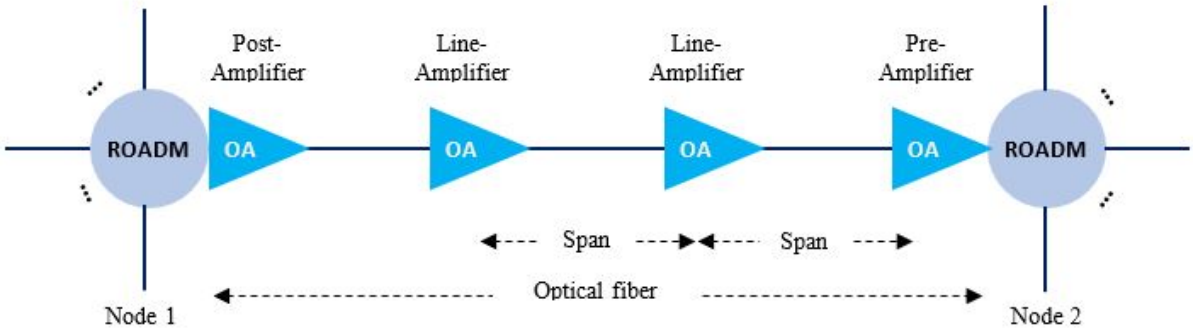

FIGURE 2.5. An example of a link between two ROADMs nodes.

In long-haul and metro networks, in order to amplify the signal in both directions, the links have one pair of fibres for bidirectional communication and unidirectional amplifiers in each direction. As shown in Figure 2.5, between the post and pre-amplifiers, there are several line amplifiers whose gain depends on the fibre spans length.

A path is a sequence of links that begins at a source node s and terminates at a destination node t (s-t path). Depending on whether the path links are unidirectional or

bidirectional, the path can also be considered as directed or undirected, respectively. In optical networks, a path is denominated a lightpath. Generally, in a transparent network, an optical path is implemented using a wavelength [**3**].

In telecommunication networks, the paths are used to carry traffic from one side of the network to the other. Traffic is defined as the set of services (e.g. data, video, voice) that is carried on the network. A demand is used to represent an individual demand between two or more nodes and corresponds to a logical link. In unicast applications, demands are performed between two nodes and are considered bidirectional and symmetric.

A connection represents a path that has been reserved for carrying a demand. The configuration process of the equipment (node) to support this demand is called provisioning.

As said previously, the physical topology refers to the physical connections between the different nodes of the network, using physical media (e.g. optical fibres). Besides representing the physical topology through a graph (as shown in Figure 2.2(a)), this topology can also be represented through an adjacency matrix G.

$$\boldsymbol{G} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \tag{2.1}$$

Generally, an adjacency matrix has a NxN dimension, where N represents the number of nodes. An example of 4x4 adjacency matrix is represented by G in eq. (2.1). When there is a physical connection between nodes i and j, the element $g_{ij} = 1$, otherwise $g_{ij} = 0$ [**3**]. Note that the adjacency matrix G, represented in eq. (2.1) considers bidirectional links.

On the other hand, a logical topology describes how traffic flows on the network and can also be defined through a graph (as shown in Figure 2.2 (b)). The traffic can be characterised by the number of traffic demands or logical connections [**2**]. Traffic demands can be unidirectional or bidirectional. It is also possible to describe the logical topology through a demand matrix Q, as shown in eq. (2.2) for the logical topology represented in Figure 2.2 (b).

$$\boldsymbol{Q} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.2}$$

If there is a traffic demand between nodes i and j, the element q_{ij} is set as $q_{ij} = 1$, otherwise, if there is no traffic demand $q_{ij} = 0$ [**3**].

The traffic matrix describes the traffic volume during a period of time between all nodes, independently of whether they are adjacent or not. The type of traffic considered in this work does not vary over time, corresponding to a static traffic scenario [**2**].

Transport networks carry different types of client signals (e.g. STM-1, GbE, etc). These client signals are then converted into adequate signals to be used in the optical transport network. It should be noted that in Synchronous Digital Hierarchy (SDH) networks, traffic units correspond to STM-N signals and, in OTN, traffic units are defined in terms of the Optical Transport Unit (OTU)-k signals. Figure 2.6, shows a graph that represents a network with four nodes that transport traffic units in GbE and the respective traffic matrix also in GbE is shown in eq. (2.3). In order to convert the matrix T into OTU-2 signals, it should be noted that one OTU-2 supports 8 GbE signals. The corresponding matrix T in OTU units is shown in eq. 2.4.

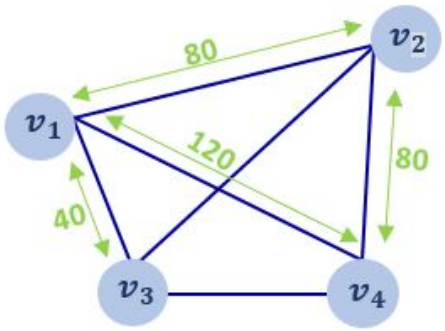

FIGURE 2.6. Example of bidirectional traffic flows between different nodes.

$$\boldsymbol{T} = \begin{bmatrix} 0 & 80 & 40 & 120 \\ 80 & 0 & 0 & 80 \\ 40 & 0 & 0 & 0 \\ 120 & 80 & 0 & 0 \end{bmatrix} \tag{2.3}$$

$$\boldsymbol{T(OTU\text{-}2)} = \begin{bmatrix} 0 & 10 & 5 & 15 \\ 10 & 0 & 0 & 10 \\ 5 & 0 & 0 & 0 \\ 15 & 10 & 0 & 0 \end{bmatrix} \tag{2.4}$$

2.3. Network Architecture

A telecommunications network is organised in two layers, the service layer and the transport layer [**12**], as shown is Figure 2.7.

Each layer offers well-defined services to the top layer. In other words, the service layer (which is the top layer) acts as the client of the transport layer. The service layer, which defines the service (e.g. video, voice and data), guarantees the collection, aggregation and introduction of information in the network for the users, while the transport

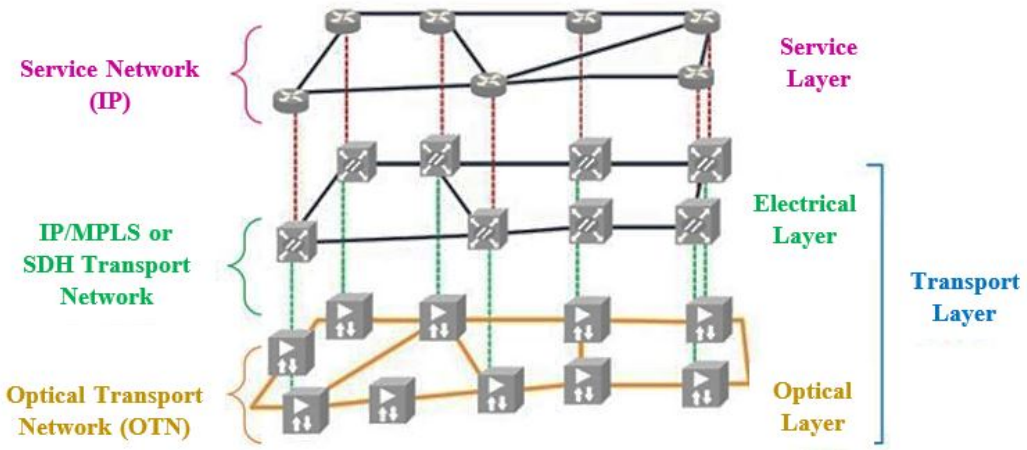

FIGURE 2.7. Layer architecture of a Telecommunications network [**13**].

layer is responsible for choosing the path between source and end nodes that better guarantees the quality of service and reliability on the information transmission. This layer is a platform that guarantees transmission, multiplexing, routing, protection and supervision and capacity provising functionalities. Also, it ensures a transparent, reliable and independent transfer of information services at a specific distance. It is also made up of different interconnected network elements, depending on the chosen physical topology [**11**]. Within the transport layer, there are two sub-layers: the electrical layer and the optical layer. Internet Protocol (IP) packets can be transported in the electrical layer (almost all information is transported in IP packets, nowadays) through a SDH transport network or a Multi-Protocol Label Switching (MPLS) transport network. In the optical layer, the SDH frames or IP/MPLS packets are encapsulated in OTN frames [**2**].

The OTN architecture is described in ITU-T (International Telecommunication Union) recommendation G.872 [**14**] and comprises functionalities such as transport, aggregation, routing, supervision and survival of client signals that are processed in the electrical domain and transported through the optical domain [**2**]. OTN combines the benefits of SDH and WDM technologies with transmission and bandwidth expansion capabilities and also monitoring capabilities. This technology offers advantages of optical monitoring capability and use of FEC (Forward Error Correction) codes, allowing a higher number of wavelengths to travel on a single fibre and allowing the signal to stay in the optical domain during a longer distance, decreasing in this way, the number of regenerators required, compared to SDH technology [**3**].

OTN defines an Optical Transport Hierarchy (OTH), conceptually similar to SDH. OTH is structured in two steps: the first step occurs in the electrical domain and the second one in the optical domain, as shown in Figure 2.8.

The first step consists in mapping the tributary signals (SONET/SDH, Ethernet, SAN) in a fixed length frame and add the appropriate headers, leading to the formation

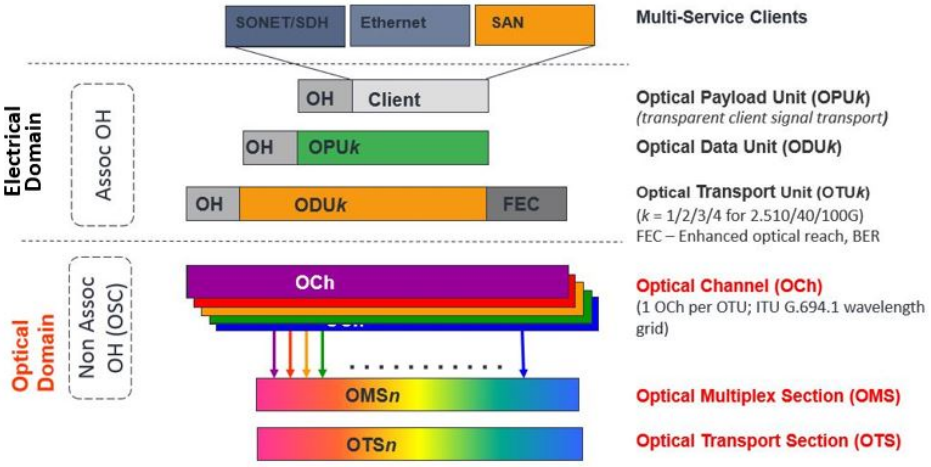

FIGURE 2.8. OTH structure [**15**].

of the OTU-k entity. The value of k is associated with the binary rate of the OTUs (OTU-1 = 2.67Gbps; OTU-2 = 10.7Gbps; OTU-3 = 43Gbps e OTU-4 = 112Gbps) [**2**]. The second domain comprises the formation of the optical channels, multiplexing and in the introduction of appropriate headers and leads to the formation of the Optical Transport Module (OTM) entity [**3**].

2.4. Network Node

A ROADM is a network node that allows for dynamically express wavelengths and add and drop wavelengths to/from the optical network [**16**]. The main characteristic of these network nodes that leads to higher flexible networks is the fact that it allows to remotely through software reconfigure the add/drop of wavelengths, without the need for manual reconfiguration of connections, so optical paths can be established and terminated according to necessities [**17**].

The ROADM degree is associated with the number of switching directions or the number of optical fibre pairs that reach and leave the ROADM and can range typically from a minimum of two up to eight degrees [**17**]. For example, a four-degree ROADM can route wavelengths between four directions (North, South, East and West) [**17**].

Nowadays, the main component that constitutes the ROADMs is the Wavelength Selective Switch (WSS) [**1**]. WSSs can, dynamically, route, block and attenuate all Dense Wavelength-Division Multiplexing (DWDM) signals within a network node [**17**]. Each DWDM signal at the 1:N input port is switched (routed) to any one of the N output ports, independently of how all the other wavelength channels are routed. This wavelength switching process can be dynamically changed through an electronic communication control interface in the WSS [**1**].

Today, the new generation of ROADMs includes the following properties: Colorless, Directionless, Contentionless and Gridless [**18**], in order to increase network flexibility. Next, each property is explained in more detail:

- *Colorless*: Any wavelength can be added/dropped from any add/drop port, i.e., a transponder port does not have a fixed wavelength.
- *Directionless*: Any wavelength added to a port can be directed to any direction and vice-versa, which means that the input signals are not directed only to one direction.
- *Contentionless*: Several signals of the same wavelength, i.e. color, can be in a single add/drop structure. When a ROADM does not support this property, contention can occur when wavelengths of the same color converge at the same WSS add/drop structure at the same time. This causes wavelength blocking [**17**].
- *Gridless*: The ROADMs filter shapes are adjustable by software typically using LCoS technology. ROADM nodes that support a flexible grid can operate at any mix of wavelength spacings, as long as each wavelength aligns with the 6.25 GHz frequency grid and the bandwidth assigned to each wavelength is a multiple of 12.5 GHz [**1**].

The most common ROADM architectures are Broadcast-and-Select architecture and Route-and-Select architecture.

A *Broadcast-and-Select architecture* is composed by passive splitters that perform the "broadcast" function at the ROADM input direction and by WSS that perform the "select" function at the output direction. When a WDM signal enters the splitter (power divider), the signal is sent to all outputs. This means that the inputs signals are replicated on all outputs WSS and drop ports. This architecture exhibits a disadvantage due to the high insertion losses in the splitter (in particular, for high degree ROADMs), since the signal input power will be divided by the number of outputs. And the more outputs there are, more power losses will occur. The Broadcast and Select architecture also supports multicast in the optical domain [**1**].

A *Route and Select architecture* consists of a stage of WSS 1xN at the ROADM input direction and a stage of WSS Nx1 at the output direction. The input WSSs have a "route" function and the output WSSs have a "select" function. An advantage of this architecture is the reduction of insertion losses, noise and crosstalk since optical splitters are not used like in the previous architecture [**1**]. This architecture does not support multicast.

2.5. Routing and Wavelength Assignment

In optical transport networks, the RWA are fully interconnected. The establishment of an optical path in a WDM transport network implies the selection of a set of links between the source and destination nodes (routing) and the reservation of a particular wavelength for each of these links that constitute the path (wavelength assignment) [**1**]. The problem

associated with this establishment is designated as RWA. This problem is more complex than the routing problem in networks that operate in the electrical domain, due to two important conditioning factors: the wavelength continuity and distinct wavelength.

Wavelength continuity means that, if there is no wavelength conversion in the network, the same wavelength must be assigned to all links that constitute the optical path. And, as the distinct wavelength indicates, two different optical paths should not have the same wavelength on the same link [**3**].

RWA can be seen as an optimisation problem whose goal is to minimise the number of wavelengths to be used in a given number of optical paths for a static network scenario. For a dynamic network scenario, the goal of RWA is to minimise the blocking probability, for a fixed number of wavelengths.

In this work, the RWA problem is studied for static networks and it is assumed that there are no wavelength conversions inside the network. As the optical connections are static, the same wavelengths must be assigned to all links in the lightpath and a lightpath that crosses the same links must be assigned to distinct wavelengths. RWA will be divided into two problems: first the routing problem, and then the wavelength assignment. For each one of these problems, heuristic solutions are used that will be discussed in the following subsections.

2.5.1. Routing

Routing is the process of selecting a path through the network for a particular traffic demand, where there are usually many possible paths from the source to the destination [**19**]. As such, routing is responsible for mapping the logical topology on the physical topology of the network [**3**].

The goal of routing is to find for each physical topology *G(V,E)* and for a given traffic matrix Q the set of paths that supports all traffic demands on this network [**1**]. Since there are many possible paths between a given source and a given destination on a network, it is important to define the most appropriate metric to apply to that network for routing. Among the various metrics that can be used in routing, the following stand out: minimising the number of hops of a path, minimising the distance of the path, maximising the OSNR (Optical Signal-To-Noise Ratio) of the path. A routing algorithm can apply several different metrics or just one metric [**1**].

Using routing algorithms, the configuration of a path can be done manually, through the management system (static routing) or through the control plane (dynamic routing). Static routing corresponds to the scenario in which the demand matrix does not change over the time, and dynamic routing corresponds to the scenario in which the demand matrix changes over time, with the arrival and termination of demands over time [**3**]. A real example of static routing application is in high-volume data transfer services between inter-datacentres on cloud platforms [**20**]. These services are significantly contributing to increasing network traffic, driving the improvement of static optical networks. These

high-volume data transfer services can be user data replication, application data synchronisation and virtual machine migration, which require long transmission times and various bandwidth resources [**20**]. These services where static routing is used are services where the begin and end of the operation are known in advanced.

2.5.1.1. Fixed-path routing

In fixed-path routing, the set of candidates paths is generated before any demand be added to the network [**1**]. It is a simpler approach where the chosen path is used to route all traffic between a given pair of nodes. Typically, the chosen path is the path that minimises distance. An example of this approach is fixed routing by shortest path [**3**]. The shortest path for each pair of nodes is calculated using shortest path algorithms, such as the Dijkstra algorithm or the *Yen's* k-shortest path algorithm. There are many other algorithms that use the shortest path routing [**19**], but in this work, we will only use these two.

The Dijkstra algorithm finds the shortest paths from one source node V to all other nodes on the network. In order to establish a shortest path from a particular node to all other nodes in the network, the Dijkstra algorithm uses the traffic matrix as the input and returns the shortest path from that node to a destination, as well as the total cost of that path [**1**].

The shortest path algorithm can be aggregated as part of a larger procedure to find the K-shortest paths (KSP). The goal of KSP algorithm is to find the first shortest path between two nodes, the second shortest path, the third shortest path, etc., until the shortest path k is found or until there are no more paths to calculate. The paths that are found may have common links and/or nodes. Among the many algorithms that exist based on KSP, the *Yen's* k-shortest path is one of the most used algorithms [**1**]. This algorithm uses the Dijkstra algorithm to calculate the shortest path between two nodes and then proceeds to find k-1 derivations of the shortest paths. The *Yen's* k-shortest path algorithm starts by finding the shortest path between two nodes, that being the principal path. Then, to find the other shorter paths between those same nodes, the various links of the main path are discarded as follows: to find the second shortest path, the first link from the main path is discarded; to find the third shortest path, the second link from the main path is discarded, and so on.

The fixed routing strategy may perform weakly as it usually leads to an unnecessary bottleneck in certain regions of the network. By always using the same path for a given pair of nodes, there is no change to adapt to the current status of the network. This can result in a premature blocking of a demand, although there are often better paths for that demand [**19**].

2.5.1.2. Alternative-path routing

In this type of routing, the set of candidate paths is also generated before any demand be added to the network. There are several M paths that can be chosen to route traffic between a determined pair of nodes (source node/destination node). One of the M paths is chosen when the demand request arrives, which makes the routing dependent on the state of the network [**19**]. This type of routing is fundamental to solving the problem of network bottleneck. Network bottleneck occurs when there are several physical links in a network that are overloaded, while other links are less loaded. In order to solve the problem of network bottleneck, the load balancing methodology is applied in order to optimise the use of network resources and avoid overloading the links [**1**]. Load balancing consists of using alternate paths in a network in order to reduce the maximum load on the most heavily loaded links.

Another situation in which it makes sense to apply alternative path routing is in a protection path link. A protection path is a path that cannot contain the links corresponding to the working path. The main path and the protection path have in common only the source node and the destination node, ensuring that a single failure does not affect both paths. Often, it is necessary to offer protection to a demand in order to improve its availability, where availability is defined as the probability of the demand being in a working state at a given point of time [**1**].

2.5.2. Survivability

Survivability in optical networks reflects the ability of a network to continue offering its services in the presence of internal failures. Failures can occur at the level of the network nodes or at the level of transmission links. In order to minimise these failures, two optical paths are usually defined. A working path that carries traffic during normal network operation, and the backup or protection path that is used when the working path fails. The paths must be disjoint, i.e. they do not use common links, but the source and destination nodes are the same [**1**]. Protection may be dedicated or shared [**2**].

In dedicated protection, spare resources are specifically allocated for a particular demand. If a demand is brought down by a failure, it is certain that there are available resources to recover the failure, assuming that the backup resources have not failed [**1**]. That is, for every working path there is a backup path. Dedicated protection can be divided into two categories: 1+1 and 1:1.

- 1+1 - the working signal is duplicated and sent simultaneously by the working path and the backup path. Upon arriving at the destination, the best of the two paths is selected by using, for example, the received signal power as a criterion.
- 1:1 - only after the failure has occurred in the working path, the backup path is activated to transport the signal.

Recovery from a failure is almost immediate when operating in 1+1 dedicated protection. In 1:1 dedicated protection, recovery from a failure is slower because first, it is

necessary to detect the failure (usually by destination node), using a signalling protocol (e.g. Automatic Protection Switching (APS)) and only then the backup path is activated [**1**].

In shared protection, a backup path is shared by several working paths. Working paths that share protection capability should not have links or intermediate nodes in common, so that a network failure does not break more than one path [**1**].

An example of a routing algorithm that has protection capability is the Suuraballe algorithm [**1**].

2.5.3. Ordering Strategies

In a static network scenario, after calculating the shortest path for all traffic demands using the routing algorithms mentioned in subsection 2.5.1.1 (Dijkstra and *Yen's* k-shortest path), it is necessary to order the demands according to a certain ordering strategy:

- *Shortest path first:* demands with the lowest cost (e.g. number of hops) in the path appear first in the list [**3**];
- *Longest path first:* demands with the highest cost (e.g. number of hops) in the path appear first in the list [**3**];
- *Random path:* demands are ordered randomly [**3**].

In a situation where there is a traffic demand with more than two paths with the same cost, the ordering strategy chooses the path that minimises the load on the most loaded link.

After ordering the demands according to the chosen ordering strategy, the demands are routed. In static networks, these ordering strategies are also used to assign wavelengths to the various path demands. In dynamic networks, demands arrive one at a time, so there is no need to order demands.

2.5.4. Wavelength Assignment

After routing the demands based on the routing algorithm most appropriate to the network, it is essential to assign the wavelengths to each lightpath.

Wavelength Assignment (WA) is an important part of the planning process of an optical network [**1**]. The problem of wavelength assignment for static networks consists in assigning a wavelength to each lightpath, given a set of lightpaths and their routes, so that none of these lightpaths share the same wavelength at a given fibre link and the same wavelength is used in all the links of a lightpath [**19**]. Moreover, the assignment of wavelengths to multiple path demands is performed at the same time, so the paths must be ordered first (see subsection 2.5.3).

In static networks, the main goal of the WA problem is to minimise the number of wavelengths to be used in a given network [**19**]. In order to correctly assign wavelengths to each path in a network, it is essential to study and implement an appropriate WA algorithm. The WA algorithms to be studied in this work will be: First-Fit, Most-Used and Graph Coloring heuristics.

The algorithms First-Fit and Most-Used are usually applied to dynamic networks [**19**]. In dynamic networks, instead of trying to minimise the number of wavelengths, we assume that the number of wavelengths is fixed and we try to minimise the blocking probability of the connection since the demands arrive one at a time. However, these two algorithms can also be applied to static networks. In this case, we have to order the lightpaths and then assign the wavelengths sequentially to the ordered lightpaths, according to the particular algorithm.

Next, a brief description of each algorithm mentioned above is presented.

2.5.4.1. First Fit

In the First-Fit algorithm, all available wavelengths are indexed with a number from 1 to W, where W is the maximum number of wavelengths supported in a fibre. Typically, this maximum number of wavelengths supported by a fibre is 96, for a 50 GHz frequency grid (corresponding to the C-Band) [**2**]. The indexes remain fixed through all the components. To index the wavelengths, the following orderings can be used: shortest-path first, longest-path first and random path. After ordering, the wavelength that has the lowest index is chosen. This means that the existing connections will be grouped at the lowest index wavelengths and successively.

2.5.4.2. Most-Used

Like First-Fit, Most-Used uses the same ordering strategies to index wavelengths. In this algorithm, there is a variable that stores the number of times that each wavelength is used per link. The wavelength that uses more links is given a higher priority [**1**], which means that this wavelength is assigned to the new path if the continuity of wavelength and distinct wavelength restrictions are fulfilled. If these conditions are not respected, the second wavelength with more links is tried, and so on.

2.5.4.3. Graph Coloring Heuristics

Another approach to solving the problem of wavelength assignment is to apply Graph Coloring techniques [**5**].

Graph Coloring is a conventional mathematical problem that consists in coloring all nodes of a graph so that there are no two adjacent nodes (adjacent nodes are nodes that share the same link) with the same color [**1**]. The procedure consists of:

(1) moving from the graph of the physical topology *G(V,E)* network to a graph *G(W,P)*, whose nodes (vertices) are the optical paths $W = (w_1, w_2, w_3, \ldots, w_M)$ and P is the set of links between these nodes [**3**]. Next, in this graph *G(W,P)*, a link is established between one or more nodes (e.g. paths) that share one or more physical links;
(2) coloring all the nodes of the graph *G(W,P)*, bearing in mind that different colors must be assigned to adjacent nodes.

The total number of distinct colors used to color a graph corresponds to the total number of wavelengths to be used in the network. For example, if in a given graph, five different colors are used to color the vertices, there are five different wavelengths in the network. That is, each color represents a different wavelength.

Any solution to the Graph Coloring problem corresponds to a valid solution to the wavelength assignment problem. Graph coloring is known to be a difficult problem to find an optimal solution, i.e. find the minimising number of colors [**1**]. However, there are bounds to the minimum number of colors required to color all nodes of the *G(W,P)* graph. This minimum number of colors required is designated as the chromatic number (Cr) of the graph. If the highest degree of a vertex in the graph to be colored is D, then it is possible to color the graph using a maximum of D+1 different colors, i.e. a maximum of D+1 different wavelengths is required for the wavelength assignment problem. This value is an upper bound [**1**], and can be written as,

$$Cr \leq D + 1 \tag{2.5}$$

To solve the problem of Graph Coloring, heuristic methods can also be used. Heuristic methods can give results near to optimal solutions or the optimal solutions [**5**]. In this work, we will study three heuristic methods. The heuristic methods/algorithms that are going to be studied are the Greedy, DSATUR and RLF [**5**].

The first of these algorithms to be studied is the *Greedy algorithm*. This algorithm is perhaps the most used for coloring graphs and the most useful for establishing bounds on the chromatic number. Greedy algorithm works by taking vertices of the graph one by one according to an arbitrary order [**5**], for example, placing the vertices in descending order of degree (the degree is the number of incident edges at each vertex). Then, the algorithm assigns each vertex to the first available color. Being a heuristic algorithm, it presents viable solutions quite quickly [**5**]. The example in Figure 2.9, shows how the algorithm works, which can be resumed in the following points:

(1) In order to start the assignment of colors to each vertex, it is necessary to analyse the graph to find out which node has the highest degree.
(2) After identifying the vertex with the highest degree, this vertex is assigned a color. Now, the graph already has a color assigned to a vertex.
(3) As the strategy chosen was to color the nodes in descending order of degree, the graph is analysed one more time to find out which vertex has the second highest degree of that graph.
(4) After identifying that vertex, a color must be assigned to it. If this vertex is adjacent to the vertex that has already been colored (point 2), then a different color is assigned to this vertex. If it is non-adjacent then the same color of the first vertex is used.

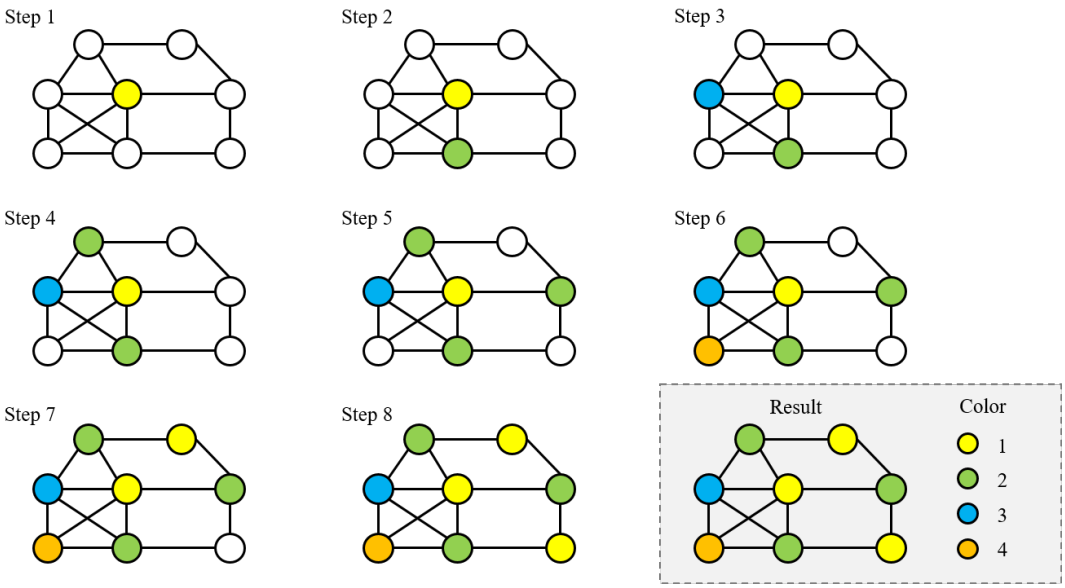

FIGURE 2.9. Example application of the Greedy algorithm.

(5) And the process is repeated again from point 1. Find the vertex that has the highest degree and assign it a different color from its adjacent vertices.

In analysing the example of Figure 2.9, it was identified that there is a vertex with 5 incident edges, so that vertex has degree 5, and it is precisely the highest degree of the graph. Therefore, it was assigned the first available color, the yellow color. Then the graph was analysed again in order to find a vertex that has the second highest degree (in this case, degree 4). Two vertices of degree 4 were identified. The choice of one or other is arbitrary. One of them was chosen and a color had to be assigned. Since the new vertex to be colored is adjacent to the first vertex, it can not use the yellow color, hence it is assigned the second available color, the green color. And this process continues until all the vertices of the graph are colored.

The second algorithm use to solve the problem of graph coloring is the *DSATUR algorithm*. This algorithm for coloring the vertices of the graph is based on the degree of saturation of each vertex. The degree of saturation of a vertex represents the number of different colors presented by the adjacent vertices of that vertex [**5**]. In this way, the degree of saturation is responsible for indicating the order in which the vertices are colored. The example in Figure 2.10 shows how the algorithm works, which can be resumed in the following points:

(1) The DSATUR algorithm begins by coloring the vertex with the highest degree, assigning it the first available color.
(2) After coloring this vertex, it is necessary to calculate the degrees of saturation of the other non-colored vertices belonging to the graph.
(3) The next step is to select the vertex with the highest degree of saturation (which was calculated in the previous step). If there is more than one vertex with the

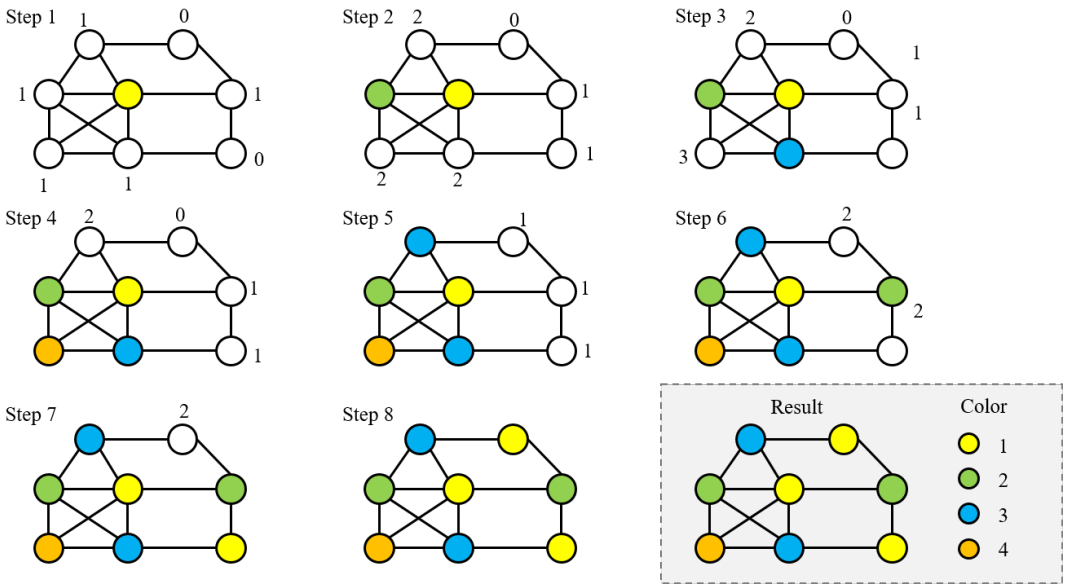

FIGURE 2.10. Example application of the DSATUR algorithm [**5**].

highest degree of saturation of the graph, then any vertex with the highest degree of saturation belonging to the non-colored subgraph is chosen.

(4) After choosing that vertex, another color is assigned. And, it is also important to know if this vertex is adjacent to the vertex that is already colored.

(5) And the process continues from point 2. Compute the degree of saturation of the vertices of the graph that are still to be colored and select the one with the highest degree of saturation.

According to the example given in Figure 2.10, it can be verified that the node with the highest degree is colored with the first available color, the yellow color. Then, the degree of saturation of the remaining vertices is computed. The next step was to select the vertex with the highest degree of saturation. When observing the step 1 of the example, it was noted that there are several vertices with the highest degree of saturation so far (degree of saturation 1), and it becomes necessary to choose from these vertices of degree of saturation 1, the vertex with the highest degree. When it is chosen, the vertex is assigned the green color, since it is adjacent to the first vertex already colored. And this cycle ends when all the vertices of the graph have been colored.

In the DSATUR algorithm, the choice of the following vertex to be colored is decided heuristically based on the characteristics of the current partial coloring of the graph. DSATUR algorithm turns out to be more accurate than the Greedy algorithm for various elementary graph topologies, such for example the bipartite graph [**5**].

The last algorithm to be studied in this work is the *RLF algorithm*. This algorithm follows a slightly different strategy regarding the graph coloring in comparison with the previous algorithms (Greedy and DSATUR algorithms), which have a similar procedure [**5**]. The RLF algorithm consists of coloring a graph with one color at a time, as opposed to one vertex at a time. At each step, the algorithm applies heuristic methods to identify

an independent set of vertices in the graph, which are then, associated with the same color. This independent set of vertices is then removed from the graph, and the process is repeated in the resultant smaller subgraph. The example in Figure 2.11 shows how the algorithm works, which can be resumed in the following points:

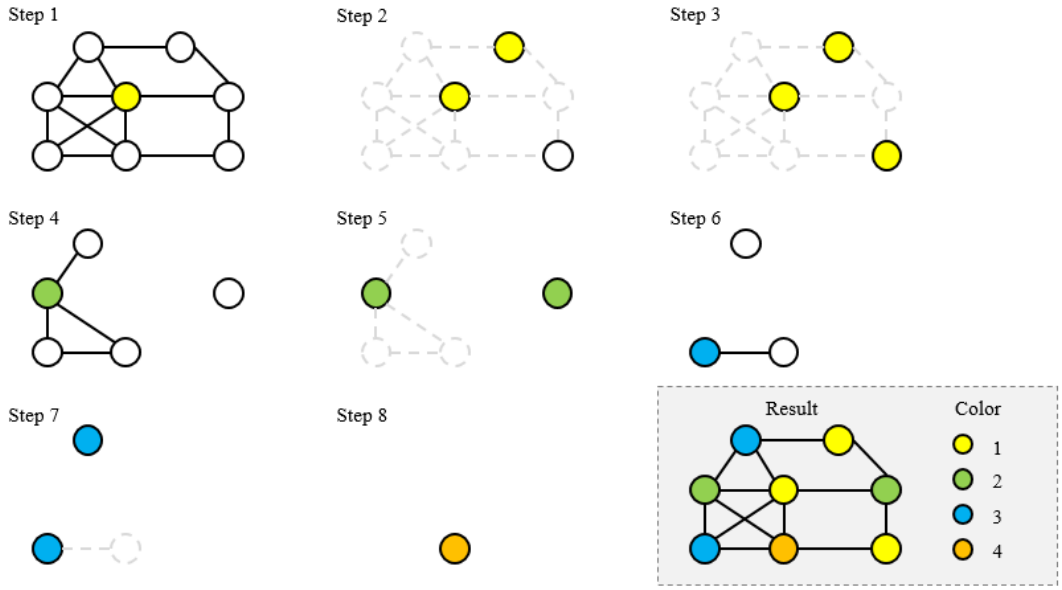

FIGURE 2.11. Example application of the RLF algorithm [**5**].

(1) The RLF algorithm begins by choosing the vertex with the highest degree, assigning it the first available color.
(2) Then, all the vertices adjacent to that vertex are moved to a subgraph, in order to identify which are the independent vertices (the vertices that have no direct link to the highest degree vertex).
(3) When identifying the independent vertices to the vertex with the highest degree, those vertices are assigned with the same color used to color the highest degree vertex.
(4) After being colored, these vertices are removed from the graph, and the subgraph with the remaining vertices appears.
(5) The procedure is repeated again from point 1. The vertex with the highest degree in the subgraph is selected by assigning it the second available color, the independent vertices are identified and assigned with the same color and then removed from the graph.

Looking at the example in Figure 2.11, we see that the node with the highest degree of the graph is colored with the yellow color. Then, the independent vertices are identified and are also colored with the yellow color. Then, these yellow vertices are removed from the graph, giving rise to a subgraph with the remaining vertices that are still to be colored. In this subgraph, the vertex with the highest degree is selected and it is colored with green color because the yellow color has been already used. And, the process continues until

all the vertices belonging to the graph are colored. This process ends when there are no more possible subgraphs, which means that all vertices have been colored, resulting in a possible solution [**5**].

The same simple network (Figures 2.9, 2.10 and 2.11) was used to explain how each one of the Graph Coloring heuristics work. As observed, 4 colors were used, i.e. 4 wavelengths were assigned in each one of the algorithms. But, the fact that these three algorithms present the same number of colors is because in the example we are using a small network.

Usually, for graphs with a higher number of nodes, the Greedy algorithm presents solutions with a high number of colors, despite requiring low computation times. In contrast, the RLF algorithm tends to produce solutions that require a lower number of colors, at a cost of longer computation times, especially for graphs that have a high number of vertices [**5**]. The solutions produced by the DSATUR algorithm are in the middle of the other two algorithms.

2.6. Conclusion

In this chapter, the basic concepts related to optical networks have been introduced. First, the concepts of physical topology and logical topology of a network are defined, alongside with the concept of the adjacency matrix and traffic matrix. Next, a brief explanation of how OTN architecture transports information in optical networks was provided. The network elements that are used in today optical networks, the ROADMs, were also briefly explained. Finally, in this chapter, the main goal of this work was introduced, which is the planning of an optical network, that involves the routing and the wavelength assignment. Routing algorithm (*Yen's* K-shortest path) and wavelength assignment algorithms (First Fit, Most Used and Graph Coloring) were identified. All these algorithms will be implemented in this work. The Graph Coloring heuristics include three algorithms: Greedy, DSATUR and RLF. Detailed explanations, as well as the use of examples for a better understanding, of the algorithms, were given. The goal of studying these three algorithms is to quantify their performance and conclude which is the algorithm that leads to the lowest number of wavelengths for a specific network physical topology with a particular traffic demand, which will be analysed in Chapters 3 and 4.

CHAPTER 3

RWA Planning Tool Based on Graph Coloring Heuristics

3.1. Introduction

This chapter presents the RWA planning tool for optical networks developed in this work using MatLab. In section 3.2, the flowchart that details the RWA tool developed is shown. First, the *Yen's* k-shortest path routing algorithm was implemented and, then, the wavelength assignment algorithms deployed in this work were: First Fit, Most Used and three Graph Coloring heuristics (Greedy, DSATUR and RLF). Since Graph Coloring heuristics are the main purpose of this work, section 3.3 presents the pseudocodes of the Greedy, DSATUR and RLF algorithms that served as the starting point to implement them in the tool. Finally, in section 3.4, a performance study of these Graph Coloring algorithms is carried out as a function of the number of vertices and their degree, using random graphs generation.

3.2. RWA Tool

In this section, the RWA problem based on heuristic solutions for static networks will be described in detail. As explained in the previous chapter, the *Yen's* k-shortest path algorithm will be used to perform the routing. After network demands are routed, the wavelength assignment algorithm is applied. In a first phase, the First Fit and Most Used algorithms are used to perform the WA because of their simplicity and efficiency. These two algorithms are usually used in dynamic networks, but they can be also applied to static networks as long as the lightpaths are firstly ordered. Then, the wavelengths can be assigned to the ordered lightpaths. In a second phase, the wavelengths are assigned to each lightpath using Graph Coloring algorithms. It should be noted that a wavelength is only assigned to a lightpath if it respects the two WA constraints: continuity of wavelength and distinct wavelength.

The flowchart in Figure 3.1 shows how the RWA problem is implemented in this work by its decomposition in 6 steps:

(1) *Physical Topology and Traffic Matrix*: in this step, the network physical topology (defined by the adjacency matrix), as well as, the logical topology (defined by the traffic matrix) are defined.

(2) *Lightpath Routing*: the lightpaths are routed over a physical topology using the *Yen's* k-shortest path algorithm that returns the shortest path of all paths of each pair of nodes accordingly to a specific metric.

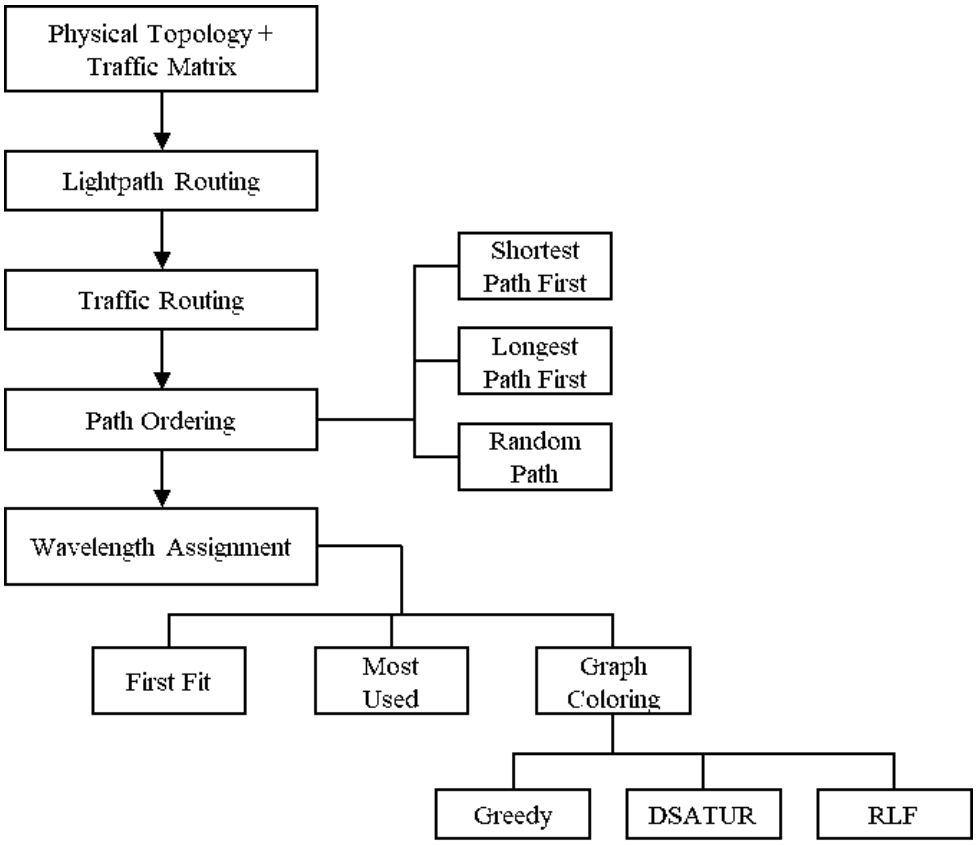

FIGURE 3.1. RWA sub-problems.

(3) *Traffic Routing*: after the route is computed and selected, the different traffic units are routed and assigned between the source and destination nodes through the logical topology.

(4) *Path Ordering*: before assigning the wavelengths to the lightpaths obtained through the routing algorithm, it is necessary to order these lightpaths. The criteria used to order the paths can be shortest path first (sort the paths in ascending order), longest path first (sort the paths in descending order), and random path (sort the paths randomly). Different metrics can be used to establish the paths order, for example, the distance in kilometers or the number of hops can be used.

(5) *Wavelength Assignment*: after the lightpaths are ordered, the wavelengths are assigned accordingly to a given algorithm. Graph Coloring heuristics such as Greedy, DSATUR and RLF, are going to be used in the RWA planning tool. For these heuristics, step 4 is not required.

In other words, in a RWA situation, after getting the list of lightpaths through the routing algorithm, one of the Graph Coloring algorithms can be used to assign the wavelengths to the network. Since Graph Coloring algorithms transform a graph *G(V,E)* that represents the physical network topology into a graph *G(W,P)*, whose vertices are the lightpaths $W = (w_1, w_2, w_3, \ldots, w_N)$ and P is the set of links between these vertices, these algorithms only need the list of the lightpaths routed independently of its order.

This means that the lightpaths only need to be ordered (step 4), when using wavelength assignment algorithms like First Fit and Most Used.

In step 2, the *Yen's* k-shortest path routing algorithm is used. This algorithm belongs to the KSP family. The *Yen's* k-shortest path consists in calculating the shortest path between two nodes and then proceeds to find the k-1 derivations of the shortest paths. The first shortest path found between two nodes is the main path. The function of MatLab responsible for finding this main path is designated by *shortestpath*. The *shortestpath* function uses the Dijkstra algorithm in its implementation. Then, to find the other shortest paths between this pair of nodes, the algorithm proceeds by eliminating one of the links in the main path for each shortest path found. In order to fulfill the requirements of the *Yen's* k-shortest path algorithm, the following function representing the algorithm was implemented in this work: $[PATH\ ,\ COST] = kshortestpaths\ (G,\ S,\ T,\ K)$.

This function determines the K shortest paths from the source node (S) to the destination node (T). The weights of the links are all positive entries in the N-by-N adjacency matrix represented by the sparse matrix G. The $COST$ matrix compiles the K distances from source to destination (from S to T) and the $PATH$ matrix is a cell array with the K shortest paths. In our work, we only use $k = 1$ and, so, the $PATH$ cell array returns a list of the first shortest path between each pair of nodes.

Once the routing of network demands has been completed and the list of lightpaths has been obtained, it is already possible to assign the wavelengths to the network. As in this work, only static networks are considered, the goal of the WA algorithms is to minimise the number of wavelengths used in each network.

One of the WA algorithms studied is the First Fit. In the First Fit algorithm, all available wavelengths are indexed. The algorithm runs through the ordered list of lightpaths (step 4) obtained through routing, and it assigns lower index wavelengths from the available wavelengths.

Similarly to the First Fit algorithm, the Most Used algorithm also runs through the list of lightpaths ordered based on one of the three existing ordering strategies. After fulfilling this step, the algorithm contains a variable that stores the number of times each wavelength is used per link. To the wavelength that is assigned to more links is given higher priority, and assigned to the new lightpath, if it meets the WA restrictions. If this wavelength does not comply with these restrictions, the second wavelength with the highest priority is assigned and so on.

3.3. Graph Coloring Heuristics Implementation

Based on the analysis of Figure 3.1, the tool developed in this work, besides contemplating the First Fit and Most Used algorithms for wavelength assignment, also includes Graph Coloring heuristics to accomplish the same goal.

The Graph Coloring algorithms studied and implemented in this work are: the Greedy algorithm, the DSATUR algorithm and the RLF algorithm [**5**]. These algorithms aim to

color all the nodes of a graph so that there are no two adjacent nodes with the same color. Adjacent nodes are defined as nodes that share the same link.

In the next subsections, the pseudocodes of these algorithms are presented, which served as a starting point to implement them in the tool. The notation that is common to the three pseudocodes is described in Table 3.1.

TABLE 3.1. Notation used in the pseudocodes.

Notation	Description
$\|X\|$	Length of vertex set X
$\leftarrow$	Assignment operator
$\cup$	Union operator
independent set	Subset of vertices mutually non adjacent
X	Set of all vertices sorted in an arbitrary order, not assigned to a color
V	Set of all vertices sorted in descending order of degree
S	Set of all colors
S_j	Color j that will be assigned to a vertex of X
v_i or v	Vertex of X selected in each iteration of the algorithm

Note: the notation v_i is used in the pseudocode of the Greedy algorithm and the notation v is applied in the pseudocodes of the DSATUR and RLF algorithms.

3.3.1. Greedy Algorithm

As explained in the previous chapter, the Greedy algorithm consists in coloring the vertices of the graph one by one according to an arbitrary order. The Greedy algorithm developed for this planning tool is implemented from the pseudocode of Figure 3.2. The notation used in this pseudocode 3.2 is described in Table 3.1 and follows the one proposed in [**5**] with some modifications.

```
GREEDY (S ← S1; X ←V)
(1)  for i ← 1 to |X| do
(2)      for j ← 1 to |S|
(3)          if (Sj ∪ {vi}) is an independent set then
(4)              Sj ← Sj ∪ {vi}
(5)              break
(6)          else j ← j +1
(7)          if  j > |S|  then
(8)              Sj ← {vi}
(9)              S ← S ∪ Sj
```

FIGURE 3.2. Pseudocode of Greedy algorithm.

The pseudocode starts by assigning the vertices of set V to set X, i.e. X = V, since set X represents the set of all vertices not initially assigned to a color and set V represents the

vertices ordered in descending order of degree. The pseudocode also defines the set S which represents the list of colors associated with the vertices of X. Initially S = S_1, because there is already the first available color to be assigned to a vertex. As the algorithm takes a vertex from the set X, it is assigned a color from the set S.

In each iteration, the algorithm takes the vertex v_i of the set X, and checks if v_i can be assigned the color S_j, that is, it checks if the color S_j was not assigned to an adjacent vertex of v_i (independent set). If the condition is confirmed, then the color S_j is assigned to the vertex v_i and the process moves on to consider the next vertex. Otherwise, the algorithm assigns a new color to the vertex v_i. Lines 7, 8 and 9 of the pseudocode represent the creation of a new color and the assignment of that color to vertex v_i, as well as its addition to the set S. The algorithm ends when all vertices of set X have been assigned a color of the set S.

An example of the algorithm on a small graph is shown in Figure 3.3 (same as Figure 2.9), but repeated here for clearness of the pseudocode of Figure 3.2.

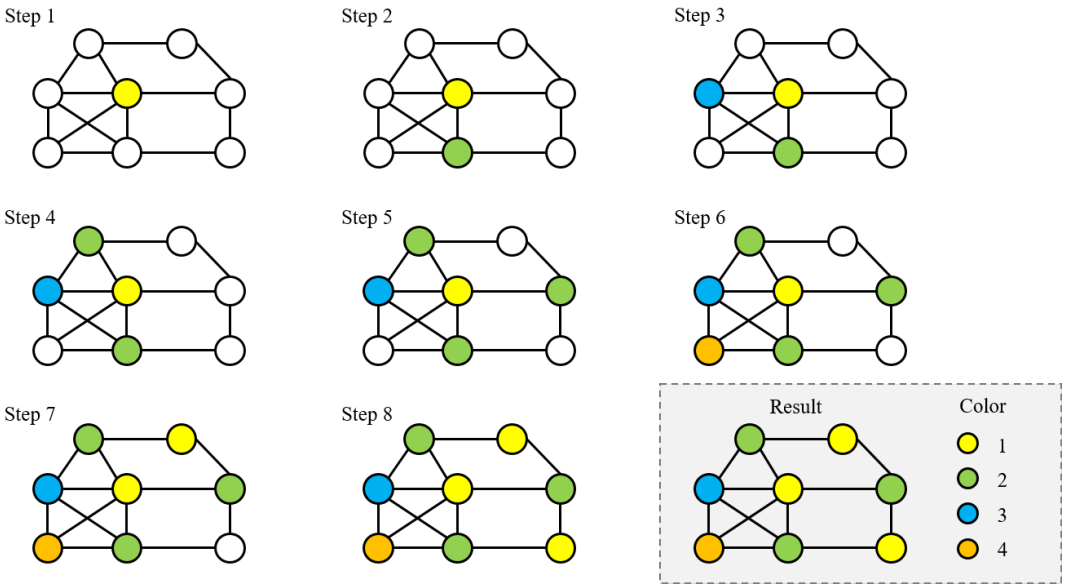

FIGURE 3.3. Example application of the Greedy algorithm on a simple network.

Considering that the vertices are ordered in descending order of degree, the first vertex to be colored will be the vertex with the highest degree. Therefore, in step 1 of Figure 3.3, the vertex with the highest degree is selected. Since the set S already has the first color available (the yellow color), this is directly assigned to the vertex with the highest degree, once that the condition of line 3 of the pseudocode is verified. In other words, since this vertex is the first selected vertex and there are still no adjacent colored vertices, the algorithm directly assigns it the yellow color, as shown in step 1 of the example. Once the vertex is already colored with the yellow color, the process considers the next vertex. The algorithm checks again which vertex has the second highest degree that exists in the set X, and identifies two vertices with the same degree. The criteria for choosing a vertex, if there is more than one vertex with the same degree, is according to the position

in which they appear in the set X. In this way, a vertex is selected, as shown in step 2 of Figure 3.3, and checked whether or not this vertex is adjacent to the vertex that was colored in step 1. Once they are adjacent, it moves directly to line 7, creating the green color, since adjacent vertices cannot have the same color. Now, the green color is assigned to that vertex and is added to the set S. If these two vertices were not adjacent, then the algorithm would assign the yellow color to the vertex (line 4). The algorithm ends until all vertices of the graph are colored. In this example, 4 colors were used, i.e. 4 wavelengths were needed.

3.3.2. DSATUR Algorithm

The DSATUR algorithm is very similar to the Greedy algorithm since it takes each vertex one by one according to some ordering and then assigns the appropriate color to the vertex. The difference between these two algorithms is on the ordering in which the vertices are generated. In the Greedy algorithm, the vertex ordering is decided before any coloring, however, in the DSATUR algorithm, the choice of the next vertex to be colored is decided heuristically based on the characteristics of the current coloring of the graph. In other words, in the DSATUR algorithm, the choice of the next vertex to be colored is based primarily on the saturation degree of the vertices. The degree of saturation of an uncolored vertex is the number of different colors presented by its adjacent vertices.

The DSATUR algorithm developed in this work is implemented from the pseudocode of Figure 3.4 using the description of the notation described in Table 3.1. Through the analysis of the pseudocode and as previously mentioned, it can be observed that this algorithm is quite similar to the Greedy algorithm, in that it is selected one vertex at a time according to a certain order and then assigned to a color. The main difference between these two algorithms lies in lines 2 and 11 of the pseudocode.

This pseudocode starts by defining the initial sets, X and S, in the same way, they are defined in the pseudocode of the Greedy algorithm. In each iteration of the algorithm, the next vertex to be colored is selected from the set X (line 2) and a color is assigned to that vertex of the set S, similar to the Greedy algorithm. Once the vertex is colored, this is removed from the set X (line 11).

The algorithm ends when X=0 which means that all vertices have a color of set S assigned. It is on line 2 that the next vertex to be colored is chosen as the vertex in X that has the maximal saturation degree. If there is more than one vertex with the maximal saturation degree, the one with the highest degree is chosen from these set of vertices. The first vertex chosen to be colored is the vertex with the highest degree.

Figure 3.5 shows an example of the algorithm on a small graph with 8 vertices.

To begin, all vertices have saturation degree of zero, since there are no colored vertices yet. In this way, in step 1 of Figure 3.5, the vertex with the highest degree is selected. Since the set S already has the first color available (the yellow color), this is directly assigned to the vertex with the highest degree, once that the condition of line 4 of the pseudocode is verified. Once the vertex is already colored, it is removed from the set X.

```
DSATUR (S ← S1; X ←V)
(1)  for i ← 1 to |X| do
(2)      Choose v ∈ X
(3)      for j ← 1 to |S|
(4)          if (Sj ∪ {v}) is an independent set then
(5)              Sj ← Sj ∪ {v}
(6)              break
(7)          else j ← j +1
(8)          if  j > |S|  then
(9)              Sj ← {v}
(10)             S ← S ∪ Sj
(11)     X ← X - {v}
```

FIGURE 3.4. Pseudocode of DSATUR algorithm.

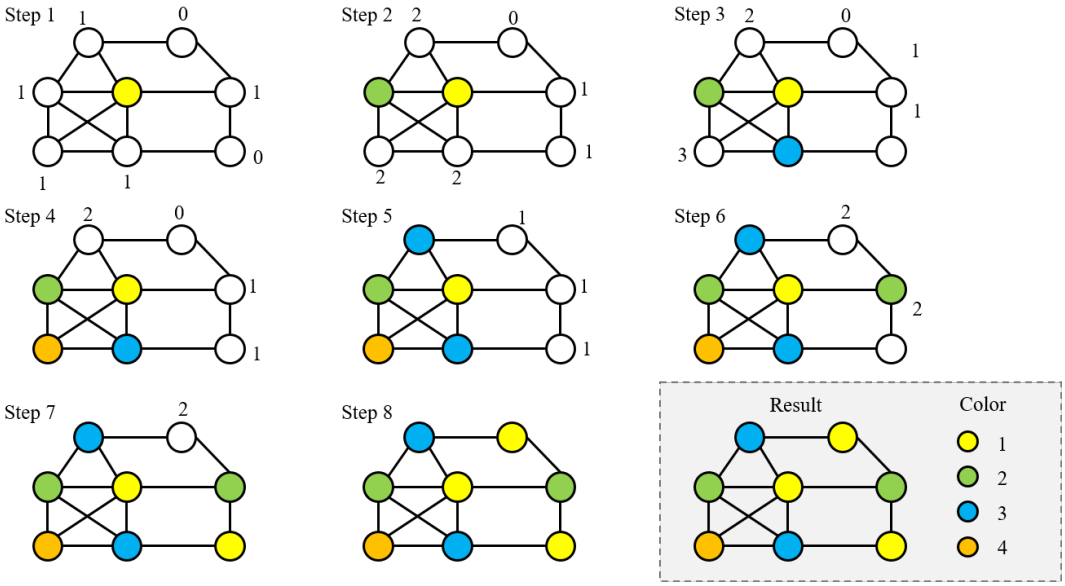

FIGURE 3.5. Example application of the DSATUR algorithm [**5**] on a simple network.

After removal, the algorithm recalculates the saturation degrees of the remaining vertices of the set. After this calculation, the algorithm checks if there is more than one vertex with the maximum degree of saturation so far (degree 1) and selects the one with the highest degree, as shown in step 2. As this vertex of step 2 is adjacent to the previous vertex (vertex with the yellow color), the pseudocode moves to line 7, where it the green color is assigned to this vertex and, then, this vertex is removed from the set X. Now the algorithm recalculates the saturation degree of remaining vertices of the set X. The cycle ends when all vertices of the graph are colored. Similarly, to what happened in the

Greedy algorithm example, in the example of Figure 3.5, 4 colors have been attributed to color the graph.

3.3.3. RLF Algorithm

The last algorithm to be implemented in the RWA tool is the RLF algorithm. As mentioned in the previous chapter, this algorithm follows a slightly different strategy regarding the graph coloring in comparison with the Greedy and DSATUR algorithms. The RLF algorithm consists in coloring a graph with one color at a time as opposed to one vertex at a time. As shown in the pseudocode of Figure 3.6, the algorithm applies heuristic methods to identify an independent set of vertices in the graph (non-adjacent vertices), which are then, associated with the same color. The notation used in the pseudocode is the same as the one used for the Greedy and DSATUR heuristics. Additionally to the notation of Table 3.1, in this pseudocode a new symbol is introduced, $\Gamma_X\{v\}$, which represents the set of adjacent vertices of v.

RLF $(S \leftarrow \emptyset\ ;\ X \leftarrow V;\ Y \leftarrow \emptyset;\ Z \leftarrow \emptyset)$

```
(1)  for i ← 1 to |X| do
(2)      for j ← 1 to |X| do
(3)          while j ≠ ∅ do
(4)              Choose v ∈ X
(5)              Z ← {v}
(6)              Y ← Y ∪ Γ_X{v}
(7)              X ← X − (Y ∪ {v})
(8)      X ← Y
(9)      Y ← ∅
(10)     for k ← |S| to |Z| do
(11)         S_k ← S_k ∪ {v}
(12)         S ← S ∪ S_k
(13)     k ← k + 1
```

FIGURE 3.6. Pseudocode of RLF algorithm.

The pseudocode begins by defining four sets. The set X that contains the initially uncolored vertices, the set Z that will aggregate throughout the algorithm the selected vertices in each cycle to be later assigned to a color of the set S, the set S that is responsible for assigning a S_k color to the vertices of Z and the set Y that will contain the uncolored vertices that cannot be feasible assigned to S_k color. At the beginning of the pseudocode execution X=V, Z=∅, S= ∅ and Y= ∅.

In each outer loop a color is created. Lines 2-7 are responsible for selecting the vertex to be colored. To start, a vertex v of X is selected and added to the set Z. All vertices adjacent to v ($\Gamma_X v$) are then transferred to the set Y, since they cannot be assigned the same color as v. Finally, on line 7, both the vertex v and its adjacent vertices are removed from the set X, since they are not considered candidates for assignment of the same color as v. As soon as X=0, no more vertices can be added to the current color. Therefore, line 8 of the algorithm represents all the vertices of the set of non-colored vertices Y to be moved to the set X, and and then in line 9, Y is emptied.

Lines 10-12 represent the assignment of the color selected by the outer loop, S_k, to all vertices that are in set Z and have no color assigned. As soon as this color is assigned to these vertices, the color is added to set S. Then a new color (line 13) is created and the algorithm repeats the loop again. The algorithm ends when both sets X and Y are empty, which means that all vertices have been colored and are in set S.

The process of selecting the next $v \in X$ on line 4, follows a similar rationale to the DSATUR algorithm. The first vertex to be chosen for the assignment of each color is the vertex in X that has the highest degree. The remaining vertices v to be assigned to the same color are selected as the vertices in X that have the highest degree in the subgraph induced by $Y \cup \{v\}$.

Figure 3.7 shows an example of the RLF algorithm step-by-step.

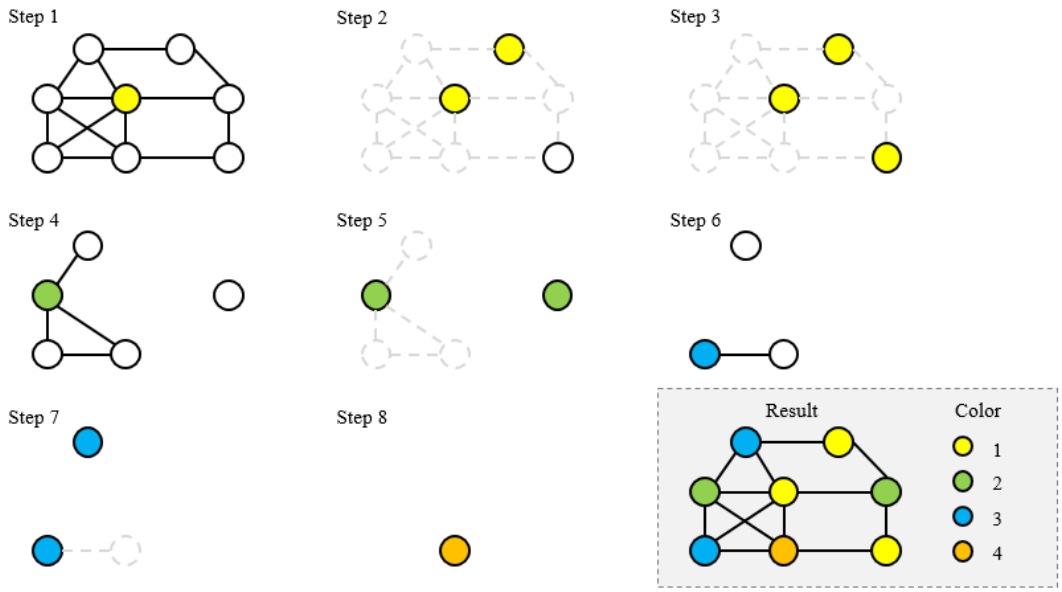

FIGURE 3.7. Example application of the RLF algorithm [**5**] on a simple network.

In step 1 of Figure 3.7, the yellow color is created and the vertex v with the highest degree is selected and added to the set Z as shown in line 5 of the pseudocode. In step 2, all vertices adjacent to v were moved to set Y (line 6 of the pseudocode), identifying the remaining vertices of set X that are not adjacent to vertex v (line 7 of the pseudocode). These two vertices are also added to the set Z. At the same time, that the vertices already

in the set Z (the vertex v and the vertices not adjacent to it) are assigned to the yellow color, the vertices that are in set Y are transferred to the set X, as shown in line 8 of the pseudocode. In step 4, the green color is created, and the process is repeated in the subgraph induced by the remaining uncolored vertices. The process ends when all vertices are assigned to a color. With the application of the RLF algorithm, 4 colors were used to color this graph.

3.4. Graph Coloring Heuristics Validation and Performance

After implementing the Greedy, DSATUR and RLF algorithms in the planning tool developed in Matlab, the performance and validation of these algorithms must be discussed. In this section, we study the behavior of these heuristics as a function of the number of vertices and their degree. Note that the number of vertices of $G(W, P)$ correspond to the number of paths. Hence, this study was applied to graphs $G(W, P)$ with 20 vertices, 50 vertices and 100 vertices.

To perform this study we use the random graph concept. A random graph, denoted by $G_{n,p}$, is a graph comprising n vertices in which pair of vertices is adjacent with probability p [**5**].

In the tool developed in this work, the parameter p of graph $G_{n,p}$ is obtained through the total average of the degree of each vertex. The expression used is the following:

$$p = \frac{\sum_{i=1}^{n} \frac{degree_i}{n-1}}{n} \tag{3.1}$$

where $degree_i$ represents the degree of the vertex i.

In our tool, it is possible to generate random path matrices according to the desired p probability. For example, if a $p = 0.5$ is intended, pairs of adjacent vertices are randomly generated in the path matrix, so that the final average of the degrees of that matrix is 0.5. When the probability p is equal to 1, it means that each vertex of the path matrix has connections to all vertices except himself.

Figures 3.8, 3.9 and 3.10 show the average number of colors as a function of p obtained, respectively, for $n = 20$, 50 and 100, for the Greedy, DSATUR and RLF algorithms. The upper bound given by (2.5) is also shown. For each value of p, 50 simulations were run. Then, an average is performed to obtain the average number of colors after these 50 simulations. In the Greedy algorithm, the vertices are ordered considering the random order of degree, so that the results can be compared with Figure 2.14 of [**5**].

The curves in Figures 3.8, 3.9 and 3.10 indicate that the three algorithms have a similar behaviour for all tested values of n. Observing with more detail the obtained curves, we verify that for probabilities close to 1, the number of colors predicted by the different algorithms tends to become more approximated, reaching the maximum limit of colors that each n can achieve. In general, the Greedy algorithm is the algorithm that produces the worst results, i.e., it needs to assign more colors, while the RLF algorithm

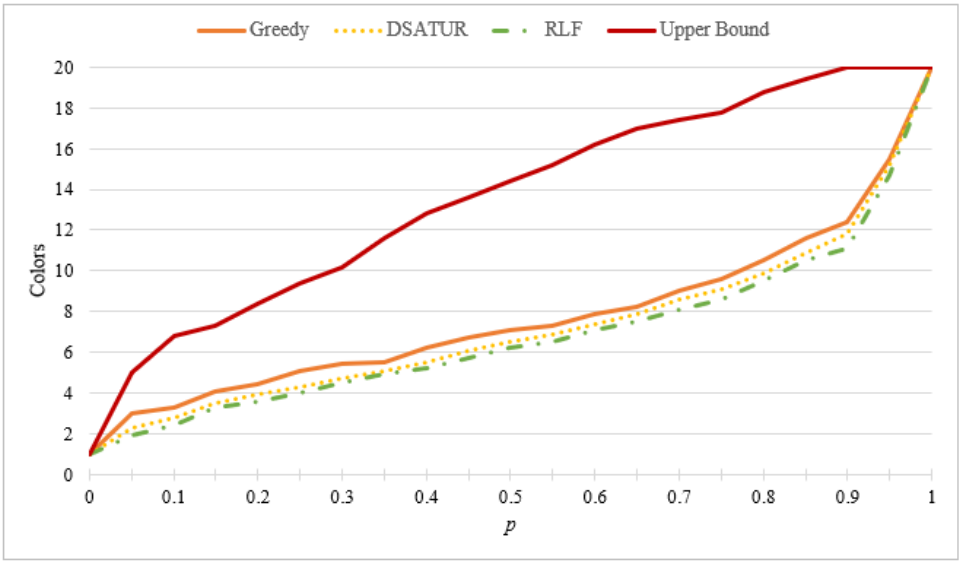

FIGURE 3.8. Average number of colors produced by the Greedy, DSATUR and RLF algorithms for a network with n=20.

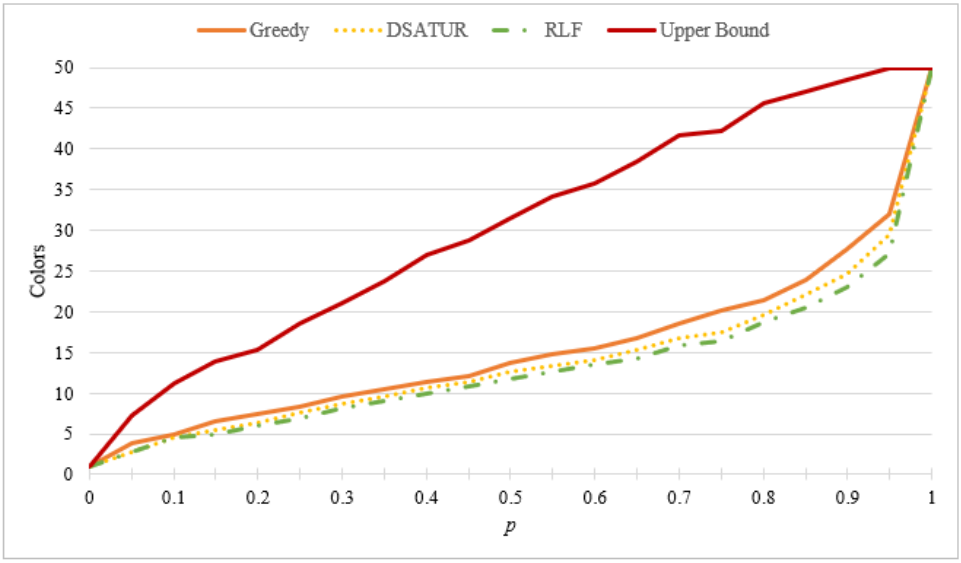

FIGURE 3.9. Average number of colors produced by the Greedy, DSATUR and RLF algorithms for a network with n=50.

generates the best solutions across the whole set, assigning a lower number of colors. The DSATUR algorithm produces solutions with fewer colors than the Greedy algorithm and slightly more colors than the RLF algorithm. The obtained results also show that the upper bounds predictions are very far from the ones obtained with the three Graph Coloring algorithms.

Analysing in more detail the solutions produced by the graph $G(W, P)$ with 100 vertices (Figure 3.10), for probabilities p lower than 0.2, the Greedy algorithm achieves about 10.8 colors, the DSATUR algorithm produces approximately 8.7 colors and the RLF algorithm generates 8.04 colors. For a probability p=0.5, the Greedy algorithm produces

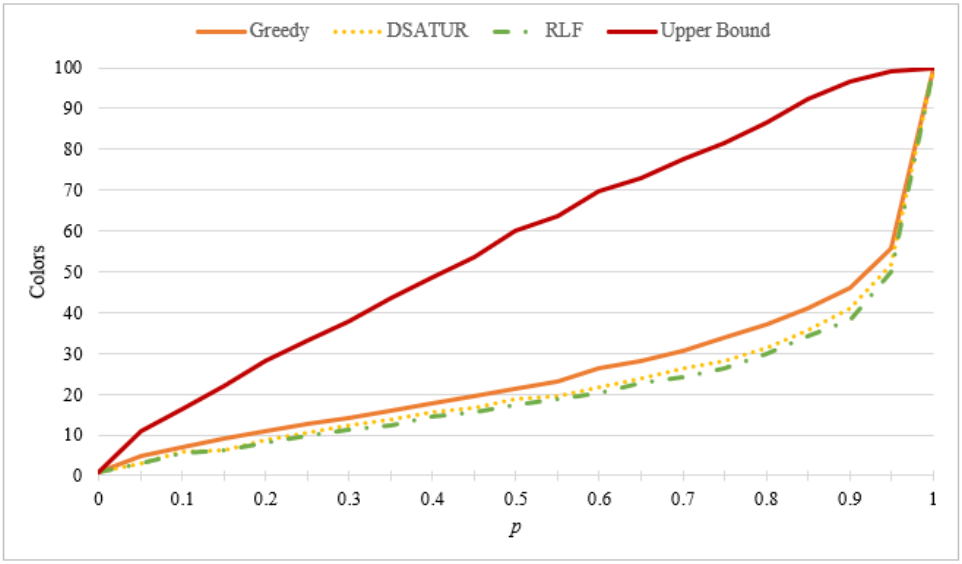

FIGURE 3.10. Average number of colors produced by the Greedy, DSATUR and RLF algorithms for a network with n=100.

around 21.3 colors, the DSATUR algorithm generates about 18.7 colors and the RLF algorithm generates only 17.5 colors. For this value of p, the upper bound on the number of colours is as great as 60 (almost the triple obtained with the analysed algorithms). These results are summarised in Table 3.2, which compares the results obtained in this work with the values in table 2.1 of [**5**], for a graph in the same conditions (n=100 vertices and p=0.5). We confirm that the values of the two simulations are quite similar, presenting only slight differences (< 0.2 colors). In this way, it can be concluded that the algorithms developed in this work have the expected performance and so their implementation can be considered validated.

TABLE 3.2. Comparison of the solutions obtained in this work with the solutions in [**5**] for a network with n=100 vertices and p=0.5.

	Greedy algorithm	**DSATUR algorithm**	**RLF algorithm**
This work	21.31	18.66	17.52
R.M.R Lewis [5]	21.14	18.48	17.44

Furthermore, the computational time in seconds that each of these algorithms required to obtain the results shown in Figures 3.8 - 3.10 has been also analysed. The Greedy algorithm despite producing more colors, generates solutions more quickly, i.e., it obtains a shorter computational time compared to the other algorithms. The RLF algorithm requires more time to generate solutions, but it is the algorithm that assigns fewer colors to the graphs. The DSATUR algorithm has slightly longer computational times than the Greedy algorithm. Table 3.3 presents the computation times that each one of these algorithms requires to produce a possible solution for n = 20, 50 and 100. All results presented in this work were obtained on a computer with an intel core i5 processor of 2.20 GHz frequency and 8 GB of RAM.

TABLE 3.3. Computational time to obtain one solution, for each one of the algorithms considered, for p=0.5.

n	Greedy algorithm	DSATUR algorithm	RLF algorithm
20	3.25 s	3.74 s	4.20 s
50	3.65 s	3.56 s	4.44 s
100	3.93 s	4.63 s	6.22 s

After assessing and validating the performance of the Greedy, DSATUR and RLF algorithms with different numbers of vertices, next, we will study the performance of only the Greedy algorithm for different types of vertex sorting, such as: random order of degree, ascending order of degree and descending order of degree.

As in the previous study, this study is performed for the same graphs $G_{n,p}$ considered in Figures 3.8 - 3.10, which are generated randomly for each p. Figures 3.11, 3.12 and 3.13 show the number of colors assigned by the Greedy algorithm with the three scenarios of vertex ordering, for $n = 20$, $n = 50$ and $n = 100$, respectively. Each of the curves shown for each ordering is obtained by averaging the number of colors obtained after 50 simulations.

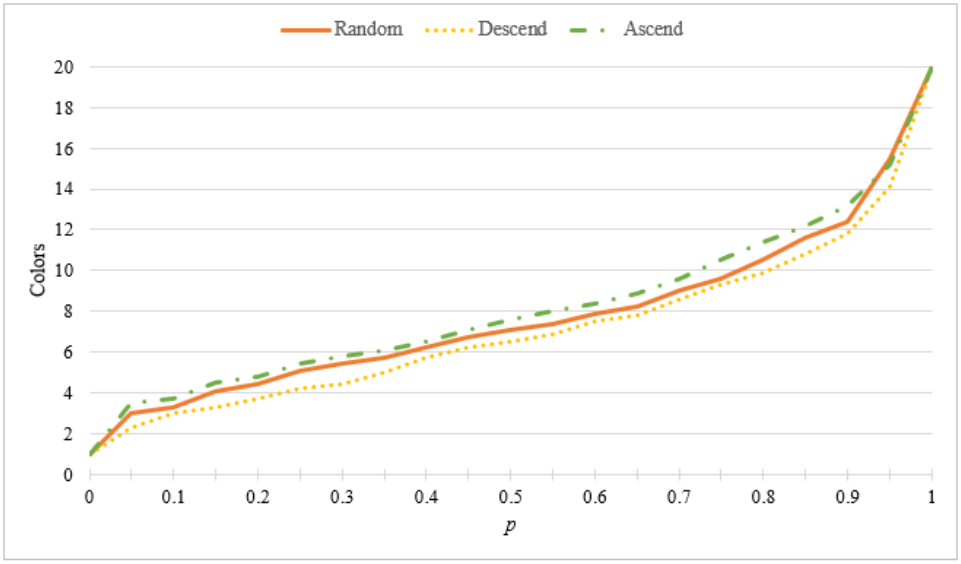

FIGURE 3.11. Average number of colors produced by the different orderings of the Greedy algorithm for a network with n=20.

By analysing the curves in Figures, 3.11, 3.12 and 3.13, we conclude that the results generated by these vertices ordering of the Greedy algorithm for graphs $G_{n,p}$ with different values of n, are quite similar. For all tested values of n, it is observed that the random ordering of degree presents a fewer number of colors than the ascending order and slightly more colors than the descending order. In turn, the descending order of degree produces a lower number of colors across the whole set, presenting the best results for all values of n and p in this study. Note that in the graph with 100 vertices, with a probability p of 0.5, the difference between the ordering with worst results (ascending order) and the ordering with the best results (descending order) is three colors. For both graphs of

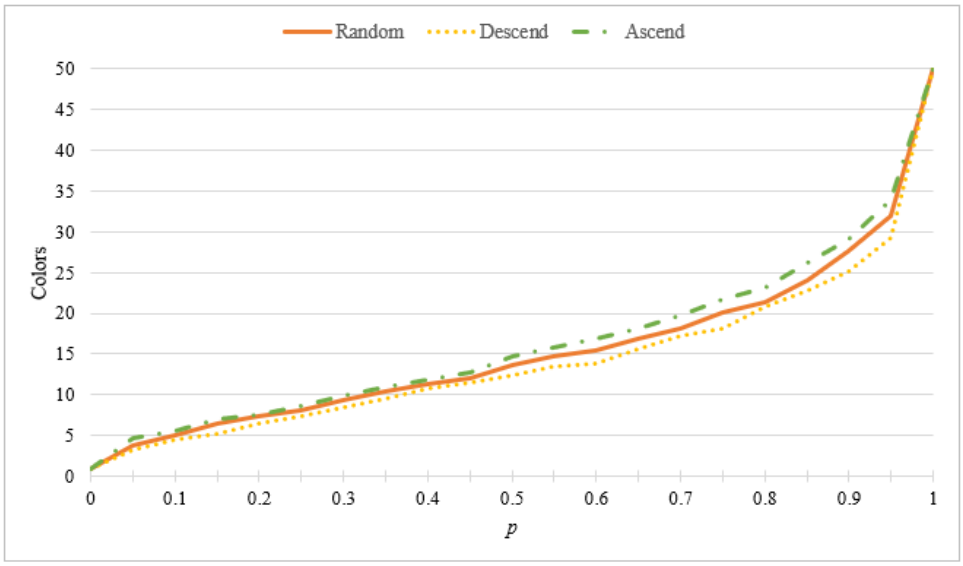

FIGURE 3.12. Average number of colors produced by the different orderings of the Greedy algorithm for a network with n=50.

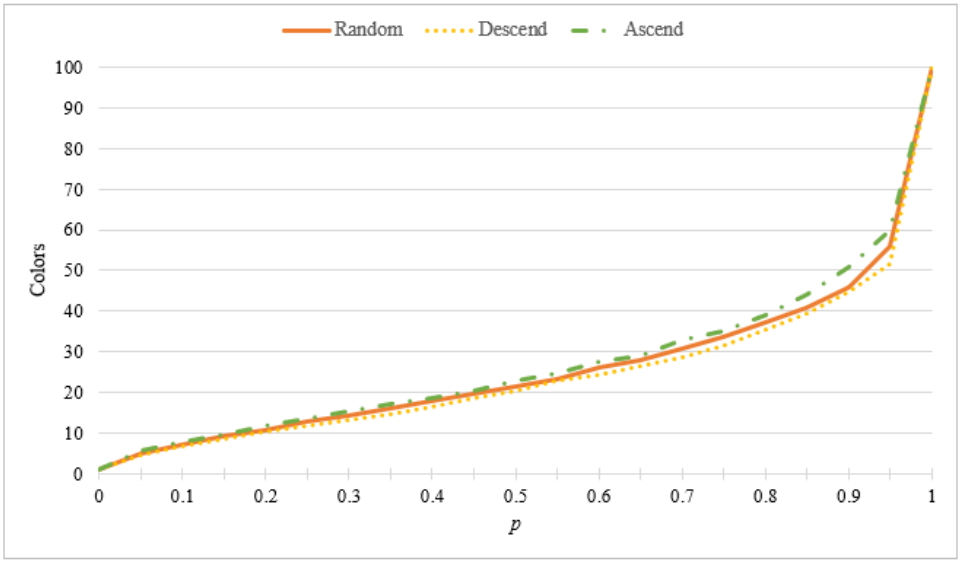

FIGURE 3.13. Average number of colors produced by the different orderings of the Greedy algorithm for a network with n=100.

50 and 20 vertices the difference between these two types of ordering is two colors. For probabilities less than 0.2, the difference between the two orderings for a graph of 100 vertices is approximately two colors, while for a graph of 50 and 20 vertices, this difference is only 1 color. Similarly to the previous study, for probabilities close to 1, the different ordering curves tend to approximate each other, reaching the maximum limit of colors that each n can achieve.

After the results obtained for the ascending, random and descending order of degree from the Greedy algorithm, we compared the Greedy algorithm that leads to the lower number of colors, i. e., with a descending order of degree, with the DSATUR and RLF algorithms results shown in Figures 3.8 - 3.10. This analysis can be seen in the Figure

3.14, for n=100. The values of the Greedy algorithm, although slightly decreased with the descending order still show that this algorithm generates slightly more colors than the DSATUR and RLF algorithms.

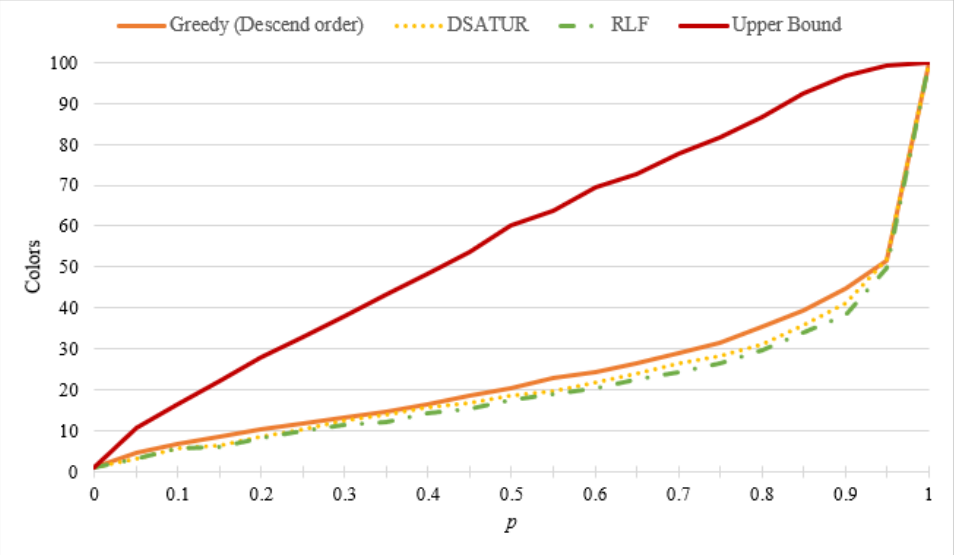

FIGURE 3.14. Average number of colors produced by the Greedy algorithms with the vertices ordered in descending order of degree, DSATUR and RLF for a network with n=100.

For values of n smaller than 100, for example for n=50 and n=20, and maintaining the descending order of degree, the Greedy algorithm generates practically the same number of colors than the DSATUR algorithm. If the Greedy algorithm had the vertices ordered in ascending order of degree for the three values of n studied, it would be observed that the algorithm would give also the worst solution, accentuating even more its values.

Regarding the computational time that these orderings require to produce a possible solution for the three values of n studied, it was verified that the descending order of degree besides producing less colors also requires less computational time. On the other hand, the ascending order of degree requires more time to generate a specific solutions. Table 3.4 presents the computation times obtained for each n studied.

TABLE 3.4. Computational time of the Greedy algorithm for different ordering strategies for a p=0.5.

n	Random order	Descending order	Ascending order
20	3.2 5s	2.98 s	3.54 s
50	3.65 s	3.46 s	3.83 s
100	3.93 s	3.64 s	4.52 s

3.5. Conclusion

In this chapter, the flowchart that characterises the planning tool developed in this work was presented. It was also explained how the routing algorithm *Yen's* k-shortest path was implemented in the tool developed. The pseudocodes of the implemented Graph

Coloring algorithms were also presented. Through these pseudocodes, it became possible to develop the Greedy, DSATUR and RLF algorithms in the planning tool for optical networks.

After the implementation of these algorithms, a performance assessment was carried out which allowed us to take some conclusions regarding the Graph Coloring heuristics for graphs with different number of vertices. The path matrices corresponding to the graphs *G(W,P)*, were randomly generated in order to comply with a given value of p. For all tested values of n, the RLF algorithm is the algorithm that produces the minimum number colors, but in contrast requires the higher computational time to generate them. The Greedy algorithm, although presenting the worst solution, since it assigns more colors to the graphs, is the quickest algorithm to generate solutions. In this particular study, the order in which the vertices of the Greedy algorithm were ordered was through, the random order of degree. The DSATUR algorithm typically attributes less colors than the Greedy algorithm and slightly more colors than the RLF algorithm. For example, for p=0.5 the Greedy heuristic provides 21.3 colors, whereas the RLF heuristics assigns 17.5 colors, which gives a 4 colors reduction.

Since the Greedy algorithm sorts the vertices in any arbitrary order, the performance of the algorithm for different vertices sorting has also been studied. From the orderings studied, it was concluded that the descending order of degree is the ordering that produces a lower number of colors for all values of n. In opposition, the ascending order of degree presents the worst solution, assigning more colors to the graphs. If the Greedy algorithm with the vertices ordered in descending order of degree is compared with the DSATUR and RLF algorithms, the differences between the number of assigned wavelengths would decrease. However, for n=100 vertices, it would still produce the worst solutions and, for n=50 and n=20 vertices their solutions would be similar to the solutions generated by the DSATUR algorithm for each of these graphs.

CHAPTER 4

Application of Graph Coloring Heuristics to the Planning of Real Networks

4.1. Introduction

In this chapter, the RWA tool, presented in Chapter 3, is applied to real networks and to ring networks with different numbers of nodes. The real networks that are studied in the RWA tool developed in this work are: COST 239, NSFNET, UBN, two variations of the CONUS network, one with 30 nodes and other with 60 nodes and two variations of the network generated by the GT-ITM tool in [**10**], one with 27 nodes and other with 34 nodes. Besides these networks, ring networks with 25, 35 and 45 nodes are also studied. Section 4.2 presents and characterises these networks from the perspective of the physical topology and, section 4.3 characterises them from the perspective of the logical topology. Next, section 4.4 presents the results and computational times obtained by the RWA tool for each of these networks. The solutions obtained by the First Fit and Most Used algorithms are also shown, and then the results obtained by the Graph Coloring heuristics are analysed. Then, the main conclusions obtained by all these algorithms are compared and discussed. Finally, for ring networks from 5 nodes to 60 nodes, the solutions obtained with the wavelength assignment algorithms are compared with the solutions obtained from an analytical expression that gives the optimum number of wavelengths.

4.2. Parameters of the Network Physical Topology

This section describes the real networks that are used in the tool developed in this work that implements various wavelength assignment algorithms, as well as characterises these networks from the perspective of their physical topology, according to the following parameters [**21**]:

- Total number of nodes in the network;
- Total number of links in the network;
- Maximum cost that exists between two adjacent nodes in the network. The cost metric for these real networks is measured through the maximum distance in km between the two adjacent nodes;
- Minimum cost that exists between two adjacent nodes in the network. The cost metric for these real networks is measured through the minimum distance in km between the two adjacent nodes;

- Average node degree that exists in the physical topology of the network that is given by [**21**]:

$$\overline{d} = \frac{\sum_{i=1}^{N} degree_i}{N} \tag{4.1}$$

  where N is the total number of the nodes of the network and $degree_i$ represents the node degree of vertex i.

- Variance of the node degree that measures the regularity of the network (how similar the nodes of the network are in terms of the number of connections) and complements the information given by the average node degree [**21**]. Its expression is as follows:

$$\sigma^2 = \frac{\sum_{i=1}^{N} (degree_i - \overline{d})^2}{N-1} \tag{4.2}$$

In a scenario where the variance of the node degree is zero, it means that all the nodes of the network have the same node degree and the same number of incoming/outcoming connections, as it happens for example in ring networks. As the value of the variance increases, it means that the similarity of the network is reduced.

The cost matrix for the real networks studied is obtained from the link cost in kilometres. The optical links are assumed as bidirectional links.

The real networks that are used in this work are described next and the physical topology of each of the networks can be seen in Appendix A.

The European Optical Network, COST 239 (Figure A.1), is a transparent optical network that is used as a reference for network topologies studies and was implemented between 1992 and 1998 to carry all international traffic between the main centres of Europe [**6**], [**22**]. The 11 nodes of the network are connected by 26 fibre optic links.

TABLE 4.1. COST239 network physical topology parameters.

Nodes	**Links**	**Max. Distance [km]**	**Min. Distance [km]**	**Avg. Node Degree**	**Variance Node Degree**
11	26	953	171	4.7	0.4

NSFNET (National Science Foundation Network) (Figure A.2) is a national computer network that was designed in the 1990's to improve communications, collaboration and resource sharing within the United States scientific and engineering research community [**7**]. This backbone computer network link was implemented with optical fibres to carry data, and is also a typical reference network to perform network studies [**7**], [**23**].

The US backbone network, better known as the UBN network (Figure A.3), is an American network, that emerged between 1997 and 1999 to respond to the rapid and massive flow of information through urban areas where the distance requirements of the

TABLE 4.2. NSFNET network physical topology parameters.

Nodes	Links	Max. Distance [km]	Min. Distance [km]	Avg. Node Degree	Variance Node Degree
14	21	2828	246	3.0	0.3

lightpath varied considerably. This network is constituted by 24 nodes interconnected by 43 bidirectional links [**8**].

TABLE 4.3. UBN network physical topology parameters.

Nodes	Links	Max. Distance [km]	Min. Distance [km]	Avg. Node Degree	Variance Node Degree
24	43	2600	250	3.6	0.9

The Continental United States (CONUS) network is a fibre optic network that covers the whole territory of the United States of America, developed for use in research on large-scale DWDM networks [**9**], [**24**]. It was designed to provide a high degree of protection, since it has some cross-continental paths completely interconnected, which is not a common feature in U.S.A carrier networks [**9**]. In this work, two variations of the CONUS network are used: CONUS network with 30 nodes (Figure A.4) and CONUS network with 60 nodes (Figure A.5) [**9**].

TABLE 4.4. CONUS network physical topology parameters with 30 nodes.

Nodes	Links	Max. Distance [km]	Min. Distance [km]	Avg. Node Degree	Variance Node Degree
30	36	1467	69	2.4	0.4

TABLE 4.5. CONUS network physical topology parameters with 60 nodes.

Nodes	Links	Max. Distance [km]	Min. Distance [km]	Avg. Node Degree	Variance Node Degree
60	79	1468	24	2.6	0.5

The GT-ITM (Georgia Tech Internet Topology Modeler) tool generates pseudo-random network topologies on which researchers can perform their analyses [**25**]. The networks presented in this work are a simplified variation, with only 27 nodes (Figure A.6) and 34 nodes (Figure A.7), of the network that was generated by this tool for the work presented in [**10**]. The nodes of the topologies generated by this tool are organised in logical domains, in which the nodes of each domain are interconnected with each other, forming a cluster of nodes, and rarely connect to the nodes outside their domain [**25**]. As this network tool does not provide distances between nodes, we have considered that the distances between the nodes of these networks are all equal.

We observe that the networks generated by the GT-ITM tool have a relatively high variance node degree compared to the other real networks studied, which means that the nodes in the GT-ITM network have little symmetry between them.

TABLE 4.6. Network physical topology parameters with 27 nodes generated by the GT-ITM tool.

Nodes	**Links**	**Avg. Node Degree**	**Variance Node Degree**
27	36	2.7	2.0

TABLE 4.7. Network physical topology parameters with 34 nodes generated by the GT-ITM tool.

Nodes	**Links**	**Avg. Node Degree**	**Variance Node Degree**
34	44	2.6	1.8

In addition to these networks, ring networks with 25, 35 and 45 nodes are also studied. These ring networks are considered, as they present a lower average node degree compared to the above mentioned networks. The cost metric of these networks is obtained from the number of hops, and similarly to the previous networks, the optical links are assumed bidirectional. In the following tables, the parameters of the maximum and minimum distances between two adjacent nodes will not be presented, since, we have considered that in these networks, the distances between nodes are all equal.

TABLE 4.8. Ring networks physical topology parameters with 25, 35 and 45 nodes.

Networks	**Nodes**	**Links**	**Avg. Node Degree**	**Variance Node Degree**
Ring 25	25	25	2.0	0.0
Ring 35	35	35	2.0	0.0
Ring 45	45	45	2.0	0.0

From the analysis of Table 4.8, we verify that regardless of the number of nodes and links in the ring networks, these networks always have the same characteristics, the average node degree of 2 and variance node degree equal to zero. The average node degree is always 2, since in this type of network, the nodes are always connected to two adjacent nodes. The variance node degree is always zero, because ring networks are symmetrical, i.e. all nodes have two outgoings connections.

4.3. Parameters of the Network Logical Topology

While the previous section focused on characterising the networks used in this work from the perspective of the physical topology, this section characterises the same networks from the perspective of the logical topology. The parameters used are the same as those used in the previous section: total number of nodes, total number of links, average node degree and variance node degree [**21**]. The maximum and minimum distance parameters do not apply to this characterisation, since in the logical topology we are only concerned with traffic units. For each network, a full mesh logical topology with one traffic unit in each

path is always considered. Considering a logical topology with these characteristics, the total number of edges is calculated using the following expression:

$$N_{edges} = \frac{N(N-1)}{2} \tag{4.3}$$

Regarding the average node degree, by applying eq. (4.1) to a full mesh logical topology, we obtain a simple expression, N-1, since the nodes in the network have the same degree, because they send traffic to all other nodes except themselves. Due to this characteristic, the variance node degree of the logical topology considered is always zero, since each node of the network has an equal number of links.

TABLE 4.9. Logical topology parameters of the networks studied in this work.

Networks	**Nodes**	**Logical Links**	**Avg. Node Degree**	**Variance Node Degree**
COST 239	11	55	10	0.0
NSFNET	14	91	13	0.0
UBN	24	276	23	0.0
CONUS 30	30	435	29	0.0
CONUS 60	60	1770	59	0.0
GT-ITM 27	27	351	26	0.0
GT-ITM 34	34	561	33	0.0
Ring 25	25	300	24	0.0
Ring 35	35	595	34	0.0
Ring 45	45	990	44	0.0

When analysing the results given in Table 4.9, we observe that the total number of logical links in each network, corresponds to the total number of paths that exist in the network, which are used to generate the path matrix corresponding to the graph *G(W,P)* used in the Graph Coloring algorithms.

4.4. Results of the RWA Tool

In this section, the RWA tool, described in section 3.2, to perform the planning of the networks mentioned in the previous sections is applied. As mentioned, for each network, a full mesh logical topology with one traffic unit in each path is considered. Furthermore, all results presented are obtained for unprotected networks. Demands are routed using the *Yen's* k-shortest path algorithm. Once the routing is completed, the wavelength assignment algorithms are applied to the different networks to finalise the optical network planning.

4.4.1. First Fit and Most Used Algorithms

Table 4.10 shows the total number of wavelengths obtained by the conventional WA algorithms, First Fit and Most Used, for each of the networks presented in section 4.2.

Before applying these algorithms, the network demands are sorted according to three different orderings: shortest path first, longest path first and random path. The values of the random path ordering strategy represent the total average of wavelengths obtained after 10 simulation runs.

TABLE 4.10. Total number of wavelengths obtained by First Fit and Most Used algorithms for the networks described in section 4.2.

Networks	Ordering Strategy	First Fit	Most Used
COST 239	Shortest path first	8	9
	Longest path first	8	8
	Random path	8.2	8.3
NSFNET	Shortest path first	24	25
	Longest path first	24	24
	Random path	24.0	24.2
UBN	Shortest path first	70	71
	Longest path first	64	64
	Random path	66.0	66.7
CONUS 30	Shortest path first	134	135
	Longest path first	123	124
	Random path	123.5	123.8
CONUS 60	Shortest path first	550	551
	Longest path first	543	543
	Random path	543.5	543.7
GT-ITM 27	Shortest path first	229	229
	Longest path first	221	221
	Random path	221.3	221.1
GT-ITM 34	Shortest path first	356	356
	Longest path first	347	347
	Random path	347.1	347.2
Ring 25	Shortest path first	98	94
	Longest path first	78	79
	Random path	89.6	90.7
Ring 35	Shortest path first	184	192
	Longest path first	153	154
	Random path	174.9	175.9
Ring 45	Shortest path first	318	316
	Longest path first	253	254
	Random path	286.0	286.4

In general, we observe that the values presented by the two algorithms in Table 4.10 are quite identical, although the Most Used algorithm tends to give a slightly superior number of wavelengths. Analysing Table 4.10 with more detail, we also verify that for networks with fewer nodes (COST 239 and NSFNET networks), the number of wavelengths obtained by both the First Fit algorithm and the Most Used algorithm using the different ordering strategies, are quite similar, and there are no significant differences between the algorithms. For networks with more nodes than the NSFNET network, some differences between the number of assigned wavelengths between the two algorithms began to

emerge, as well as between the ordering strategies. The shortest path first ordering used by both algorithms gives, in general, a higher number of wavelengths, when compared to the longest path first ordering strategy, which for all networks assigns the lowest number of wavelengths. The longest path first ordering strategy assigns the lowest number of wavelengths to all networks because the first demands to be assigned wavelengths are the demands that cross several links of the network, which have more hops/distance between the nodes. When a demand goes through several links in the network, those links will have all the same wavelength. In other words, this strategy starts by assigning the wavelengths to the longest paths and, therefore, the same wavelength will be used in many links, which allows reducing the total number of assigned wavelengths. The same does not happen with the shortest path first ordering. In this strategy, as wavelengths start to be assigned from the shortest paths, it is not guaranteed that a specific wavelength will be used in many links. Hence, it is more probable that the total number of assigned wavelengths becomes higher.

The random path ordering applied to real networks gives wavelength values very similar to the longest path first strategy. But it is deceitful to conclude that the optimal wavelength solution can be obtained with this ordering, because the values presented in Table 4.10 are obtained after an average of 10 simulations, which is a clear disadvantage compared to the longest path first ordering. This similarity between the results obtained by the random path and longest path first ordering strategies is not verified in ring networks. In these networks, the random path ordering gives a higher number of wavelengths than the longest path first and lower values than the shortest path first. The fact that the random path ordering presents intermediate values between the other two ordering is not only due to the symmetry of the ring networks (as explained at the beginning of this chapter), but also due to the fact that there is no pre-defined ordering strategy. Because the demands are randomly ordered, the paths of the demands are equally distributed across nodes and network links, to avoid overloading a link with many paths, which leads to an intermediate number of wavelengths between the longest path first and shortest path first strategies in ring networks. Furthermore, it should be mentioned that, in [**13**], the obtained results are much similar to those presented in table 4.10.

The computation times that the RWA tool requires to return a viable RWA solution applying the First Fit and Most Used algorithms are shown in Table 4.11. The computational times obtained by the two algorithms are quite similar, independently of the network considered. These computational times are consistent with the network size, i.e. the larger is the network in terms of number of nodes, the longer the algorithms require to return a viable solution. Regarding the ordering strategies, the random path strategy requires the lower computational time, followed by the longest path first strategy and the shortest path strategy with longer computational times. These results are due to the fact that the random path strategy uses fewer decisions/instructions during the execution of

the algorithms, since the demands have no particular order, as opposed to the other two strategies, which have to take more decisions/instructions.

TABLE 4.11. Computational times of RWA tool regarding the First Fit and Most Used algorithms.

Networks	Ordering Strategy	First Fit	Most Used
COST 239	Shortest path first	3.9 s	4.0 s
	Longest path first	3.8 s	3.5 s
	Random path	3.2 s	3.2 s
NSFNET	Shortest path first	4.7 s	4.9 s
	Longest path first	4.3 s	4.7 s
	Random path	3.9 s	3.9 s
UBN	Shortest path first	8.6 s	8.2 s
	Longest path first	7.2 s	7.2 s
	Random path	6.5 s	6.5 s
CONUS 30	Shortest path first	8.5 s	8.2 s
	Longest path first	7.2 s	7.4 s
	Random path	7.0 s	6.4 s
CONUS 60	Shortest path first	33.3 s	28.9 s
	Longest path first	32.5 s	30.2 s
	Random path	31.5 s	30.7 s
GT-ITM 27	Shortest path first	8.5 s	8.8 s
	Longest path first	7.6 s	7.8 s
	Random path	6.9 s	6.9 s
GT-ITM 34	Shortest path first	9.2 s	9.6 s
	Longest path first	8.1 s	8.4 s
	Random path	7.8 s	8.0 s
Ring 25	Shortest path first	7.3 s	7.5 s
	Longest path first	6.9 s	6.8 s
	Random path	6.3 s	6.3 s
Ring 35	Shortest path first	10.3 s	10.3 s
	Longest path first	9.4 s	9.7 s
	Random path	9.2 s	8.9 s
Ring 45	Shortest path first	16.4 s	16.8 s
	Longest path first	15.7 s	15.7 s
	Random path	14.9 s	14.4 s

4.4.2. Graph Coloring Heuristics

Before presenting the results obtained by the Graph Coloring heuristics, it is necessary to characterise some parameters related to the graph *G(W,P)*:

- Total number of paths obtained by the graph *G(W,P)*;
- The probability of each pair of vertices of the graph *G(W,P)* being adjacent, *p*, as defined in equation (3.1), since this parameter influences the number of wavelengths in the network [**5**].

Table 4.12 shows the total number of paths and the parameter *p* for each of the networks studied in this work.

TABLE 4.12. Parameters of the network logical topology for the application of the Graph Coloring heuristics.

Networks	Total Paths	p
COST 239	55	0.1010
NSFNET	91	0.2501
UBN	276	0.2250
CONUS 30	435	0.3792
CONUS 60	1770	0.3208
GT-ITM 27	351	0.5571
GT-ITM 34	561	0.5965
Ring 25	300	0.4783
Ring 35	595	0.4848
Ring 45	990	0.4884

From the analysis of Table 4.12, we observe that the parameter p of each of the ring networks is around 0.48 when compared to most of the networks used in this work since a specific link in the ring networks is used by half of the paths that originate from that node. This occurs since ring networks have only two physical connections departing from each node of the network. This leads to a high p in ring networks compared to the other networks, because one physical link is most probably used by a higher number of logical paths. This justification also applies to the real networks generated by the GT-ITM tool that exceed a parameter p of 0.5 and achieve a $p = 0.5571$ with 27 nodes and a $p = 0.5965$ with 34 nodes. These networks are divided into 3 and 4 domains, with more than 7 nodes inside each domain, as shown in Figures A.6 and A.7, respectively, which are interconnected with each other by two central nodes. This centralisation increases the number of paths that pass through the physical links connected to the central node, increasing the p parameter of the network.

Table 4.13 presents the total number of wavelengths obtained by the Greedy, DSATUR and RLF algorithms for each of the networks characterised in section 4.2. Note that the Greedy algorithm in these simulations considers the vertices ordered in descending order of degree, which as shown in Chapter 3, leads to the minimum number of wavelengths estimated by the Greedy algorithm.

Analysing the results of Table 4.13, we verify that the COST239, NSFNET and UBN networks do not present any differences between the numbers of wavelengths obtained by any of the heuristics. These results were expected, since the parameter p of each of these networks is low. Remembering the conclusions taken in section 3.4 regarding the performance of the Greedy, DSATUR and RLF algorithms, we have observed that the number of wavelengths given by the three algorithms is quite similar, for networks with low p. The same conclusion can be taken for the CONUS 60 network. This network, despite having 1770 nodes, presents a low parameter p for the number of nodes it has. As we saw in section 3.4, the higher the number of nodes existing in the networks, the nearer the number of wavelengths estimated by the three Graph Coloring algorithms. Therefore, there is

TABLE 4.13. Total number of wavelengths obtained by Greedy, DSATUR and RLF algorithms for the networks described in section 4.2.

Networks	Greedy	DSATUR	RLF
COST 239	8	8	8
NSFNET	24	24	24
UBN	64	64	64
CONUS 30	123	123	119
CONUS 60	543	543	543
GT-ITM 27	221	217	216
GT-ITM 34	347	342	338
Ring 25	78	76	76
Ring 35	153	149	147
Ring 45	253	249	247

no difference between the results of the Graph Coloring heuristics for the CONUS 60 network. The CONUS 30 network presents a slight difference in the number of wavelengths obtained by the RLF algorithm (119 wavelengths), since it has a higher p ($p = 0.3792$) and has a smaller number of vertices compared to the CONUS 60 network. Therefore, for the CONUS 30 network, the RLF algorithm estimates better results than the other algorithms. The GT-ITM networks, having a p above 0.5, shows a more clear difference between the number of wavelengths predicted by the algorithms as expected from the results shown in chapter 3. For these networks, the RLF algorithm presents the best results reducing the number of assigned wavelengths by 5 for the network with 27 nodes and by 9 for the network with 34 nodes, in comparison with the Greedy algorithm. In this case, the DSATUR brings also advantages, since it assigns less 4 wavelengths than the Greedy algorithm in the network with 27 nodes and less 5 wavelengths in the network with 34 nodes. Therefore, these results for the GT-ITM networks, are in agreement with the conclusions taken in Chapter 3, which have shown that, when the p parameter varies between 0.5 and 0.9, the variations between these three Graph Coloring heuristics become more notorious.

The ring networks, having a p value around 0.5, also show a visible difference between the number of wavelengths obtained by the Graph Coloring algorithms. The ring network with 25 vertices presents the same number of wavelengths for both the DSATUR and RLF algorithms (76 wavelengths). Rings networks with 35 and 45 nodes present a higher differences in the number of colors obtained by the three heuristics.

Regarding the computational times of the RWA tool with the application of these Graph Coloring heuristics shown in Table 4.14, the conclusions are similar to the ones found in section 3.4. The Greedy algorithm requires less time to obtain a solution, while the RLF algorithm requires a higher computational time to return a viable solution.

Comparing Tables 4.10 and 4.13, we verify that the results obtained by the First Fit algorithm with the demands ordered through the longest path first ordering strategy, are quite similar to the results achieved by the Graph Coloring heuristics, more precisely by

TABLE 4.14. Computational times of the RWA tool regarding the Greedy, DSATUR and RLF algorithms.

Networks	Greedy	DSATUR	RLF
COST 239	4.0 s	4.2 s	4.5 s
NSFNET	4.1 s	4.7 s	5.7 s
UBN	7.2 s	8.4 s	9.2 s
CONUS 30	14.2 s	16.5 s	17.5 s
CONUS 60	162.4 s	164.4 s	167.5 s
GT-ITM 27	8.9 s	10.7 s	11.2 s
GT-ITM 34	16.7 s	17.4 s	18.1 s
Ring 25	6.7 s	7.4 s	7.8 s
Ring 35	16.9 s	17.1 s	18.6 s
Ring 45	43.4 s	49.0 s	53.1 s

the Greedy algorithm with descending order of degree, where exactly the same number of wavelengths is obtained, for all studied networks. Depending on the different network scenarios, we conclude that the RLF algorithm tends to present the best results in comparison with the remaining algorithms, assigning a smaller number of wavelengths to the real networks with higher p. It is also shown that both this algorithm and the DSATUR algorithm are advantageous for wavelength assignment for networks with a p parameter above 0.48, since they obtain lower wavelength values than the "most popular" algorithms (First Fit, Most Used and Greedy). Although, the DSATUR and RLF algorithms tend to minimise the number of wavelengths, they take higher computational times to return a viable wavelength planning solution.

4.4.3. Application of the RWA Tool in Ring Networks

Since ring networks are an example of regular networks [**26**], there is an analytical expression that allows obtaining the minimum number of wavelengths that the network requires. Therefore, in this section, the results obtained by the RWA tool will be compared with the results obtained by the analytical expression given in [**27**]. This comparison is applied to ring networks with size varying from 5 vertices to 60 vertices.

Table 4.15 shows the total number of nodes (paths) that each network has and the corresponding p parameter. As expected, for higher network size in terms of the number of nodes, the p parameter tends to approach 0.5, since the number of paths through each physical link tends to be half of the total number of paths.

Since ring networks are regular networks [**26**], as they have a regular geometry, there is an analytical expression that allows us to obtain the optimal number of wavelengths to be used in this type of networks [**27**], that is, the minimum value of wavelengths that the network requires. For a total meshed logical network, if the network has an even number of nodes, the expression is:

TABLE 4.15. Total paths and parameter p of each ring network.

Networks	Total Paths	p
Ring 5	10	0.3333
Ring 10	45	0.4444
Ring 15	105	0.4615
Ring 20	190	0.4737
Ring 25	300	0.4783
Ring 30	435	0.4828
Ring 35	595	0.4848
Ring 40	780	0.4872
Ring 45	990	0.4884
Ring 50	1225	0.4898
Ring 55	1485	0.4906
Ring 60	1770	0.4915

$$W = \frac{N^2 + 2N}{8} \tag{4.4}$$

If the network has an odd number of nodes, the expression is:

$$W = \frac{(N-1)^2 + 2(N-1)}{8} \tag{4.5}$$

Table 4.16 shows the total number of wavelengths obtained by the First Fit, Most Used, Greedy, DSATUR, RLF algorithms, and by the analytical expressions (4.4) and (4.5) for each of these networks.

TABLE 4.16. Total number of wavelengths obtained by First Fit, Most Used, Greedy, DSATUR, RLF algorithms and by the analytical expressions (4.4) and (4.5) for each ring network.

Ring Size	First Fit			Most Used			Greedy	DSATUR	RLF	Eqs. 4.4 & 4.5
	SPF	LPF	RP	SPF	LPF	RP				
5	4	3	3.5	4	3	4.0	3	3	3	3
10	16	15	15.8	16	15	15.9	15	15	15	15
15	34	28	32.7	35	29	33.7	28	28	28	28
20	66	55	59.5	64	55	59.8	55	55	55	55
25	98	78	86.9	94	79	90.7	78	76	76	78
30	144	120	130.5	139	120	129.1	120	120	120	120
35	184	153	174.9	192	154	175.9	153	149	147	153
40	257	210	229.7	253	216	228.9	211	210	210	210
45	318	253	286.0	316	254	286.4	253	249	247	253
50	384	325	353.7	387	326	356.3	327	325	323	325
55	448	378	423.6	465	379	424.6	378	376	373	378
60	563	465	503.9	555	466	506.6	469	465	463	465

The results presented in Table 4.16 are in agreement with the conclusions taken from Tables 4.10 and 4.13 for ring networks. The First Fit algorithm ordered using the longest path first strategy tends to give wavelength values very close or identical to the results obtained by the Graph Coloring heuristics. Using the random path ordering strategy, the First Fit and Most Used algorithms present a number of intermediate wavelengths between the number of wavelengths obtained by the longest path first and shortest path first strategies. Regarding the number of wavelengths obtained by the Graph Coloring algorithms, the higher the p parameter of each network, the differences between the results obtained by the WA algorithms become more evident.

Comparing the values obtained by equations (4.4) and (4.5) with the values obtained by the RWA tool, we observe that the algorithms that obtain the minimum number of wavelengths for most of the ring networks analysed are the First Fit algorithm ordered through the longest path first strategy and the Greedy algorithm. In networks with 40, 50 and 60 nodes, the algorithm that obtains the optimal value, in addition to the First Fit algorithm combined with the longest path first ordering, is the DSATUR algorithm, except for the network with 40 nodes, where the optimum value is also obtained by the RLF algorithm.

However, by a detailed analysis of Table 4.16, we verify that in ring networks with 25, 35, 45, 50, 55 and 60 nodes, the number of wavelengths generated by the DSATUR and/or RLF algorithms (because in networks with 50 and 60 nodes only happen in the RLF algorithm) are lower than the value obtained by the analytical expressions for those networks. This situation happened only in these networks because, during their simulation, a wavelength assignment error occurred in some paths of the networks. The fact that this error appeared only in these regular networks does not invalidate the results generated by the same algorithms for the remaining real networks. After the implementation of the DSATUR and RLF algorithms in the RWA tool, the results obtained by the simulations in small size networks (networks up to 15 nodes) were confirmed manually. Furthermore, the implementation of these algorithms and the results of the RWA tool were also confirmed in chapter 3 with the results shown in [**5**] for networks with randomly generated path matrices up to 100 nodes. The average number of wavelengths predicted by the developed RWA tool has shown a very good agreement (to the decimal level) compared to the results presented in [**5**].

4.5. Conclusion

This chapter focused on presenting and discussing the results of the number of wavelengths assigned to the networks studied by the developed RWA tool. At the beginning of this chapter, the parameters of the physical and logical topologies that characterise the networks used in the tool were defined, as well as a brief description of each of these networks has been provided. Next, we showed the solutions produced by the RWA tool applied to some real networks using the WA algorithms studied: First Fit, Most Used, Greedy, DSATUR and RLF. In order to cover most of the existing scenarios, networks

of various sizes and shapes (COST 239, NSFNET, UBN, CONUS 30, CONUS 60, GT-ITM 27 and GT-ITM 34) have been considered. Firstly, we verified that the longest path first ordering strategy used by both First Fit algorithm and Most Used algorithm is the ordering that generates solutions more identical to the solutions generated by the Graph Coloring heuristics. Regarding the Graph Coloring heuristics and having in mind the network size, when the parameter p varies between 0.5 and 0.9, a more noticeable difference of the number of wavelengths attributed by the Greedy (with nodes ordered in descending order of degree), DSATUR and RLF algorithms is found.

As concluded also in the previous chapter, the RLF algorithm tends to produce the best solutions, minimising the number of wavelengths used in the networks. In addition, for a parameter p above 0.5, it has also been proven that this algorithm and the DSATUR algorithm are advantageous for wavelength assignment in the planning of optical networks, because they obtain better solutions than the most usual wavelength assignment algorithms. One example where these improvements are evident is in the networks generated by the GT-ITM tool whose physical topology is based on several cluster of nodes interconnected by central nodes using few links. For this type of networks, more specifically for a network with 34 nodes, the RLF algorithm reduces the number of assigned wavelengths by 9 and the DSATUR algorithm by 5 compared to the Greedy algorithm. The disadvantage of these two Graph Coloring algorithms is the computational time that they require to return a solution since they need more time compared to First Fit, Most Used and Greedy algorithms. Furthermore, for the networks studied, there is no significant advantage in applying these algorithms to networks with a parameter p below 0.5, because the number of wavelengths obtained is similar to the values obtained by the Greedy and First Fit algorithms.

CHAPTER 5

Conclusions and Future Work

5.1. Conclusions

This work focused on studying different Graph Coloring heuristics for solving the WA problem for static traffic in transport optical networks. The main goal of a static RWA scenario is to minimise the number of wavelengths to be used in a specific number of optical paths that compose the network. Thus, Chapter 2 focused on the introduction of the basic concepts concerning optical networks. The concepts of physical topology and logical topology of a network were defined, as well as the concepts of adjacent and traffic matrices. This chapter also includes a brief explanation of the OTN architecture and the network elements, known as ROADMs, used in these optical networks. The key concepts of planning an optical network were also presented, the routing and wavelength assignment, as well as the algorithms that were studied and implemented in this work: the routing algorithm *Yen's* k-shortest path and the WA algorithms First Fit, Most Used, Greedy, DSATUR and RLF. These last three algorithms belong to the Graph Coloring heuristics group.

Chapter 3 presented the flowchart that characterises the planning tool developed in this work in Matlab for network optimisation. Besides the explanation of how the tool was developed, this chapter also described the pseudocodes of the Graph Coloring heuristics. After implementing the Greedy (with three different degree ordering strategies), DSATUR and RLF algorithms based on the presented pseudocodes, a performance assessment of these three algorithms was carried out. This study was carried out isolated from the whole optical network planning tool, with the random generation of path matrices with a different number of vertices, in order to compare the obtained results with other published works and validate the implementation of the Graph Coloring heuristics. From this study, it was concluded that for values of p from 0.5 to 0.9, the RLF algorithm tends to produce fewer colors than the other algorithms, but in contrast, it requires more time to generate them, while the Greedy algorithm is faster and tends to give a slight worst solution, producing more colors. The DSATUR algorithm produces better solutions than the Greedy algorithm and slightly worse than the RLF algorithm. For values of p between 0.9 and 1, the colors produced by the three algorithms tend to approximate until the maximum limit of assigned colors is achieved. For values of p up to 0.2, the number of colors generated by each algorithm is also quite identical and, from this limit, Graph Coloring algorithms tend to show the behaviour already mentioned.

In Chapter 4, the results generated by the WA algorithms for real networks (COST 239, NSFNET, UBN, two versions of the CONUS network and two networks generated

by the GT-ITM tool) and for ring networks of 25, 35 and 45 vertices were presented and discussed. At the beginning of the chapter, a brief description of these networks has been presented and the parameters of the physical and logical topologies of these networks were characterised and calculated. Based on the results estimated by the several algorithms, the conclusions taken in chapter 3 were confirmed and generalised for real networks. Since the focus of this work was on the solutions produced by the RLF and DSATUR algorithms, due to the fact that they are not normally used in the optical networks planning, it was verified that these two algorithms are advantageous for wavelength assignment for the networks generated with the GT-ITM tool, as they obtain a smaller number of wavelengths than the most usual WA algorithms. In the GT-ITM network with 34 nodes, the RLF algorithm reduces the number of assigned wavelengths by 9 and the DSATUR algorithm by 5 when compared to the Greedy algorithm with the vertices ordered in descending order of degree. Hence, the RLF and DSATUR algorithms bring advantages in networks that exhibit several node clusters interconnected by one or two links, such as the ones generated by the GT-ITM tool. In contrast, these two algorithms, DSATUR and RLF, have a reduced advantage when applied to real networks with a parameter p below 0.5 because the number of wavelengths generated by these algorithms is quite similar to the one generated by the Greedy and First Fit algorithms. Furthermore, they require more computational time to return a solution.

5.2. Future Work

The study developed in this work focused on the performance and behaviour of the Graph Coloring heuristics for assigning wavelengths in the planning of an optical network. In addition, based on the study carried out in this dissertation, the future work that can be performed is the following:

- Expand the tool developed in this work to consider network protection;
- Consider other logical topologies besides the full mesh;
- Consider the OSNR of each path as a cost metric instead of the distance or the number of hops;
- Use different traffic demands with more than one traffic unit;
- Compare the results obtained by the heuristics developed in this work with the results predicted by exact algorithms;
- Generalise the tool developed in this work to respond to the dynamic traffic demands.

References

[1] J. Simmons. Optical Network Design and Planning, 2nd edition. New York, USA: Springer, 2014.

[2] R. Ramaswami, K. Sivarajan, and G. Sasaki. Optical Networks: A Practical Perspective, 3rd edition. USA: Morgan Kaufmann, 2010.

[3] L. Cancela. Slides from the course in Optical Networks. In *Telecommunications and Computer Engineering MSc.* ISCTE-IUL, Lisbon, Portugal, 2019.

[4] M. Aslan and N. Baykan. A performance comparison of Graph Coloring algorithms. In *International Conference on Advanced Technology Sciences (ICAT'16)*, pages 266–273, September 2016.

[5] R. M. R. Lewis. Algorithms and applications. In *A Guide to Graph Coloring.* Switzerland: Springer, 2016.

[6] M. Niksirat, S. Mehdi Hashemi, and M. Ghatee. Branch-and-price algorithm for fuzzy integer programming problems with block angular structure. *Fuzzy Sets Syst.*, vol. 296, no. 2, pp. 70–96, August 2016.

[7] T. L. LaQuey. NSFNET. In *The User's Directory of Computer Networks*, pages 247–250. Boston: Digital Press, 1990.

[8] E. Biernacka, J. Domzal, and R. Wójcik. Investigation of dynamic routing and spectrum allocation methods in elastic optical networks. *International Journal of Electronics and Telecommunications*, vol. 63, no. 1, pp. 85–92, February 2017.

[9] CONUS WDM network topology, http://monarchna.com/topology.html.

[10] R. Godoi, P. Hernandez, and E. Luque. Control Evaluation in a LVoD System Based on a Peerto-Peer Multicast Scheme. *Journal of Computer Science and Technology*, vol. 8, no. 2, pp. 97–103, 2008.

[11] J. Pires. Course Sistemas e Redes de Telecomunicações. In *Electrical and Computer Engineering MSc.* Instituto Superior Técnico, Lisbon, Portugal, 2006.

[12] G. Ellina, E. Bouillet, R. Ramamurthy, J.-F. Labourdette, S. Chaudhuri, and K. Bala. Restoration in Layered Architectures with a WDM Mesh Optical Layer. November 2003.

[13] C. Rodrigues. Optical Network Planning for Static Applications, Master dissertation in Telecommunications and Computer Engineering. pages 1–61. Lisboa, ISCTE-IUL, October 2018.

[14] International Telecommunication Union ITU. Architecture of optical transport networks. In *Digital Networks - Optical transport networks*, pages 1–70, January 2017.

[15] R. Valiveti. OTN Overview. CA, USA: System Architecture Group, Infinera Corp, 2012.

[16] M. Fukutoku. Next generation ROADM technology and applications. In *2015 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3, March 2015.

[17] S. Perrin. The Need for Next-Generation ROADM Networks. vol. 1, pp.1–15, September 2010.

[18] S. Gringeri, B. Basch, V. Shukla, R. Egorov, and T. J. Xia. Flexible architectures for optical transport nodes and networks. *IEEE Communications Magazine*, vol. 48, no. 7, pp. 40–50, July 2010.

[19] H. Zang, J. P. Jue, and B. Mukherjeey. A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks. *Optical Networks Magazine, SPIE publishers*, vol. 1, pp.47–60, March 2000.

[20] H. Chen, Y. Zhao, J. Zhang, W. Wang, and R. Zhu. Static Routing and Spectrum Assignment for Deadline-Driven Bulk-Data Transfer in Elastic Optical Networks. *IEEE Access*, vol. 5, pp. 13645–13653, July 2017.

[21] C. Fenger, E. Limal, and U. Gliese. Statistical study of the influence of topology on wavelength usage in WDM networks. In *Optical Fiber Communication Conference. Technical Digest Postconference Edition. Trends in Optics and Photonics Vol.37 (IEEE Cat. No. 00CH37079)*, vol. 1, pages 171–173. Baltimore, Maryland, USA, March 2000.

[22] M. C. Sinclair. Minimum Cost Topology Optimisation of the Cost 239 European Optical Network. In *Artificial Neural Nets and Genetic Algorithms*, pages 26–29. Springer-Verlag, Vienna, 1995.

[23] D. L. Mills and H. Braun. The NSFNET Backbone Network. In *Proceedings of the ACM Workshop on Frontiers in Computer Communications Technology*, SIGCOMM '87, page 191–196, New York, NY, USA, 1987.

[24] T. Zami, B. Lavigne, and M. Bertolini. How 64 GBaud optical carriers maximize the capacity in core elastic WDM networks with fewer transponders per Gb/s. *IEEE/OSA Journal of Optical Communications and Networking*, vol. 11, no. 1, pp. A20–A32, 2019.

[25] J. R. Eagan, J. Stasko, and E. Zegura. Interacting with Transit-Stub Network Visualizations. pages 1–2. GVU Center, College of Computing, Georgia Institute of Technology, Atlanta, 2005.

[26] P. Chentsho, L. Cancela, and J. Pires. A framework for analyzing in-band crosstalk accumulation in ROADM-based optical networks. *Optical Fiber Technology*, vol 57, pp.102238–102238, July 2020.

[27] I. Djordjevic and M. Cvijetic. Advanced Optical Communications: Systems and Networks. Norwood, MA: Artech House, 2013.

Appendices

APPENDIX A

Real Networks Topology

The physical topology of the real networks used in Chapter 4 is the following. Note that the numbers in the links represent the distance in km and the nodes are numbered sequentially with no particular order.

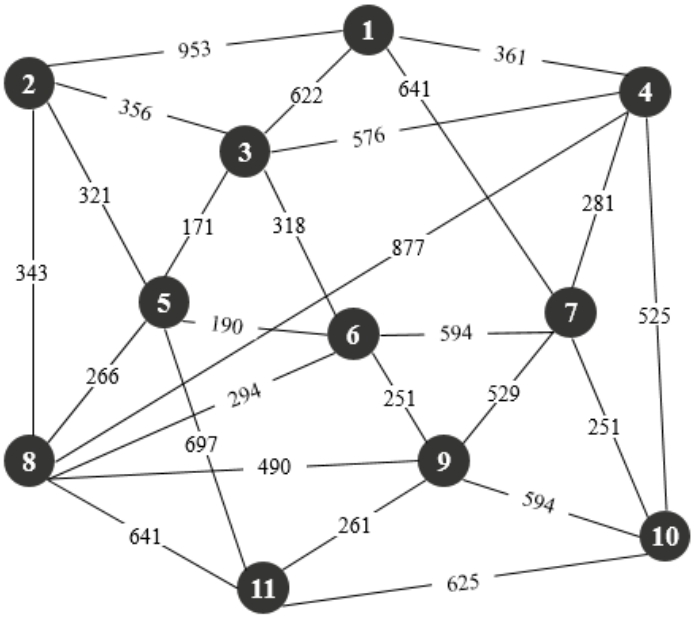

FIGURE A.1. Physical topology of the COST 239 network.

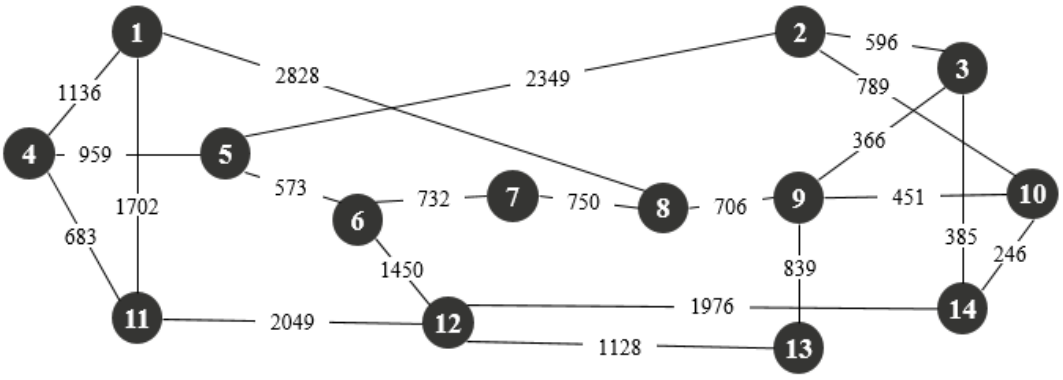

FIGURE A.2. Physical topology of the NSFNET network.

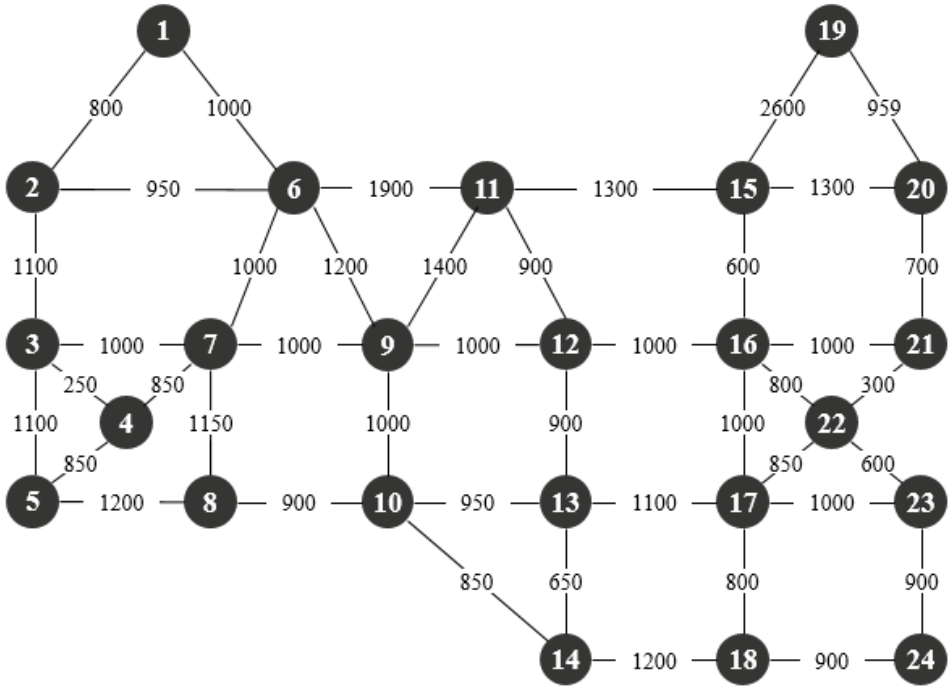

FIGURE A.3. Physical topology of the UBN network.

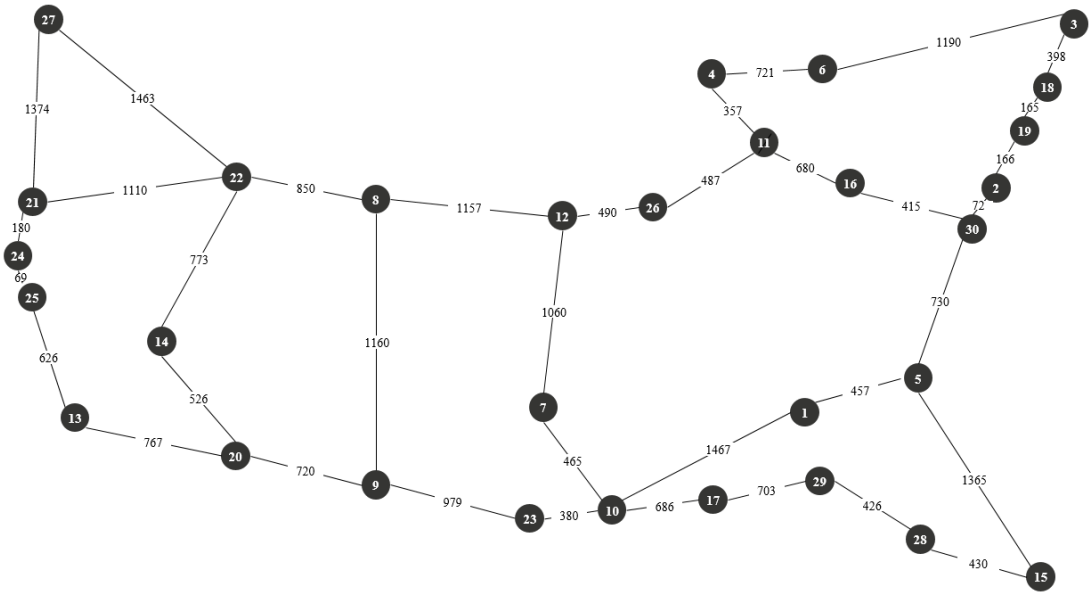

FIGURE A.4. Physical topology of the CONUS network with 30 nodes.

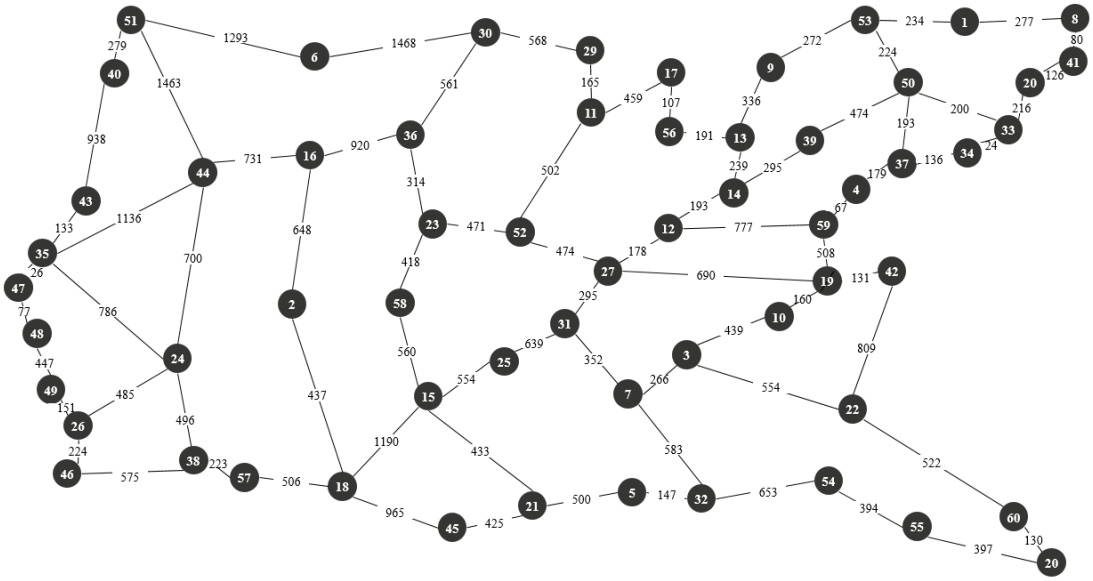

FIGURE A.5. Physical topology of the CONUS network with 60 nodes.

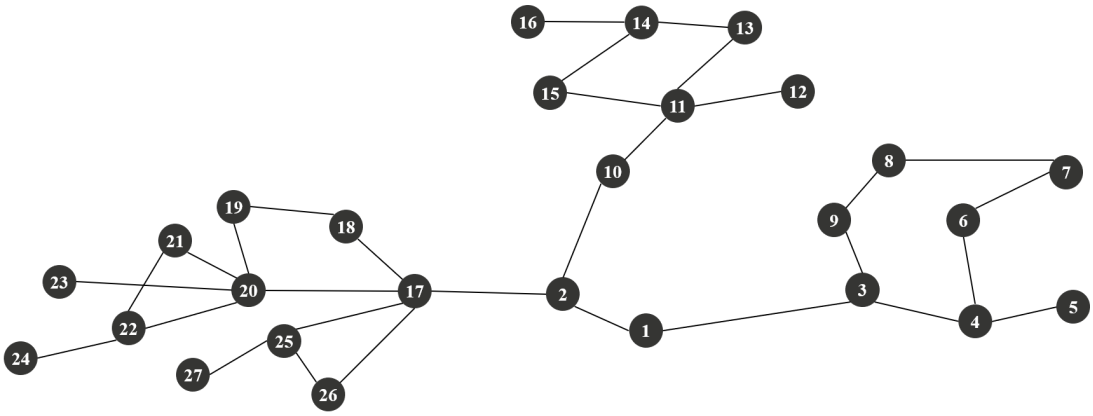

FIGURE A.6. Physical topology of the network with 27 nodes of the network generated by the GT-ITM tool.

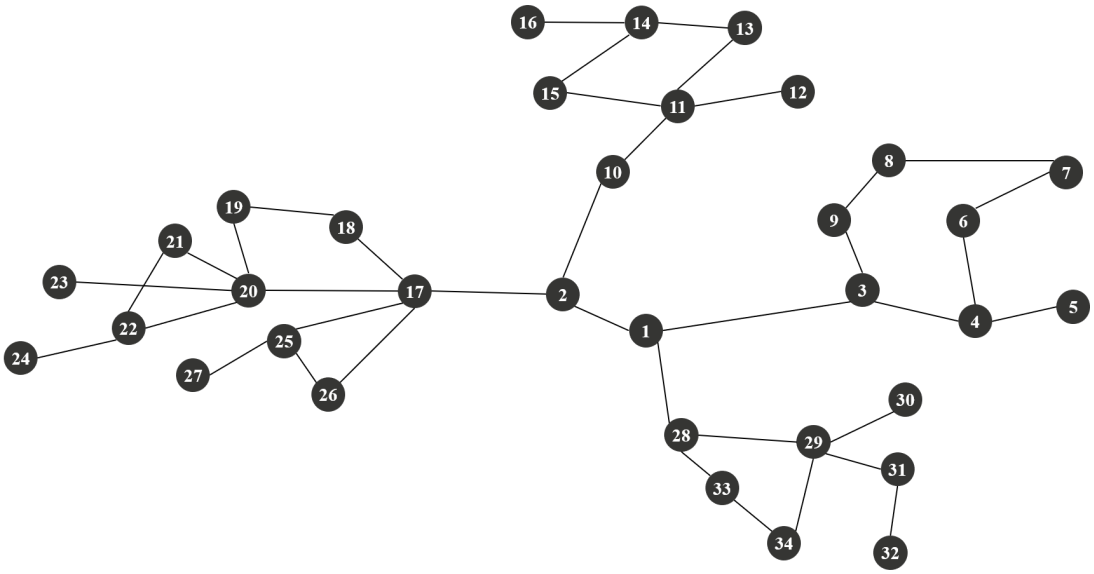

FIGURE A.7. Physical topology of the network with 34 nodes of the network generated by the GT-ITM tool.

APPENDIX B

Article: Graph Coloring Heuristics for Optical Networks Planning

The article was submitted to the conference ConfTele 2021, which will be realized in Leiria in February 2021.

Graph Coloring Heuristics for Optical Networks Planning

Inês Duarte[2], Luís Cancela[1,2], João Rebola[1,2]
[1]Optical Communications and Photonics Group, Instituto de Telecomunicações, Lisboa, Portugal
[2]Department of Science and Information Technology, Instituto Universitário de Lisboa (ISCTE-IUL), Lisboa, Portugal
imlde@iscte-iul.pt; luis.cancela@iscte-iul.pt; joao.rebola@iscte-iul.pt

***Abstract*—This work focuses on the study of wavelength assignment algorithms based on Graph Coloring techniques. We analyze the performance of the Greedy heuristic, a well-known Graph Coloring heuristic, as well as the Degree of Saturation (DSATUR) and the Recursive Largest First (RLF) heuristics, for planning optical networks. These last two heuristics, to the best of our knowledge, have not yet been applied in the context of optical networks. Extensive simulations have been performed, using real network topologies under a static traffic scenario and we have concluded that the DSATUR and RLF heuristics can outperform the Greedy heuristic in network scenarios where there are several network clusters interconnected by only one or two links. In these cases, the RLF and DSATUR heuristics can provide less 9 and 5 wavelengths, respectively, than the Greedy heuristic, in networks with 34 nodes.**

***Index Terms*—DSATUR, Graph Coloring, Greedy, Optical Networks, RLF, Wavelength Assignment.**

I. Introduction

Optical networks are essential in today's global communications, and the study of planning tools that efficiently allocate network resources is crucial to network providers [1]. Routing and wavelength assignment are two vital functions for allocating the network resources. The routing function consists on finding the best optical path to be used by an optical channel, and implies the selection of a set of links between the source and destination nodes, whereas the wavelength assignment function is responsible for choosing an appropriate wavelength, from the whole wavelength division multiplexing (WDM) signal, for routing the optical channel taking into account the wavelength continuity and the distinct wavelength constraint. These two functions are commonly designated as the Routing and Wavelength Assignment (RWA) problem [1].

RWA can be seen as an optimization problem whose goal is to minimize the number of wavelengths to be used in a static network scenario. For a dynamic network scenario, the goal of RWA is to minimize the wavelength blocking probability, for a fixed number of wavelengths. Moreover, the RWA problem can be solved as a whole or can be solved by finding independent solutions for the routing and also for the wavelength assignment (WA) problems. Usually this last approach is applied to solve RWA problems. Additionally, we can rely, for both problems, on exact solutions, based on Integer Linear Programming (ILP) solutions, or heuristics solutions, that can give approximate or even the exact solutions with shorter computation times [2]. Some examples of the heuristic algorithms typically used for the routing function are the Dijkstra and *Yen's* k-shortest path algorithms [3]. Also, the most common algorithms used for the WA problem are the First-Fit and Most-Used. Another type of algorithms that can be used for the WA problem are the Graph Coloring algorithms [4].

Graph Coloring algorithms can be applied in many different areas, besides optical networking, like social networking, chemistry, scheduling, satellite navigation, electrical engineering and computer networking [4]. The problem of Graph Coloring, in short, consists in coloring all the graph vertices with the minimum number of colors so that no vertices connected by an edge are given the same color. In this work, we explore three Graph Coloring heuristics for solving the WA problem in real network topologies assuming a static scenario and a full mesh logical topology. In particular, we study the Greedy algorithm, usually used in WA studies, as well as the Degree of Saturation (DSATUR) and Recursive Largest First (RLF) algorithms [4], [5]. These last two algorithms, to the best of our knowledge, have not yet been applied to WA problems. Besides analyzing and comparing the performance of these Graph Coloring algorithms, using real network topologies, such as COST 239 and CONUS networks, we also compared their performance in terms of the number of wavelengths with the traditional First-Fit and Most Used algorithms. Lastly, we assess and compare the computation time between all the studied algorithms for the various network topologies.

The reminder of the paper is organized as follows. In Section II, the three Graph Coloring algorithms, Greedy, DSATUR and RLF heuristics are explained and their pseudocode are detailed. Also, a validation and a comparative performance study between the algorithms is done in this section. In Section III, the developed planning tool is explained and the real network topologies studied are presented, together with the main features of the network physical and logical topologies. In Section IV, the results are discussed in terms of the estimated number of wavelengths, as well as in terms of computation time required by each one of the algorithms. Finally, the conclusions are drawn in Section V.

II. Graph Coloring Heuristics

In this section, the pseudocodes of the three Graph Coloring algorithms, Greedy, DSATUR and RLF, are explained. A

comparative performance study between these heuristics is performed and validated.

A. Greedy Algorithm

The Greedy algorithm is perhaps the most used algorithm for coloring graphs and consists in coloring the vertices of the graph one by one according to an arbitrary order [4], for example, placing the vertices in descending order of degree (the degree is the number of incident edges at each vertex). Then, the algorithm assigns each vertex to the first available color. The pseudocode of this algorithm is given in Figure 1.

```
GREEDY (S ← S_1; X ←V)
(1)  for i ← 1 to |X| do
(2)      for j ← 1 to |S|
(3)          if (S_j ∪ {v_i}) is an independent set then
(4)              S_j ← S_j ∪ {v_i}
(5)              break
(6)          else j ← j +1
(7)          if  j > |S|  then
(8)              S_j ← {v_i}
(9)              S ← S ∪ S_j
```

Fig. 1. Pseudocode of Greedy algorithm.

The pseudocode starts by creating the set X that has all the vertices of the graph in a particular order, e.g., descending. The pseudocode also defines the set S which represents the list of colors associated with the vertices of X. Initially S = S_1, which means that S_1 is the first available color to be assigned to a vertex. As the algorithm takes a vertex from the set X, it first tries to assign a color from the set S to that vertex. In each iteration, the algorithm takes each vertex v_i of the set X, and checks if v_i can be assigned to the color S_j, i.e., the algorithm checks if the color S_j was not assigned to an adjacent vertex of v_i (i.e. the vertices associated with color S_j are an independent set of v_i). If the condition is true, then the color S_j is assigned to the vertex v_i and the process moves on to consider the next vertex. Otherwise, if the colors from the set S can not be assigned to that vertex, the algorithm assigns a new color to the vertex v_i. Lines 7, 8 and 9 of the pseudocode represent the creation of a new color and the assignment of that color to vertex v_i, as well as its addition to the set S. The algorithm ends when all vertices of set X have been assigned to a color.

B. DSATUR Algorithn

The DSATUR algorithm is very similar to the Greedy algorithm, since it takes each vertex one by one according to some ordering and then assigns the appropriate color to the vertex. The difference between these two algorithms is on the ordering in which the vertices are colored. In the Greedy algorithm, the vertex ordering is decided before any coloring, whereas, in the DSATUR algorithm, the choice of the next vertex to be colored is decided heuristically based on the characteristics of the current coloring of the graph [4]. In the DSATUR algorithm, the choice of the next vertex to be colored is based primarily on the saturation degree of the vertices. The degree of saturation of an uncolored vertex is the number of different colors existing in its adjacent vertices [4]. The pseudocode of this algorithm is given in Figure 2.

```
DSATUR (S ← S_1; X ←V)
(1)  for i ← 1 to |X| do
(2)      Choose v ∈ X
(3)      for j ← 1 to |S|
(4)          if (S_j ∪ {v}) is an independent set then
(5)              S_j ← S_j ∪ {v}
(6)              break
(7)          else j ← j +1
(8)          if  j > |S|  then
(9)              S_j ← {v}
(10)             S ← S ∪ S_j
(11)     X ← X - {v}
```

Fig. 2. Pseudocode of DSATUR algorithm.

The pseudocode of the DSATUR algorithm (Fig. 2) is very similar to the Greedy algorithm pseudocode (Fig. 1). The main difference between these two algorithms lies in lines 2 and 11 of the pseudocode given in Fig. 2. In each iteration of the algorithm, the next vertex to be colored is selected from the set X (line 2) and a color from the set S is assigned to that vertex, similar to the Greedy algorithm. However, in the DSATUR algorithm, the next vertex to be colored is chosen as the vertex in X that has the maximal saturation degree. If there is more than one vertex with the maximal saturation degree, the one with the highest degree is chosen from these set of vertices. The first vertex chosen to be colored is the vertex with the highest degree. Once the vertex is colored, it is removed from the set X (line 11). The algorithm ends when X=∅, which means that all vertices have been assigned to a color of the set S.

C. RLF Algorithm

The RLF algorithm follows a slightly different strategy regarding the Graph Coloring in comparison with the previous algorithms, which have a similar procedure [4]. The RLF algorithm consists in coloring a graph with one color at a time, as opposed to one vertex at a time. At each step, the algorithm applies heuristic methods to identify an independent set of vertices in the graph (i.e. non-adjacent vertices), which are then, associated with the same color. This independent set of vertices is then removed from the graph, and the process is repeated in the resulting smaller subgraphs. The pseudocode of this algorithm is given in Figure 3.

The pseudocode begins by defining four sets. The set X that contains the initially uncolored vertices, the set Z that will aggregate throughout the algorithm the selected vertices in each cycle to be later assigned to a color of the set S, the set S that is responsible for assigning a color S_k to the vertices of Z and the set Y that will contain the uncolored vertices that

cannot be feasible assigned to color S_k. At the beginning of the pseudocode execution X=V, Z=$\emptyset$, S= $\emptyset$ and Y= $\emptyset$.

```
RLF (S ← ∅ ; X ← V; Y ← ∅; Z ← ∅)
(1)  for i ← 1 to |X| do
(2)     for j ← 1 to |X| do
(3)        while j ≠ ∅ do
(4)           Choose v ∈ X
(5)           Z ← {v}
(6)           Y ← Y ∪ Γ_X{v}
(7)           X ← X − (Y ∪ {v})
(8)     X ← Y
(9)     Y ← ∅
(10)    for k ← |S| to |Z| do
(11)       S_k ← S_k ∪ {v}
(12)       S ← S ∪ S_k
(13)    k ← k + 1
```

Fig. 3. Pseudocode of RLF algorithm.

In each outer loop, a color is created. Lines 2-7 are responsible for selecting the vertex to be colored. The pseudocode begins by selecting a vertex v of X (line 4), which is added to the set Z (line 5). All vertices adjacent to v, represented by $\Gamma_X\{v\}$, are then transferred to the set Y (line 6), since they cannot be assigned the same color as v. Finally, on line 7, both the vertex v and its adjacent vertices are removed from the set X, since they are not considered candidates for the assignment of the same color as v. As soon as X=$\emptyset$, no more vertices can be added to the current color. Therefore, in line 8, all the vertices of the set of non-colored vertices Y are moved to the set X, and then in line 9, the set Y is emptied.

In lines 10-12 the assignment of the color selected by the outer loop, S_k, to all vertices that are in set Z and have no color assigned is done. As soon as this color is assigned to these vertices, the color is added to set S. Then, a new color (line 13) is created and the algorithm repeats the loop again. The algorithm ends when both sets X and Y are empty, which means that all vertices have been colored and are in set S.

The process of selecting the next $v \in X$ on line 4, follows a similar rationale to the DSATUR algorithm. The first vertex to be chosen for the assignment of each color is the vertex in X that has the highest degree. The remaining vertices v to be assigned to the same color are selected as the vertices in X that have the highest degree in the subgraph defined by Y$\cup v$.

D. Graph Coloring Heuristics Performance

After the implementation of these algorithms in the software tool developed in this work, the performance of the algorithms was analyzed with random generation of graphs. A random graph, denoted by $G_{n,p}$, is a graph comprising n vertices, where each pair of vertices is adjacent with probability p [4]. In this work, the parameter p of graph $G_{n,p}$ is obtained through the average of the degrees of each vertex, given by,

$$p = \frac{\sum_{i=1}^{n} \frac{degree_i}{n-1}}{n} \quad (1)$$

where $degree_i$ represents the degree of the vertex i. When p=0, it means that all vertices are non-adjacent, and when p=1, all vertices of the graph are adjacent.

Figure 4 shows the average number of colors as a function of p obtained for n=100, for the Greedy, DSATUR and RLF algorithms. In Figure 4, an upper bound is also represented, which is defined as the highest degree of a vertex in the graph to be colored plus one [1]. For each value of p, 50 random graphs are generated. Then, an average is performed to obtain the average number of colors generated by these 50 random graphs realizations. In the Greedy algorithm, the vertices are ordered considering the descending order of degree.

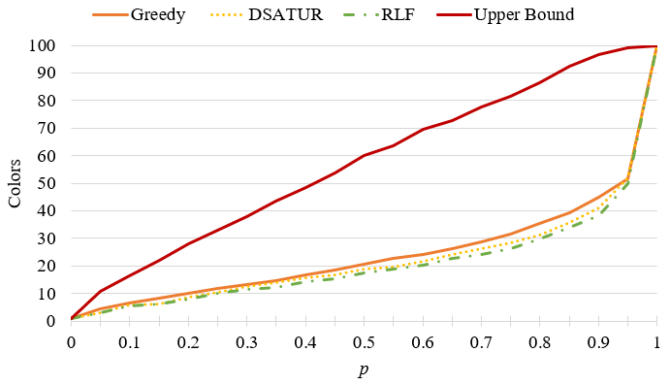

Fig. 4. Average number of colors produced by the Greedy (descending order), DSATUR and RLF algorithms for a network with 100 vertices (n=100).

Observing Figure 4, we verify that for values of p close to 1, the number of colors predicted by the different algorithms tends to become similar, reaching the maximum limit of colors i.e., n=100 meaning that all vertices are adjacent to each other. We also observe that, the Greedy algorithm is the algorithm that produces the worst results, i.e., it needs to assign more colors, while the RLF algorithm generates the best solutions across the whole set, assigning a lower number of colors. The DSATUR algorithm produces solutions with fewer colors than the Greedy algorithm and slightly more colors than the RLF algorithm. In Figure 4, it is also shown that the results obtained with the upper bound are quite different from the ones obtained with the three Graph Coloring algorithms studied. From the simulations that we have performed for various values of n, we observe that the higher the value of n, the nearer the number of wavelengths estimated by the three Graph Coloring algorithms tends to be.

Analyzing in more detail the solutions produced in Figure 4, for p<=0.2, the Greedy algorithm achieves about 10.2 colors, the DSATUR algorithm produces approximately 8.7 colors and the RLF algorithm generates 8.04 colors. For p=0.5, the Greedy algorithm produces around 20.5 colors, the DSATUR algorithm generates about 18.7 colors and the RLF algorithm generates only 17.5 colors. For this value of p, the upper bound on the number of colors is as high as 60 (almost the triple obtained with the analyzed algorithms). These findings were also obtained in [4], which allows validating our implementation

of the three Graph Coloring algorithms that are going to used in Section IV for planning real optical networks.

III. Developed Planning Tool

In this section, we explain the main building blocks of our planning tool with special focus on the implementation of the Graph Coloring algorithms. Then, we present the real networks studied in this work, and characterize their physical topologies features and also some of their logical topology features.

The main building blocks of our planning tool are represented in the flowchart of Figure 5. As shown in Figure 5, our planning tool consists on 6 sub-problems:

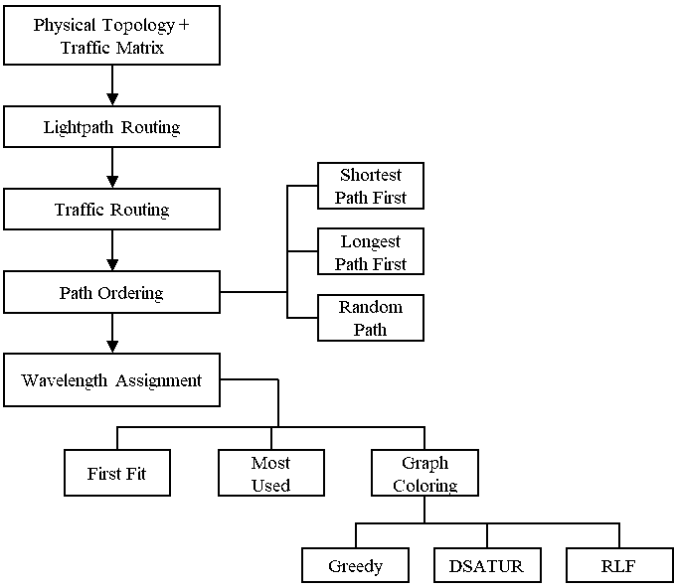

Fig. 5. RWA sub-problems.

1) *Physical Topology and Traffic Matrix*: the network physical topology, as well as, the logical topology (characterized by the traffic matrix) are defined.
2) *Lightpath Routing*: the lightpaths are routed over a physical topology using the *Yen's* k-shortest path algorithm.
3) *Traffic Routing*: after the path is computed and selected, the different traffic units are routed and assigned between the source and destination nodes through the logical topology.
4) *Path Ordering*: before assigning the wavelengths to the lightpaths obtained through the routing algorithm, it is necessary to order these lightpaths. The criteria used to order the paths can be shortest path first, longest path first, and random path. Different metrics can be used to establish the paths order, for example, the distance in kilometers or the number of hops.
5) *Wavelength Assignment*: after the lightpaths are ordered, the wavelengths are assigned accordingly to a given algorithm. Graph Coloring heuristics such as Greedy, DSATUR and RLF, are going to be used in the RWA planning tool. For these heuristics, step 4 is not required.

Before using the Graph Coloring algorithms, the path graph *G(W,P)* must be found. This graph is obtained from the graph *G(V,E)* that represents the physical topology. The vertices of *G(W,P)* are the optical paths $W = (w_1, w_2, w_3, \ldots, w_M)$ and *P* is the set of links between these vertices [1]. These links are established between one or more vertices (i.e. paths) that share one or more physical links. After obtaining the graph *G(W,P)*, the vertices can be colored considering the three Graph Coloring algorithms studied in this work and explained in Section II. The number of colors obtained corresponds to number of wavelengths needed for solving the RWA problem.

The real networks used in this work are: COST 239 [6], [7], NSFNET [8], [9], UBN [10], two variations of the CONUS network [11], [12], one with 30 nodes and other with 60 nodes and two variations of the network generated by the GT-ITM tool in [13], one with 27 nodes and other with 34 nodes.

The GT-ITM (Georgia Tech Internet Topology Modeler) tool generates pseudo-random network topologies on which researchers can perform their analyses [14]. The nodes of the topologies generated are organized in clusters that interconnect with each other by few interconnection links [14]. Figure 6 represents the network with 34 nodes generated by this tool.

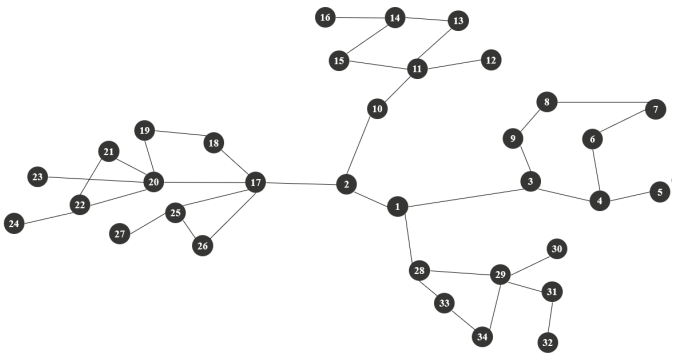

Fig. 6. Physical topology of the network with 34 nodes generated by the GT-ITM tool.

Some of the parameters of the physical topology of the real networks studied in this work are given in Table I.

TABLE I
PHYSICAL TOPOLOGY PARAMETERS.

Network	Node	Link	Average Node Degree	Variance Node Degree
COST 239	11	26	4.7	0.4
NSFNET	14	21	3.0	0.3
UBN	24	43	3.6	0.9
CONUS 30	30	36	2.4	0.4
CONUS 60	60	79	2.6	0.5
GT-ITM 27	27	36	2.7	2.0
GT-ITM 34	34	44	2.6	1.8

The average node degree presented in Table I is defined by [15]:

$$\overline{d} = \frac{\sum_{i=1}^{N} degree_i}{N} \quad (2)$$

where *N* is the total number of the nodes of the network. The variance node degree, in Table I, measures the regularity of

the network, i.e., how similar the nodes of the network are in terms of the number of connections, and is defined as [15]:

$$\sigma^2 = \frac{\sum_{i=1}^{N} (degree_i - \overline{d})^2}{N-1} \quad (3)$$

When the variance node degree is zero, it means that all network nodes have the same node degree and the same number of incoming/outcoming connections, as it happens for example in ring networks. As the value of the variance increases, it means that the similarity of the network is reduced.

In Table II, two parameters of the network logical topology used in the context of the Graph Coloring algorithms, the total number of paths of the graph *G(W,P)* and the parameter *p* defined in equation (1) for several real networks are presented.

TABLE II
TOTAL NUMBER OF PATHS AND *p* PARAMETER.

Networks	Total Paths	*p*
COST 239	55	0.10
NSFNET	91	0.25
UBN	276	0.23
CONUS 30	435	0.38
CONUS 60	1770	0.32
GT-ITM 27	351	0.56
GT-ITM 34	561	0.59

From the analysis of Table II, we observe that the parameter *p* of the networks generated by the GT-ITM tool is *p* = 0.56 with 27 nodes and *p* = 0.59 with 34 nodes. These networks are divided into 3 and 4 domains, with more than 7 nodes inside each domain, which are interconnected with each other by two central nodes. This centralization increases the number of paths that pass through the physical links connected to the central node, increasing the *p* parameter of the network compared to the remaining real networks. As can be seen in Figure 4, when *p* is greater than 0.5, the DSATUR and RLF algorithms tend to give even better results than the Greedy algorithm in comparison with $p<=0.5$.

IV. RESULTS AND DISCUSSION

In this section, we present the number of wavelengths as well as the computation times obtained by the planning tool developed for the real networks presented in Section III. Besides the three Graph Coloring algorithms we also considered the First Fit and Most Used algorithms. The results presented in this section assume a full mesh logical topology with one traffic unit in each path.

Tables III and IV present the total number of wavelengths obtained by each of the Graph Coloring heuristics and the First Fit and Most Used, for the real networks discussed in Section III. Before applying these algorithms, the network demands are sorted according to three different orderings: shortest path first (SPF), longest path first (LPF) and random path (RP). The values of the random path ordering strategy represent the average obtained after 10 simulation runs.

By analyzing Tables III and IV, we verify that the results obtained by the First Fit algorithm with the LPF ordering strategy, are quite similar to the results achieved by the Graph Coloring heuristics, in particular with the Greedy algorithm with descending order of degree, where exactly the same number of wavelengths is obtained for all studied networks.

TABLE III
NUMBER OF WAVELENGTHS OBTAINED BY FIRST FIT AND MOST USED ALGORITHMS FOR THE NETWORKS DESCRIBED IN SECTION III.

Networks	First Fit			Most Used		
	SPF	LPF	RP	SPF	LPF	RP
COST 239	8	8	8.2	9	8	8.3
NSFNET	24	24	24.0	25	24	24.2
UBN	70	64	66.0	71	64	66.7
CONUS 30	134	123	123.5	135	124	123.8
CONUS 60	550	543	543.5	551	543	543.7
GT-ITM 27	229	221	221.3	229	221	221.1
GT-ITM 34	356	347	347.1	356	347	347.2

TABLE IV
NUMBER OF WAVELENGTHS OBTAINED BY GREEDY, DSATUR AND RLF ALGORITHMS FOR THE NETWORKS DESCRIBED IN SECTION III.

Networks	Greedy	DSATUR	RLF
COST 239	8	8	8
NSFNET	24	24	24
UBN	64	64	64
CONUS 30	123	123	119
CONUS 60	543	543	543
GT-ITM 27	221	217	216
GT-ITM 34	347	342	338

Regarding the behavior of the three Graph Coloring algorithms, we observe that the COST239, NSFNET and UBN networks do not present any differences between the numbers of wavelengths obtained. These results were expected, since the parameter *p* of each of these networks is less or equal to 0.25, and as shown in Figure 4, the number of colors is quite similar for networks with such *p*. The same conclusion can be applied to the CONUS 60 network. This network, despite having 1770 nodes, presents a low value of *p* for the number of nodes it has (*p*=0.32). The CONUS 30 network presents a slight difference in the number of wavelengths obtained by the RLF algorithm (119 wavelengths), since it has a higher *p* (*p* = 0.3792) and has a smaller number of vertices compared to the CONUS 60 network. Therefore, for the CONUS 30 network, the RLF algorithm gives better results than the other algorithms. The GT-ITM networks, having a *p* above 0.5, show a clearer difference between the number of wavelengths predicted by the algorithms. For these networks, the RLF algorithm presents the best results reducing the number of assigned wavelengths by 5 for the network with 27 nodes and by 9 for the network with 34 nodes, in comparison with the Greedy algorithm. In this case, the DSATUR also brings advantages, since it assigns less 4 wavelengths than the Greedy algorithm in the network with 27 nodes and less 5 wavelengths in the network with 34 nodes. Therefore, the results for the GT-ITM networks are in agreement with the conclusions taken in

Figure 4, which have shown that, when the p parameter varies between 0.5 and 0.9, the variations between these three Graph Coloring heuristics become more notorious.

Next, the computation time (in seconds) of each one of the wavelength assignment algorithms studied is assessed. The results are shown in Tables V and VI. All the results presented in this work were obtained in a computer with an Intel(TM) core i5 processor at 2.20 GHz and with 8 GB of RAM.

TABLE V
COMPUTATION TIME (IN SECONDS) CONSIDERING FIRST FIT AND MOST USED ALGORITHMS.

Networks	First Fit			Most Used		
	SPF	LPF	RP	SPF	LPF	RP
COST 239	3.9	3.8	3.2	4.0	3.5	3.2
NSFNET	4.7	4.3	3.9	4.9	4.7	3.9
UBN	8.6	7.2	6.5	8.2	7.2	6.5
CONUS 30	8.5	7.2	7.0	8.2	7.4	6.4
CONUS 60	33.3	32.5	31.5	28.9	30.2	30.7
GT-ITM 27	8.5	7.6	6.9	8.8	7.8	6.9
GT-ITM 34	9.2	8.1	7.8	9.6	8.4	8.0

TABLE VI
COMPUTATION TIME (IN SECONDS) CONSIDERING GREEDY, DSATUR AND RLF ALGORITHMS.

Networks	Greedy	DSATUR	RLF
COST 239	4.0	4.2	4.5
NSFNET	4.1	4.7	5.7
UBN	7.2	8.4	9.2
CONUS 30	14.2	16.5	17.5
CONUS 60	162.4	164.4	167.5
GT-ITM 27	8.9	10.7	11.2
GT-ITM 34	16.7	17.4	18.1

The computation time obtained by the First Fit and Most Used algorithms are quite similar to the computation time of the Greedy algorithm for networks up to 25 nodes. For networks above 25 nodes, the First Fit and Most Used algorithms generate solutions quicker than the Greedy algorithm.

Comparing the three Graph Coloring heuristics, the Greedy algorithm, despite producing more colors, generates solutions in a shorter computation time. The RLF algorithm requires more time to generate solutions, but it is the algorithm that assigns fewer colors. The DSATUR algorithm has slightly longer computation time than the Greedy algorithm.

V. CONCLUSIONS

In this work, we have implemented an optical network planning tool that uses three Graph Coloring heuristics to solve the WA problem, the Greedy, the DSATUR and the RLF algorithms.

We have compared the performance of these three Graph Coloring heuristics in some real network topologies, COST 239, NSFNET, UBN, CONUS 30, CONUS 60, GT-ITM 27 and GT-ITM 34 in a static network scenario considering a full mesh logical topology with one unit of traffic in each path.

We have concluded that the RLF algorithm produces the best solutions, minimizing the number of wavelengths used in the network. In addition, for a parameter p above 0.5, it has also been proven that this algorithm and the DSATUR algorithm are advantageous for WA in the planning of optical networks, because they obtain better solutions than the most usual wavelength assignment algorithms, the First Fit and Most Used. One such example where these improvements are evident is in networks whose physical topology is based on several cluster of nodes interconnected by central nodes using few links, e.g. as the networks generated by the GT-ITM tool. For this type of networks, more specifically for a GT-ITM network with 34 nodes, the RLF algorithm reduces the number of assigned wavelengths by 9 and the DSATUR algorithm by 5 compared to the Greedy algorithm. The disadvantage of these two Graph Coloring algorithms is slightly higher computation time they require to return a solution, in comparison to First Fit, Most Used and Greedy algorithms.

ACKNOWLEDGMENT

This work was supported under the project of Instituto de Telecomunicações UIDB/EEA/50008/2020.

REFERENCES

[1] J. Simmons, Optical Network Design and Planning, 2nd edition, New York, USA: Springer, 2014.
[2] R. Ramaswami, K. Sivarajan, and G. Sasaki, Optical Networks: A Practical Perspective, 3rd edition, USA: Morgan Kaufmann, 2010.
[3] H. Zang, J. P. Jue, and B. Mukherjeey, "A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks," *Optical Networks Magazine, SPIE publishers*, vol. 1, pp.47–60, Mar 2000.
[4] R. M. R. Lewis, Algorithms and applications in *A Guide to Graph Coloring*, Switzerland: Springer, 2016.
[5] M. Aslan and N. Baykan. A performance comparison of Graph Coloring algorithms. In *International Conference on Advanced Technology Sciences (ICAT'16)*, pp. 266–273, Sept 2016.
[6] M. Niksirat, S. Mehdi Hashemi, and M. Ghatee, "Branch-and-price algorithm for fuzzy integer programming problems with block angular structure," *Fuzzy Sets Syst.*, vol. 296, pp. 70–96, Aug 2016.
[7] M. C. Sinclair, Minimum Cost Topology Optimisation of the Cost 239 European Optical Network, in *Artificial Neural Nets and Genetic Algorithms*, pp. 26–29, Springer-Verlag, Vienna, 1995.
[8] T. L. LaQuey, NSFNET, in *The User's Directory of Computer Networks*, pp. 247–250, Boston: Digital Press, 1990.
[9] D. L. Mills and H. Braun, The NSFNET Backbone Network, in *Proceedings of the ACM Workshop on Frontiers in Computer Communications Technology*, SIGCOMM '87, pp 191–196, New York, NY, USA, 1987.
[10] E. Biernacka, J. Domzal, and R. Wójcik, "Investigation of dynamic routing and spectrum allocation methods in elastic optical networks," *International Journal of Electronics and Telecommunications*, vol. 63, no 1, pp. 85–92, Feb 2017.
[11] CONUS WDM network topology, http://monarchna.com/topology.html.
[12] T. Zami, B. Lavigne, and M. Bertolini, "How 64 Gbaud optical carriers maximize the capacity in core elastic WDM networks with fewer transponders per Gb/s," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 11, no. 1, pp. A20–A32, 2019.
[13] R. Godoi, P. Hernández, and E. Luque, "Control Evaluation in a LVoD System Based on a Peer-to-Peer Multicast Scheme," *Journal of Computer Science and Technology*, vol. 8, no. 2, pp. 97–103, 2008.
[14] J. R. Eagan, J. Stasko, and E. Zegura, Interacting with Transit-Stub Network Visualizations, GVU Center, College of Computing, Georgia Institute of Technology, Atlanta, pp 1–2, 2005.
[15] C. Fenger, E. Limal, and U. Gliese. Statistical study of the influence of topology on wavelength usage in WDM networks. In *Optical Fiber Communication Conference*, vol. 1, pages 171–173. Baltimore, USA, March 2000.