Advanced Column Generation Decompositions for Optimizing Provisioning Problems in Optical Networks

Julian Enoch

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Science (Computer Science) at

Concordia University

Montréal, Québec, Canada

June 2018

© Julian Enoch, 2018

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Julian Enoch**

Entitled: **Advanced Column Generation Decompositions for Optimizing Provisioning Problems in Optical Networks**

and submitted in partial fulfillment of the requirements for the degree of

Master of Science (Computer Science)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Sudhir Mudur

_____ External Examiner
Dr. Thomas G. Fevens

_____ Examiner
Dr. Lata Narayanan

_____ Supervisor
Dr. Brigitte Jaumard

_____ Administrative Supervisor
Dr. Chadi Assi

Approved by _____
Sudhir Mudur, Chair
Department of Computer Science and Software Engineering

_____ 2018

_____
Amir Asif, Dean
Faculty of Engineering and Computer Science

Abstract

Advanced Column Generation Decompositions for Optimizing Provisioning Problems in Optical Networks

Julian Enoch

With the continued growth of Internet traffic, and the scarcity of the optical spectrum, there is a continuous need to optimize the usage of this resource. In the process of provisioning optical networks, telecommunication operators must deal with combinatorial optimization problems that are NP-complete. One of these problems is the *Routing and Wavelength Allocation* (RWA) which considers the fixed frequency grid, and the *Routing and Spectrum Allocation* (RSA) which is defined for the flexible frequency grid. While the flexible frequency grid paradigm attempted to improve the spectrum usage, the RSA problem has an additional spectrum dimension that makes it harder than the RWA problem.

In this thesis, in continuation of the previous studies, and using the advanced techniques of *Integer Linear Programing*, we propose a *Column Generation* algorithm based on a *Lightpath* decomposition which we implement for both the RWA and the RSA problems. This algorithm proved to be the most efficient so far producing optimal or near optimal solutions, and improving the computation times by two orders of magnitude on average. This algorithm is based on the approach of finding the right decomposition scheme as to be able to solve the *Pricing Problem* in a polynomial time. This approach can be used in other optimization problems.

In addition, we consider the same *Configuration* decomposition as the previous studies, and we propose an algorithm based on *Nested Column Generation*. We implemented this algorithm for both the RSA and the RWA problems, which led to a considerable improvement on the previous algorithms that use the same *Configuration* decomposition. This *Nested Column Generation* approach can be adopted in other optimization problems.

Acknowledgments

I would like to acknowledge the people who:

- Strive to be better by hard work and self-disipline;

- Respect the right of other people to their integrity just as they claim the right to the same;

- Work with responsibility, honesty and fairness toward their organization and collaborators.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

According to the latest report of Cisco [2017], the annual global IP traffic in 2016 was 1.2 Zetta-Byte, and is expected to increase threefold by 2021. This growth is driven in a large proportion by the IP video traffic which would account for 82% of all IP traffic by then. In particular, applications such as Live Internet video, Virtual Reality, Internet gaming will contribute for the largest growth portion. This continuous growth in the demand translates into a continuous need to enhance the design and the engineering of the networking infrastructure and technologies.

The core networks of Internet (called also long-haul networks) are based on optical technology and their links are made of optical fibers. These long-haul networks cover in general entire continents and their links span hundreds if not thousands of kilometers. As a way to increase the capacity, these networks implement Wavelength Division Multiplexing (WDM), where the idea is to transmit data simultaneously at multiple carrier wavelengths (or, equivalently, frequencies) over a fiber. Traditionally, the multiplexing was done according to a Coarse WDM frequency grid (called fixed grid) where the carrier wavelength granularity is 50 GHz. As of 2012, the International Telecommunication Union issued a new recommendation that defines a flexible Dense WDM frequency grid (flexible grid for short) with a finer frequency granularity of 12.5 GHz called a frequency slot [IUT, 2012]. This evolution has allowed the emergence of Elastic Optical Networks.

The main phase in the process of designing an optical network is concerned with traffic provisioning. In all instances, traffic provisioning consists of assigning a routing path, and allocating a frequency range to each traffic demand. In the case of fixed grid, the frequency range corresponds to a wavelength. In this case, the provisioning problem is called the *Routing and Wavelength Allocation* problem. In the case of flexible grid, the frequency range corresponds to a number of contiguous frequency slots. In this case, the provisioning problem is called the *Routing and Spectrum Allocation* problem.

1.2 Objectives and contributions

Although the flexible grid was introduced to allow a more economic use of the optical resources, the fixed grid remains in use in a lot of optical networks for many reasons such as the low cost of the optical devices and the simplicity of the engineering algorithms. Hence, this thesis discusses both RWA and RSA problems.

Among the existing algorithms for these problems, heuristics are in general fast but they do not provide any guarantee on the quality of the solutions. On the other hand, exact algorithms do not scale well, and are usually used on small network instances.

Our objective is to propose exact algorithms with better scalability than the existing ones, as to allow the resolution of real-life network instances. In doing so, we use a large scale optimization technique called *Column Generation*. While using RWA and RSA as study cases, we explore the possibility to maximize the potential of *Column Generation* by finding the ideal decomposition of these problems. In doing so, we designed two algorithms:

(1) The first algorithm employs a *Lightpath* decomposition combined with a *Column Generation* scheme

(2) The second algorithm employs a *Configuration* decomposition combined with a *Nested Column Generation* scheme

Throughout this document, we use the following acronyms to refer to different algorithms:

- L_CG refers to a *Column Generation* algorithm using a *Lightpath* decomposition introduced in this thesis for both RWA and RSA;
- C_NCG refers to a *Nested Column Generation* algorithm using a *Configuration* decomposition introduced in this thesis for RSA;
- W_NCG refers to a *Nested Column Generation* algorithm using a *Configuration* decomposition introduced in this thesis for RWA;
- C_CG refers to a *Column Generation* algorithm using a *Configuration* decomposition introduced by Jaumard and Daryalal [2016] for RSA;
- W_CG refers to a *Column Generation* algorithm using a wavelength *Configuration* decomposition introduced by Jaumard and Daryalal [2017] for RWA;

All the implementations were coded in Java and the *Branch-and-Bound* part was done using CPLEX. Shortest path calculation was done with an open-source Java library JGraphT implementing Dijkstra's algorithm. The results were produced on a 3.6-4.0 GHz 4-core machine with 32 GB of memory.

We were able to publish part of our work, and we are in the process of publishing the remaining part. Therefore in Chapter 2, we present the paper [Enoch and Jaumard, 2018a] which uses a *Lightpath* decomposition to solve the RWA problem. Chapter 3 presents the paper [Enoch and Jaumard, 2018b] which uses a similar decomposition applied to the RSA problem. Chapter 4 introduces the more advanced *Nested Column Generation* approach applied to the RWA problem. Chapter 5 shows the results obtained when applying *Nested Column Generation* to the RSA problem.

Next, we factored out the literature review of the two optimization problems discussed in this thesis.

1.3 Literature Review

1.3.1 Routing and Wavelength Assignment

While we are interested in the so-called max RWA problem in this study, many studies consider the min RWA. In the max RWA problem, the objective is to maximize the number of granted requests, i.e., it corresponds to a provisioning problem. On the other hand, in the min RWA problem, the objective is to minimize the number of required wavelengths in order to grant all the requests, hence it corresponds to a dimensioning problem.

The max RWA problem has been extensively studied in various flavours. In order to set the stage for the present work, we provide the key references on the past works that have studied the RWA problem with the objective to maximize the GoS, using Integer Linear Programming, in an attempt to solve it exactly, or nearly exactly with an estimate of the accuracy. We also provide few references related to the min RWA as it is a very similar problem.

The first class of studies used link based models. Such models are considered to be compact for they have a polynomial number of variables and constraints. Yet, none of them is scalable for large network instances. Jaumard et al. [2007] provide an extensive review of the different proposed link models.

The second class of studies considered path based models, see, e.g., [Saad and Luo, 2002]. Therein, the authors used a Lagrangian decomposition to show that the LP relaxation can be solved in polynomial time, but the ILP itself was solved using a pre-computed subset of paths, which makes the model, and therefore the algorithm non-exact. Other path formulations were explored in the context of the min RWA problem, see, e.g., [Christodoulopoulos et al., 2010; Liu and Rouskas, 2012].

The need for scalable and exact ILP models triggered the introduction of advanced models that can make use of decomposition techniques. While providing a review of the existing CG algorithms, Jaumard et al. [2009] hinted to an exact column generation (CG) model and CG-ILP algorithm with a polynomial-time pricing problem (i.e., generator of lightpaths). However, the authors refrained from pursuing this direction, arguing that it suffers from a wavelength symmetry issue and an optimal LP relaxation that is not better than the one of the link formulations. Indeed,

an integer linear program (ILP) is symmetric if its variables can be permuted without changing the structure of the problem. Symmetry is usually viewed as the prelude of great difficulties for solving exactly an ILP with traditional branch-and-bound, branch-and-cut or branch-and-price algorithms [Margot, 2010]. Still, we decided to revisit the lightpath decomposition model following its good performances for the RSA problem [Enoch and Jaumard, 2018b].

In the prospect of improving the characteristics of the path model, many studies considered new ILP formulations with coarse variables, where each variable is associated with wavelength *Configuration*, i.e., a set of lightpaths all routed with the same wavelength. This led to the so-called Independent Set (IS) (also called Independent Routing *Configuration* decomposition (IRC)) models, using an underlying graph in which each node is associated with a path, and two nodes are connected if their associated paths share a link. Finding a wavelength *Configuration* is equivalent to identifying an independent set (also called stable set). Ramaswami and Sivarajan [1995] introduced a first *Independent Set* (IS) model and solved its linear relaxation to obtain an upper bound while explicitly enumerating the exponential number of variables. Lee et al. [2000] introduced a slightly more compact model than Ramaswami and Sivarajan [1995], resulting in the so-called Independent Routing *Configuration* decomposition model and solved it using a column generation algorithm where the pricing problem is written as a weighted independent set problem, with an exponential number of variables. Two years later, Lee et al. [2002] proposed to solve the same pricing problem using an embedded branch-and-price technique, which is quite computationally expensive. Jaumard et al. [2009] then proposed to model the pricing problem as a multi-flow problem using a link formulation, thus overcoming the issue of the exponential number of variables in the pricing problem. More recently, Jaumard and Daryalal [2017] improved the scalability of Jaumard et al. [2009] combining the link formulation of the pricing problem with a path one in order to speed up the generation of wavelength *Configurations*. The resulting algorithm is able to solve efficiently data instances with up to 90 nodes (ATT network) and up to 150 wavelengths (USA, Germany and NTT networks).

While it was proved by Jaumard et al. [2009] that a wavelength decomposition ILP model could theoretically provide a better LP relaxation bound than the lightpath decomposition, it is usually not the case on classical data instances. Therefore, while providing a better LP bound in general,

additional complexity is put on the pricing problems, and impact the computational times as will be seen in Section 2.4. In the current work, we revisit the lightpath decomposition, and compare its performances on the same instances as [Jaumard and Daryalal, 2017]. We show that despite its wavelength symmetry, we are able to produce significantly better results than the latest Maximum Independent Routing *Configuration* decomposition proposed model and algorithm of Jaumard and Daryalal [2017] both in terms of solution quality and algorithm efficiency.

1.3.2 Routing and Spectrum Assignment

Various mathematical models have already been proposed for solving the RSA problem, including several column generation models, however with different decomposition schemes, and different solution processes. All classical Integer Linear Programming (ILP) models are not scalable and can only solve very small data instances.

We now review the column generation models. Ruiz et al. [2013b] proposed a first column generation model, with the minimization of the number of denied demands and the amount of unserved bit-rate. They are able to solve data instances with up to 96 slots, and an overall demand distributed over a set of 180 node pairs in the Spain network (21 nodes). Klinkowski et al. [2014] improved the formulation of Ruiz et al. [2013b] with the use of valid inequalities (cuts), but did not go significantly farther than in [Ruiz et al., 2013b]. Klinkowski and Walkowiak [2015] reformulated the RSA problem as a mixed-integer program and solved it using a branch-and-price algorithm. In order to enhance the performance of their algorithm, a simulated annealing-based heuristic was added. In a recent attempt to exactly solve large-size instances of RSA problem, Klinkowski et al. [2016] proposed a branch-and-price algorithm. However, the resulting algorithm is not an exact algorithm and the LP (Linear Programming) value is not a valid bound to assess the quality of the ILP solutions, as the authors use pre-computed lightpaths, and consequently did not consider, explicitly or implicitly, all possible lightpaths.

In this study, we propose an ε-optimal solution scheme, i.e., an algorithm that outputs an ε-solution. The addition of a branch-and-price algorithm would make it possible to reach an optimal solution.

Chapter 2

Enhanced RWA Exact Solution with a New Lightpath Decomposition Algorithm

Enoch and Jaumard [2018a]. Enhanced RWA exact solution with a new lightpath decomposition algorithm. In IEEE International Conference on Computing, Networking and Communications - ICNC.

2.1 Introduction

The Routing and Wavelength Assignment (RWA) problem is concerned with the provisioning of fixed-grid Wavelength Division Multiplexing (WDM) optical networks.

Despite the recent introduction of the flexible WDM grid, the RWA problem is still of great interest for many reasons [Saleh and Simmons, 2011, 2012; Simmons, 2017; Muciaccia and Passaro, 2017]. While flexible grids are widely investigated and moved forward, they add significant network management cost and complexity, e.g., the requirement of many bandwidth-variable transponders together with the need to track how the spectrum is sliced up on each fiber [Simmons, 2017]. In addition, the advantage of the flexible grid over the fixed grid is very much dependent on the granularity of the demands in terms of bandwidth: while it may offer up to a 1.5 effective capacity

multiplier for heavy granularities (400 Gbs and higher) [Simmons, 2017], it offers no significant advantages for low granularities in view of its additional network costs and complexities.

We revisit the static RWA problem that consists of provisioning a set of demands over a set of lightpaths subject to the wavelength continuity constraint (i.e., an all-optical wavelength channel between two nodes, which may span more than one fiber link). We consider the objective of maximizing the Grade-of-Service (GoS), which is suitable for the provisioning phase of network management, as opposed to the objective of minimizing the number of required wavelengths, which is involved in the dimensioning phase. These two objectives lead to the so-called max-RWA and min-RWA problems, respectively.

Under these assumptions, given a WDM network and a traffic demand matrix, the RWA problem consists of assigning a routing path and one wavelength to each unit of traffic connection as to maximize the number of granted connections, while avoiding any wavelength clash on a link and a wavelength simultaneously. This problem has been extensively studied since its inception [Zang et al., 2000; Jaumard et al., 2007], yet, with the unrelentingly growing traffic demand on Internet, there is a need to further improve the solution of large and practical network instances.

Among the previous studies of the RWA problem, the heuristics are of practical use for their scalability, but they do not provide any information on the quality of their solution, see, e.g., [Jaumard et al., 2006; Noronha and Ribeiro, 2006; Duhamel et al., 2016]. On the other hand, algorithms for solving exact models, using ILP (Integer Linear Programming) solvers, produce in general near optimal solutions, with an estimate of the accuracy of the solution. In this work, we propose an exact lighpath decomposition model and an ε-optimal algorithm. We aim to improve upon the quality and the efficiency of the previous existing studies. In doing so, the proposed lightpath decomposition model will be solved using column generation (CG) techniques with a polynomial-time lightpath generator. This contrasts with previously proposed decomposition schemes, e.g., with wavelength *Configurations* [Jaumard and Daryalal, 2017], in which symmetry was eliminated at the expense of a non-polynomial algorithm for generating wavelength *Configurations*.

The paper is organized as follows. We define the notations and state the lightpath decomposition ILP model in Section 2.2. Next, in Section 2.3, we describe precisely the L_CG algorithm to solve it efficiently. Performances of the L_CG algorithm is extensively evaluated in Section 2.4, and

shown to be significantly faster than the best currently available RWA algorithms. Conclusions are drawn in the last section.

2.2 L_CG: Lightpath Decomposition Model

We assume a transparent All-Optical Network (AON), and we consider the general case of asymmetrical traffic.

We propose a lightpath-based model similar to the one introduced in [Lee et al., 2002]. It differs from the path formulation studies, e.g., [Jaumard et al., 2009; Liu and Rouskas, 2012], in that the variables are indexed by both a path and a wavelength.

Consider a WDM optical network represented by a directed graph $G = (V, L)$ where V represents the nodes of the network, and L is the set of links representing the directional optical fibers. The optical spectrum is divided into a set of wavelengths with equal capacity, denoted by Λ. The traffic can be viewed as a $|V| \times |V|$ matrix where each element D_{sd} represents the number of wavelength units from a source node v_s to a destination node v_d. Let $\mathcal{SD}$ be the set of node-pairs with traffic.

For a given node-pair $(v_s, v_d) \in SD$, we define a *Lightpath* π as the routing path of a traffic connection from v_s to v_d on a given wavelength $\lambda \in \Lambda$. Let Π be the set of all lightpaths π for all pairs in $\mathcal{SD}$. The decision variable x_π determines whether lightpath π is retained in the optimal solution. We define the following parameters:

- a_π^{sd}: indicates if lightpath π serves a unit connection $(v_s, v_d) \in \mathcal{SD}$.
- $b_{\pi\ell}^{\lambda}$: indicates if lightpath π is routed on wavelength $\lambda \in \Lambda$ and a path that contains link $\ell \in L$.

The ILP model reads as follows:

$$z \quad = \quad \max \quad \sum_{\pi \in \Pi} x_\pi \qquad \text{(number of granted requests)} \tag{2.1}$$

subject to:

$$\sum_{\pi \in \Pi} a_{\pi}^{sd} x_{\pi} \leq D_{sd} \qquad (v_s, v_d) \in \mathcal{SD} \tag{2.2}$$

$$\sum_{\pi \in \Pi} b_{\pi \ell}^{\lambda} x_{\pi} \leq 1 \qquad \ell \in L, \lambda \in \Lambda \tag{2.3}$$

$$x_{\pi} \in \{0, 1\} \qquad \pi \in \Pi. \tag{2.4}$$

Constraints (2.2) ensure that the number of connections from a source to a destination does not exceed the actual demand. Constraints (2.3) mean that each wavelength on each link is used by at most one connection.

Note that the above model remains valid even in a network with multi-fiber links. It suffices to build a graph, which is a multi-graph, where each link is associated with one fiber link.

2.3 Solution Design

After reviewing very briefly the characteristics of a column generation algorithm, we describe in detail the L_CG algorithm we used for solving the lightpath decomposition model that was proposed in the previous section.

2.3.1 Column Generation Framework

Given the exponential number of columns - lightpaths - in the proposed model, we resort to the *Column Generation* method [Gilmore and Gomory, 1961] to solve its linear relaxation, and then derive an integer solution. This technique consists of decomposing the original problem into a Restricted Master Problem - RMP - (i.e., model (2.1) - (2.4) with a very restricted number of variables) and one or several pricing problems - PPs. In the particular case of model (2.1) - (2.4), we will show in the next section that the pricing problem can be decomposed into $|V|^2 \times |\Lambda|$ independent smaller pricing problems, each denoted by $\text{PP}_{sd,\lambda}$. The RMP and the PP(s) are solved alternately. Solving the RMP consists in selecting the best lightpaths, while solving the PPs allows the generation of improving potential lightpaths, i.e., lightpaths such that, if added to the current

RMP, improve the optimal value of its linear relaxation. The process continues until the optimality condition is satisfied, that is, all the so-called reduced costs that define the objective function of the pricing problems are negative, see [Chvatal, 1983] if not familiar with linear programming concepts.

Once the optimal solution of the linear relaxation has been reached, we derive an integer solution by solving exactly the last RMP with integrality requirements. This leads to an integer solution, denoted by $\tilde{z}_{\text{ILP}}$. This last value is not necessarily the optimal value, but we can easily compute its accuracy ε. As we will see with the numerical results in Section 2.4, the accuracy gap is small, so we decided not to resort to a solution scheme, such as, e.g., a branch-and-price algorithm [Barnhart et al., 1998], in order to compute an integer optimal solution. Indeed, while a branch-and-price algorithm would allow to compute an optimal integer solution, it would require much more computational resources, which is not justified in view of the very good accuracy of the solutions in practice (see Section 2.4).

The accuracy (ε) of the integer solution, derived by solving exactly the ILP model associated with the last RMP, is defined as follows:

$$\varepsilon = \frac{z^{\star}_{\text{LP}} - \tilde{z}_{\text{ILP}}}{\tilde{z}_{\text{ILP}}}, \tag{2.5}$$

where $z^{\star}_{\text{LP}}$ and $\tilde{z}_{\text{ILP}}$ denote the optimal LP value and the optimal ILP value of the last RMP, respectively. The solution process is illustrated in the flowchart of Figure 2.1.

Before any LP re-optimization, we solve the whole series of pricing problems, where each pricing problem $\text{PP}_{sd,\lambda}$ is associated with one node pair and one wavelength.

After each re-optimization of the RMP, we eliminate the variables that are not in the basis, as described in [Chvatal, 1983]. While we may need to regenerate some of the eliminated columns (but they are not expensive to generate), it helps to maintain a reasonable number of variables, indeed equal to the number of constraints. Moreover, before deriving an ILP solution, i.e., solving exactly the last RMP subject to integer requirements, we remove all variables which are equal to zero, i.e., not contributing to the value of $z^{\star}_{\text{LP}}$.

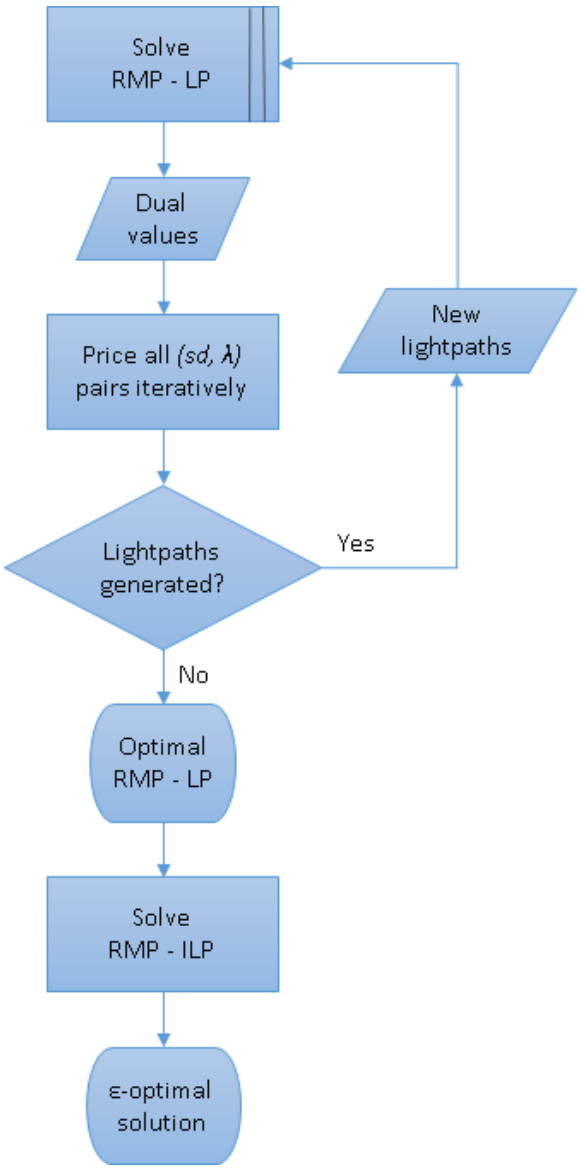


Figure 2.1: Flow chart of the L_CG Algorithm for RWA

2.3.2 Decomposition of the Pricing Problem

We, next, show that the column generation decomposition involves a set of independent elementary pricing problems, denoted by $\text{PP}_{sd,\lambda}$, one for each node pair and wavelength.

Let $\mu_{sd} \geq 0$ and $\mu_{\ell}^{\lambda} \geq 0$ be the values of the dual variables associated with constraints (2.2) and (2.3), respectively. For a given vertex $v \in V$, we denote by $\omega^{-}(v)$ (respectively $\omega^{+}(v)$) the set of ingoing (respectively outgoing) links. The objective of the pricing problem PP is defined by the reduced value [Chvatal, 1983] and can be written as follows.

$$\text{[PP]} \qquad \max \quad 1 - \sum_{(v_s,v_d)\in\mathcal{SD}} \mu_{sd}\, a_{\pi}^{sd} - \sum_{\ell\in L}\sum_{\lambda\in\Lambda} \mu_{\ell}^{\lambda}\, b_{\pi\ell}^{\lambda}. \tag{2.6}$$

subject to:

$$\sum_{(v_s,v_d)\in SD} a_{\pi}^{sd} = 1 \tag{2.7}$$

$$\sum_{\ell\in\omega^{-}(v_s)} b_{\pi\ell}^{\lambda} = \sum_{\ell\in\omega^{+}(v_d)} b_{\pi\ell}^{\lambda} = a_{\pi}^{sd} \qquad \lambda\in\Lambda, (v_s,v_d)\in SD \tag{2.8}$$

$$\sum_{\ell\in\omega^{-}(v)} b_{\pi\ell}^{\lambda} = \sum_{\ell\in\omega^{+}(v)} b_{\pi\ell}^{\lambda} \qquad \lambda\in\Lambda, (v_s,v_d)\in SD, v\in V\setminus\{v_s,v_d\} \tag{2.9}$$

$$a_{\pi}^{sd} \in \{0,1\} \qquad (v_s,v_d)\in SD \tag{2.10}$$

$$b_{\pi\ell}^{\lambda} \in \{0,1\} \qquad \lambda\in\Lambda, \ell\in L. \tag{2.11}$$

The set of constraints corresponds to flow constraints in order to determine a path for each (v_s, v_d) and each λ.

The objective of the *Pricing Problem* is to find a lightpath that optimizes the reduced value. Instead of generating only one optimal lightpath, we will compute all feasible lightpaths for which the reduced value is positive, in a multi-column generation scheme.

Given the wavelength continuity constraint, i.e., each lightpath must be routed on only one wavelength, it is possible to fix the wavelength and solve multiple smaller *Pricing Problems*, one for each wavelength, sequentially.

In addition, a lightpath is defined for one node-pair only. Therefore, we can decompose the

Pricing Problem further into sequential smaller problems, by fixing the node-pair at each iteration (i.e., by setting a_π^{sd} to 1 for the current node-pair and to 0 for all the others).

Thus, rather than solving the pricing problem as a whole using an ILP solver, it is of interest to decompose it into $|\mathcal{SD}| \times |\Lambda|$ independent smaller pricing problems $\text{PP}_{sd,\lambda}$, whose reduced value is defined as follows:

$$[\text{PP}_{sd,\lambda}] \qquad \max \quad 1 - \mu_{sd} - \sum_{\ell \in L} \mu_\ell^\lambda \, b_{\pi\ell}^\lambda. \tag{2.12}$$

For a given (v_s, v_d), the first terms of the reduced value, i.e., $1 - \mu_{sd}$ is a constant. Hence, the maximization of the reduced value is equivalent to:

$$[\text{PP}_{sd,\lambda}] \qquad \min \quad \sum_{\ell \in L} \mu_\ell^\lambda \, b_{\pi\ell}^\lambda. \tag{2.13}$$

Taking into account that the set of constraints amounts to a set of flow constraints for a given node pair and wavelength, each pricing problem $\text{PP}_{sd,\lambda}$ is equivalent to a shortest-path problem with non-negative weights μ_ℓ^λ, for $\ell \in L$.

As a result of this decomposition, we are able to solve the pricing problem using $|\mathcal{SD}| \times |\Lambda|$ times a polynomial-time graph algorithm (e.g., Dijkstra [Ahuja et al., 1993]), instead of an ILP. As we will see in Section 2.4, this approach yields substantial increase of efficiency.

2.3.3 Initial Set of Columns

We propose a symmetry breaking initialization, solving in sequence the various pricing problems and re-optimizing the RMP more often than in the steady state, as described in the flowchart of Figure 2.2.

Initially, the first RMP is empty and all its dual values are null. This implies that, for each unit connection (v_s, v_d), the first series of pricing problems will compute a shortest path in terms of the number of hops only. In order to avoid getting the same lightpaths for a given node pair, for all the wavelengths, we need to be cautious and therefore re-optimize the RMP more often.

Indeed, in the beginning of the column generation process, we solve all the elementary pricing problems $\text{PP}_{sd,\lambda}$ for a given λ. After each λ *round*, we re-optimize the current RMP, which produces

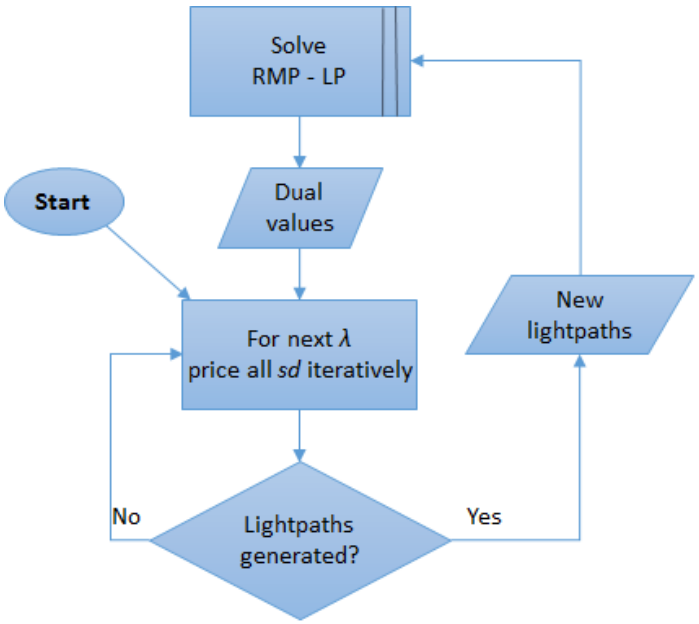


Figure 2.2: Computation of the Initial Set of Columns

new dual values. Those are then used in the elementary pricing problems for the next wavelength, see Figure 2.2) for the details.

The combination of *(i)* the generation of an initial set of columns, *(ii)* the pricing/elimination of the columns after each RMP re-optimization, and *(iii)* the decomposition of the pricing problem into a set of $|\mathcal{SD}| \times |\Lambda|$ elementary pricing problems define the L_CG algorithm.

2.4 Numerical Results

In order to assess the efficiency of the L_CG algorithm, we compare its performance with the current best algorithm, called here W_CG, for solving exactly (ε-optimal algorithm) the max RWA problem, which was introduced and referred to as *CG++* in [Jaumard and Daryalal, 2017].

2.4.1 Data sets

For the sake of comparison, we use the same data set as in [Jaumard and Daryalal, 2017]. It involves six real network topologies: NSFNET, USANET, GERMANY, NTT, ATT, and BRAZIL (see Appendix D). Table 2.1 displays a summary of their characteristics: number of nodes and links, nodal degree, number of available wavelengths (i.e., transport capacity), number of node-pairs with

traffic, overall demand. The traffic instances have been generated randomly. For more details on the data set references, see [Jaumard and Daryalal, 2017].

Table 2.1: Characteristics of the Datasets

Data instances	$\|V\|$	$\|L\|$	nodal degree	W	$\|\mathcal{SD}\|$	$\sum D_{sd}$
NSF_{30}	14	40	3.0	30	141	436
NSF_{75}				75	182	1,371
NSF_{115}				115	182	2,194
USA_{75}	24	88	3.7	75	455	1,336
USA_{125}				125	541	2,422
USA_{150}				150	552	3,509
GER_{100}	50	176	3.5	100	660	2,365
GER_{130}				130		3,041
GER_{150}				150		4,989
NTT_{42}	55	144	2.6	42	338	1,038
NTT_{50}				50	452	1,362
NTT_{150}				150	452	5,684
ATT_{20}	90	274	3.0	20	272	359
ATT_{113}	71	350	4.9	113	2,869	2,918
$BRAZIL_{48}$	27	140	5.2	48	549	1,370

2.4.2 Efficiency of the L_CG model and algorithm

The newly proposed L_CG algorithm, described in Section 2.3, was implemented and tested on a 3.6-4.0 GHz 4-cores machine with 32 GB RAM, using IBM [2016a] for the solution of the (integer) linear programs. Shortest path calculation was done with an open-source Java library JGraphT [Naveh, 2016] implementing Dijkstra's algorithm.

When computing the ILP solution of the last RMP, we observe that CPLEX can take exceedingly longer time without adding much to the accuracy of the integral solution. Therefore, in order to achieve a better trade-off between the solution accuracy and the computational time, we used the following values for the CPLEX parameters (see [IBM, 2016b] for more details on these parameters):

- Relative MIP gap tolerance = 10^{-2} instead of the default CPLEX 10^{-4} value.
- Deterministic Time Limit = 200,000 ticks[1]. We are using a time limit in terms of the number of CPU ticks rather than in seconds in order to have a deterministic behavior across different hardware (server/computer) configurations. Only one instance (ATT_{20}) actually reaches

[1]Deterministic Time Limit a computer-independent metric introduced by IBM. It measures how much algorithmic work is required to obtain a provable optimum, independently of the computer on which it is run on.

Table 2.2: Computational comparisons

Data sets	Solution accuracies					Computational performances (seconds)					
	L_CG			ε (%) comparative		L_CG			Overall CPU comparative		
	$z^{\star}_{LP}$	$\tilde{z}_{ILP}$	GoS (%)	L_CG	W_CG	# cols	CPU LP	CPU ILP	L_CG	W_CG	Ratio
NSF_{30}	430	429	98.4	0.2	2.1	600	0.8	0.1	0.9	9	10
NSF_{75}	1,242	1,233	89.8	0.7	1.0	1,673	2.0	0.3	2.4	10	4.2
NSF_{115}	1,924	1,922	87.6	0.1	0.8	2,104	3.4	0.3	3.7	10	2.7
USA_{75}	1,281	1,275	95.4	0.5	3.1	1,532	5.6	0.2	5.8	141	24.3
USA_{125}	2,255	2,239	92.4	0.7	2.4	4,099	26.3	20.0	46.6	155	3.3
USA_{150}	3,029	3,008	85.7	0.7	1.8	3,460	28.2	0.6	29.0	176	6.1
GER_{100}	2,306	2,277	96.3	1.3	2.7	3,576	51.7	52.0	103.9	474	4.6
GER_{130}	2,960	2,928	96.3	1.1	2.4	4,520	86.0	44.1	130.3	521	4.0
GER_{150}	4,663	4,611	81.4	1.1	3.1	6,054	248.7	23.4	272.4	1,817	6.7
NTT_{42}	1,038	1,037	99.9	0.1	0.0	1,096	0.7	0.1	0.9	30	33.3
NTT_{50}	1,362	1,356	99.6	0.4	0.0	1,415	1.2	0.0	1.3	37	28.5
NTT_{150}	5,553	5,507	96.9	0.8	0.9	5,878	14.3	0.1	14.5	231	15.9
ATT_{20}	359	347	96.7	3.5	1.4	1,065	2.7	286.6	289.4	1,549	5.4
ATT_{113}	2,918	2,890	99.0	1.0	0.5	4,461	342.3	119.5	462.0	1,597	3.5
$Brazil_{48}$	1,370	1,358	99.1	0.9	3.9	2,199	4.5	9.2	13.8	586	42.5
Average				**0.9**	**1.7**				**91.8**	**489**	**5.3**

that limit. Hence, our parameter selection helps to mitigate the effect of this particular data instance on the overall performance of the L_CG algorithm.

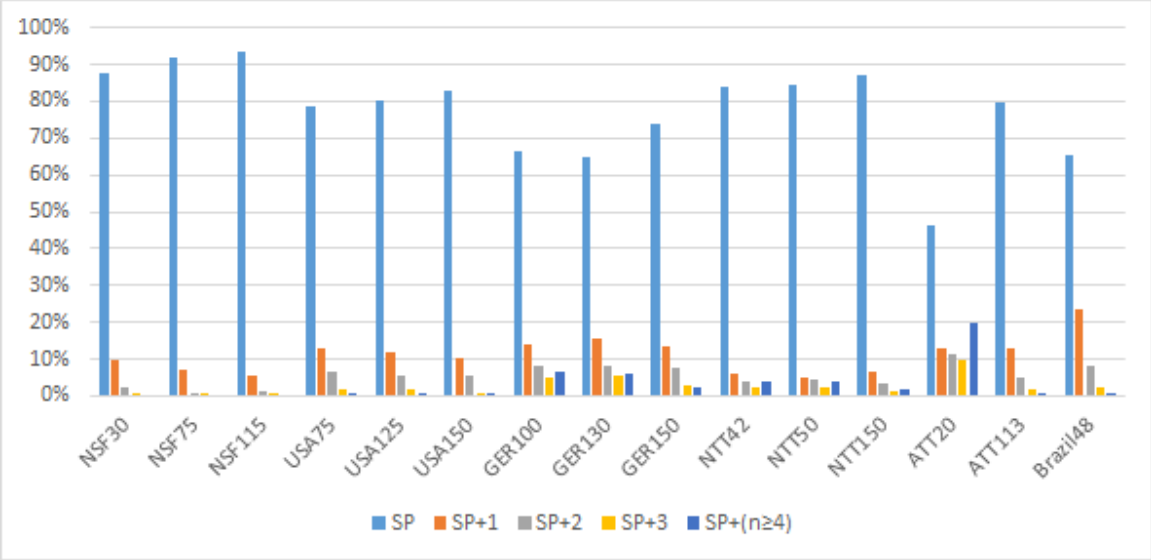
Comparative results are presented in Table 2.2. In the first part of Table 2.2, we provide the lower and upper bounds ($z^{\star}_{LP}$ and $\tilde{z}_{ILP}$) output by the L_CG algorithm. As both algorithms solve exactly the linear relaxations, the lower bounds they output are identical. However, as can be seen with their comparative accuracy, their ILP solution differ. Accuracy (ε) is measured by the relative difference between the lower bound provided by the optimal solution ($z^{\star}_{LP}$) of the linear relaxation of the Master Problem (PM), and the upper bound derived from the value of the ILP solution ($\tilde{z}_{ILP}$), see formula (2.5).

We observe that the L_CG algorithm outputs RWA provisioning with a higher accuracy than the W_CG algorithm, down from an average of $\varepsilon = 1.7$ to less than 1%.

In the second part of Table 2.2, we compare the computational times. We observe that the L_CG algorithm is faster than the W_CG algorithm: from 3 to 42 times faster, for an average of 5 times faster.

These results indicate that the new L_CG algorithm using the *Lightpath* decomposition is more

Figure 2.3: Characteristics of the Selected Lightpaths



scalable than the prior study, and is capable of finding better integral solutions with a smaller accuracy gap.

2.4.3 Percentage of Shortest Paths in the Optimal RWA Provisioning

In order for an algorithm to find a near optimal solution, it must be able not only to consider all the shortest paths for any given node pair, but also to find the lightpaths that do not necessarily correspond to shortest paths. On the other hand, the intuition dictates that, in general, the use of lightpaths longer than shortest paths restrains the capability of provisioning other connections. Consequently, the algorithm must also use the shortest paths as much as possible.

In Figure 2.3, we provide the distribution of the selected lightpaths in terms of their lengths in the optimal RWA provisioning, as output by the L_CG algorithm. The columns of the chart correspond to the length of the routing paths used by the selected connections in unit of number of hops (e.g., SP+1 represents the number of connections routed on a path that is one hop longer than the shortest path of the corresponding node-pair).

We can observe that the solutions output by the L_CG model and algorithm contain a large fraction of shortest paths. Indeed, the average percentage of usage of shortest paths among all the instances is 78%, which is much better that the percentage in the solutions output by the W_CG

model and algorithm, i.e., 66%.

Taking into account that the accuracy of the solutions is less than 1% for nearly all of them, it is unlikely that the percentage of shortest paths would be able to decrease much more.

These improvements were obtained using a modeling that bear some symmetry and require more variables (columns) than the W_CG model, i.e., around at least $|V|^2 \times |\Lambda|$ columns (or in other words, at least one lightpath per node pair with traffic and wavelength), while the W_CG model requires only at most $|\Lambda|$ variables with non zero values. Indeed, the extra time that is potentially required for solving larger LPs is counterbalanced by the polynomial-time solution of the pricing problems, again even if there is a large number of them.

2.5 Conclusion

We proposed a very simple yet very efficient column generation decomposition to RWA problem using a lightpath-based decomposition model. Routes are dynamically generated using a weighted shortest path algorithm, where weights are determined by the values of the dual variables of the RMP, and take implicitly into account both the network connectivity and the traffic distribution.

Performances of the proposed decomposition schemes are compared with the best previously proposed ε-optimal algorithm. Resulting accuracy values are on average less than 1%, down from 1.7% in [Jaumard and Daryalal, 2017], and 5 times faster on average.

Additionally, we showcased that model symmetry is not a hindrance to its efficiency, although it may affect the performance of the branch-and-bound when deriving an ILP solution.

Future work will investigate the re-use of such a decomposition scheme for RSA networks, with preliminary promising results in [Enoch and Jaumard, 2018b].

Chapter 3

Towards Optimal and Scalable Solution for *Routing and Spectrum Allocation*

Enoch and Jaumard [2018b]. Towards optimal and scalable solution for routing and spectrum allocation. Electronic Notes in Discrete Mathematics, 64, 335 - 344. 8th International Network Optimization Conference - INOC 2017.

3.1 Introduction

In order to face the steady growth of the optical networks, network operators are now moving to flexible or elastic optical networking. Therein, the optical spectrum is used more efficiently by allowing finer grid spacing, resulting in sub-streams, called slots. The challenge is then to optimize the spectrum usage through the so-called Routing and Spectrum Assignment (RSA) problem. It appears to be a much more difficult problem than the classical routing and wavelength assignment problem.

The paper is organized as follows. In Section 3.2, after a detailed statement of the RSA problem, we describe the new decomposition model we propose. The solution scheme follows in Section 3.3. Numerical results are described in Section 3.4. Conclusions are drawn in the last section.

3.2 A New RSA Decomposition Model

3.2.1 Statement of the RSA Problem

The RSA problem assumes an undirected graph $G = (V, L)$ with optical node set V and link set L. We denote by $\omega(v)$ the set of links adjacent to v, for $v \in V$. The bandwidth is slotted into a set S of spectrum slots, and a guard band of one slot is required between spectrum slices assigned to different requests. The traffic is defined by a set K of requests where each request $k \in K$ has a source (s_k), a destination (d_k) and a spectrum demand D_k, expressed in terms of a number of slots. The traffic is assumed to be symmetrical.

3.2.2 RSA Decomposition Model

We propose a column generation model relying on lightpaths, where a lightpath, denoted by c, refers to the provisioning of a request using a routing path and a spectrum slice characterized by a starting slot denoted by s. Let C be the set of all possible lightpaths. Each lightpath is characterized by:

- a_k^c: indicates if a request k is provisioned by lightpath c.
- b_ℓ^{sc}: indicates if a slot s is used on link ℓ in lightpath c.

The model uses decision variables z_c such that each z_c indicates if lightpath c is selected in the solution.

The objective maximizes the throughput and is written:

$$z \quad = \quad \max \quad \sum_{c \in C} \left(\sum_{k \in K} D_k a_k^c \right) z_c \tag{3.1}$$

subject to:

$$\sum_{c \in C} a_k^c z_c \leq 1 \qquad k \in K \tag{3.2}$$

$$\sum_{c \in C} b_\ell^{sc} z_c \leq 1 \qquad \ell \in L, s \in S \tag{3.3}$$

$$z_c \in \{0, 1\} \qquad c \in C. \tag{3.4}$$

Constraints (3.2) express that each request is provisioned at most once. Constraints (3.3) make sure that each slot is never used more than once on each fiber link.

3.3 Solution Design

Given the large number of variables/columns in the proposed model, we resort to the *Column Generation* method to solve its Linear Programming (LP) relaxation [Chvatal, 1983]. This technique consists of decomposing the original problem into a restricted master problem - RMP - (i.e., model (3.1) - (3.4) with a very restricted number of variables) and one or several pricing problem(s) - PP. RMP and PPs are solved alternately. Solving RMP consists in selecting the best lightpaths, while solving one PP allows the generation of an improving potential lightpath, i.e., a lightpath such that, if added to the current RMP, improves the optimal value of its LP relaxation. The process continues until the optimality condition is satisfied, that is, the so-called reduced cost that defines the objective function of the pricing problems is negative for all of them, see [Chvatal, 1983] if not familiar with linear programming concepts. An ε-optimal solution for the RSA problem is derived by solving exactly the ILP model associated with the last RMP.

Let K_s denote the set of requests that have the potential to be provisioned by a lightpath which starts at slot s: $K_s = \{k \in K : s + D_k - 1 \leq |S|\}$. Let D_k^s be the number of slots needed for request k in K_s taking into account the guard-band requirement:

$$D_k^s = D_k \qquad k \in K_s \ : \ s + D_k - 1 = |S| \tag{3.5}$$

$$D_k^s = D_k + 1 \qquad k \in K_s \ : \ s + D_k - 1 < |S|. \tag{3.6}$$

Each pricing problem is indexed by k and s, and produces a single potential lightpath for provisioning demand k, starting at slot s.

The definition of the decision variables is as follows:

- β_ℓ indicates if the lightpath uses link ℓ or not ;
- $b_\ell^{s'}$ indicates if slot s' is used on a link ℓ or not.

Let $\mu_k^{(3.2)}$ and $\mu_{\ell s'}^{(3.3)}$ be the values of the dual variables associated with constraints (3.2) and (3.3), respectively. The pricing problem can be written as follows:

$$rc \quad = \quad \max \quad D_k - \mu_k^{(3.2)} - \sum_{\ell \in L} \sum_{s' \in S} \mu_{\ell s'}^{(3.3)} \, b_\ell^{s'} \tag{3.7}$$

subject to:

$$\sum_{\ell \in \omega(s_k)} \beta_\ell = \sum_{\ell \in \omega(d_k)} \beta_\ell = 1 \tag{3.8}$$

$$\sum_{\ell \in \omega(v)} \beta_\ell \leq 2 \qquad v \in V \setminus \{s_k, d_k\} \tag{3.9}$$

$$\sum_{\ell' \in \omega(v) \setminus \{\ell\}} \beta_{\ell'} \geq \beta_\ell \qquad v \in V \setminus \{s_k, d_k\}, \ell \in \omega(v) \tag{3.10}$$

$$\sum_{s'=s}^{s+D_k^s-1} b_\ell^{s'} = D_k^s \, \beta_\ell \qquad \ell \in L \tag{3.11}$$

$$\beta_\ell, b_\ell^{s'} \in \{0, 1\} \qquad \ell \in L, s' \in S \tag{3.12}$$

Constraints (3.8), (3.9) and (3.10) define the routing of the current request. Constraints (3.11) reserve a contiguous spectrum channel for the current request.

We observe that for each link ℓ:

$$b_\ell^{s'} = \beta_\ell \qquad s' \in \{s, \ldots, s + D_k^s - 1\}; \tag{3.13}$$

$$b_\ell^{s'} = 0 \qquad s' \notin \{s, \ldots, s + D_k^s - 1\}. \tag{3.14}$$

Therefore, the reduced cost can be rewritten:

$$rc \quad = \quad \max \qquad D_k - \mu_k^{(3.2)} - \sum_{\ell \in L} \left(\sum_{s'=s}^{s+D_k^s-1} \mu_{\ell s'}^{(3.3)} \right) \beta_\ell. \tag{3.15}$$

The first term is a constant for each request, and the second term corresponds to a summation over the links of the network. Therefore, we can solve the pricing problem using the following objective function:

$$\min \quad \sum_{\ell \in L} \left(\sum_{s'=s}^{s+D_k^s-1} \mu_{\ell s'}^{(3.3)} \right) \beta_\ell, \tag{3.16}$$

where $\mu_{\ell s'}^{(3.3)}$ are non-negative dual values. We conclude that, for each request k, the lightpath generator corresponds to a weighted shortest-path problem with link weight: $\sum_{s'=s}^{s+D_k^s-1} \mu_{\ell s'}^{(3.3)}$. As a result, the pricing problem can be solved with a polynomial time algorithm, e.g., Dijkstra's algorithm.

The idea of using a shortest path algorithm during the pricing phase has been used by Ruiz et al. [2013a], but in our solution scheme, we solve exactly the LP relaxation of model (3.1) - (3.4) whereas Ruiz et al. [2013a] do not.

The flowchart of the solution process is depicted in Figure 3.1. We reach the optimal LP solution as soon as the LP optimality condition is satisfied, i.e., we get a complete round of iterations spanning all starting slots and all requests without any new improvement in the LP value.

We have noticed that this approach produces a large number of lightpaths that end up in the ILP problem, which can make the ILP solution process very long. In order to alleviate this problem, we followed an academic way of implementing column generation, which consists of removing the non basic columns, i.e., with variables which are equal to zero in the current LP solution after each optimization of the RMP. This technique allows a great speedup both during the column generation and the ILP solution phases.

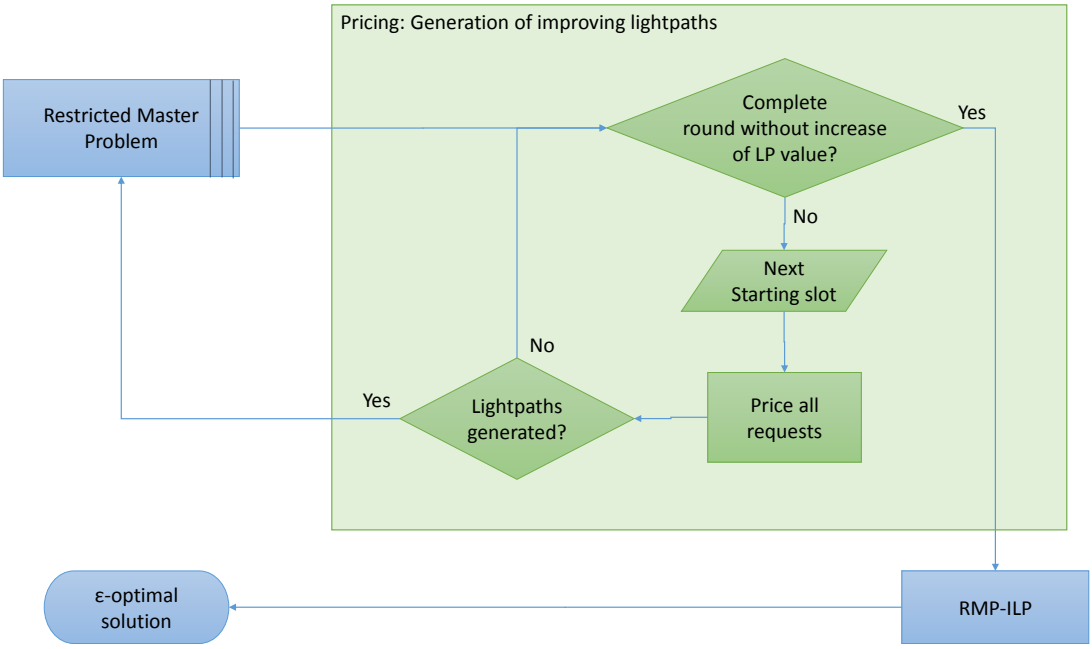


Figure 3.1: Flow chart of the L_CG for RSA

3.4 Numerical Results

The model and solution design described above was implemented on a 3.6-4.0 GHz 4-cores machine with 32 GB of RAM, with the use of CPLEX (version 12.6.0) for solving the (integer) linear programs. Shortest path calculation was done with an open-source java library JGraphT implementing Dijkstra's algorithm.

In a first set of experiments, we conducted experiments in order to assess the scalability of our solution process, and the accuracy of the RSA solutions that were output. We used the same set of data instances as [Jaumard and Daryalal, 2016], i.e., the Spain network with 21 nodes and 35 links (Figure 3.2), the same demand sets, and the same spectral efficiency of 25 Gbps per slot. With regard to the criterion for removing less promising columns during the LP phase, and considering that these instances are relatively small, and therefore relatively easy to solve, we used a non-aggressive strategy consisting of removing columns with negative reduced price (within a small tolerance). Results are shown in Table 3.1, and include a comparison with those of Jaumard and Daryalal [2016].

We observe that the LP optimal values, denoted by $z^{\star}_{LP}$, are all equal to the offered load. We

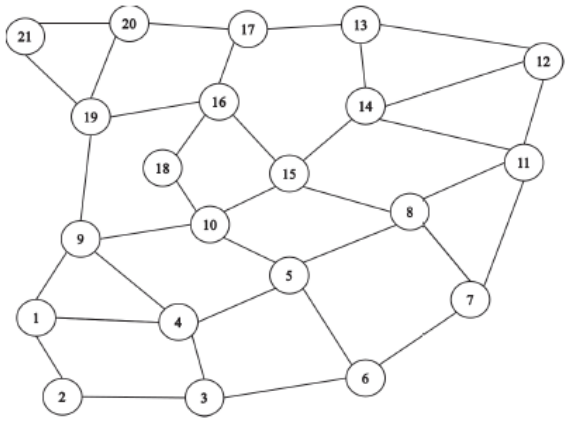


Figure 3.2: Spain network topology

observe also that our new model/algorithm is much faster, up to 2 orders of magnitude, and is capable of finding the optimal RSA solutions, while all solution accuracies for the model and algorithm of Jaumard and Daryalal [2016] are larger than 10%.

In a second set of experiments, we use larger data sets: First series of data sets uses again the Spain network; Second series of data sets uses USA Network with 24 nodes and 43 links. In addition, these data sets have the following characteristics:

- Demands are drawn from granularities {4, 8, 16} with respective proportions {70%, 20%, 10%}, thus representing demands that are more representative of those encountered in today's networks.

- Offered load is spread over all node pairs; Therefore, after aggregation, the number of requests given as input to our algorithm corresponds to the number of node pairs in the network. This means that number of aggregated requests is maximal, and that the size of the network is actually an accurate indicator of the size of the RSA problem instance.

Considering that these data sets are relatively large, we modified some of the execution parameters to get the best performance as follows:

- During the LP phase, the least interesting columns are removed based on whether they are in the basis or not, and not based on the value of their reduced cost.

- CPLEX parameter *NumericalEmphasis* is turned on, in order to get better numerical accuracy

Traffic demand			Current study							Previous study	
Offered load (Tbps)	$\|D\|$	$\|S\|$	$z^{\star}_{LP}$	# columns generated	# columns selected	$\tilde{z}_{ILP}$ Throughput	ε	CPU (sec) LP	CPU (sec) Total	ε	CPU (sec)
Spain network: demand granularities in $\{1, 2, \ldots, 8\}$ slots, i.e., {25, 50, ..., 200} Gbps											
3.7	35	50	3.7	1661	35	3.7	0.0%	0.3	0.3	14%	50
4.8	45	60	4.8	2619	45	4.8	0.0%	0.3	0.4	13%	86
6.8	60	75	6.8	4409	60	6.8	0.0%	0.6	0.7	15%	147
7.5	64	85	7.5	5273	64	7.5	0.0%	1.1	1.3	19%	176
7.4	70	100	7.4	6857	70	7.4	0.0%	1.4	1.7	16%	263
9.7	80	120	9.7	9456	80	9.7	0.0%	2.3	2.5	16%	323
12.0	112	150	12.0	14377	112	12.0	0.0%	4.3	5.3	14%	417
20.5	180	330	20.5	59272	180	20.5	0.0%	23.5	25.6	18%	1,606
Spain network: demand granularities in $\{2, 4, \ldots, 16\}$ slots, i.e., {50, 100, ..., 400} Gbps											
7.5	35	80	7.5	2601	35	7.5	0.0%	0.3	0.9	10%	134
9.8	45	110	9.8	4208	45	9.8	0.0%	1.5	2.0	10%	177
10.7	60	156	10.7	9121	60	10.7	0.0%	2.6	3.1	12%	261
15.5	64	170	15.5	10472	64	15.5	0.0%	4.2	4.7	16%	630
15.1	70	236	15.1	16356	70	15.1	0.0%	7.2	7.8	13%	1,342
16.9	80	256	16.9	20085	80	16.9	0.0%	9.6	10.3	14%	1,419
Average							**0.0**		**4.8**	**14.3**	**502.2**

Table 3.1: Comparative Model/Algorithm Performance

from CPLEX, although it entails larger computational times.

- For the largest of these data sets, the number of columns in the ILP phase can be very large causing the ILP solver to run out of memory. To work around this issue, we apply a rounding technique where we remove the columns with the smallest values in the LP optimal solution (less than 0.2). This also improves the speed of the ILP phase while slightly sacrificing on the quality of the solution.

The summary of these experiments is shown in Table 3.2. Even for the larger network USA, and with large offered loads (up to 70 Tbps), the gap between the linear solution and the integral solution remains small, and only for the largest data sets does the gap become greater than 10%. More remarkably, the time of execution for most of the data sets is within seconds, and only for the largest data sets does the time become in the order of minutes, and in any case, less than 30 minutes, which is much faster than any previous work with comparable datasets.

Indeed, instances with an upper bound ($z^{\star}_{LP}$) that differs from the offered load appear to be more difficult to solve. They require significantly more computational time, and the accuracy of the ILP solution decreases.

Offered load (Tbps)	$\|D\|$	$\|S\|$	$z^\star_{LP}$	# columns Generated	# columns Selected	$z^\star_{ILP}$ Throughput	ε	CPU (sec) LP	CPU (sec) Total
Spain Network - Demands in $\{4, 8, 16\}$ slots, i.e., {100, 200, 400} Gbps									
50.2	210	400	50.2	341	196	48.0	4.6%	21.7	22.4
60.0	210	400	60.0	454	197	57.8	3.8%	27.7	28.1
70.2	210	400	70.2	548	192	66.6	5.4%	41.7	42.2
80.0	210	400	80.0	702	185	73.6	8.7%	71.4	71.9
90.2	210	400	86.9	1266	161	75.0	15.8%	1112.7	1131.1
USA Network - Demands in $\{4, 8, 16\}$ slots, i.e., {100, 200, 400} Gbps									
50.2	276	400	50.2	489	257	46.9	7.0%	28.1	29.4
60.0	276	400	60.0	671	248	55.0	9.1%	53.3	55.8
70.2	276	400	70.2	865	243	63.9	9.9%	101.0	108.4
80.0	276	400	78.3	1685	216	65.5	19.6%	1050.8	1122.0
90.2	276	400	85.7	1602	205	72.4	18.4%	1495.2	1532.0

Table 3.2: Numerical results for new datasets

3.5 Conclusions

In this work, we proposed a new column generation model and algorithm for solving the RSA problem. In doing so, we defined a decomposition based on starting slots and on requests in such a way that the pricing problem was reduced to a shortest path problem, thus improving greatly the performance and improving significantly the accuracy of the RSA solutions that are output. As a result, we achieved a performance that is clearly better than the previous works.

However, further investigations on alternate models are needed. With the proposed model, for larger data sets with, e.g., 100 nodes or more, the number of columns in the ILP phase will be too large. Although the key for getting better ILP solutions than in the literature for this study, such a large number of columns is a bottleneck for solving the RSA problem with larger data sets.

Chapter 4

Nested Column Generation Algorithm for the RWA Problem

In our previous RWA algorithm [Enoch and Jaumard, 2018a], we proposed a *Lightpath* decomposition model and solved it using *Column Generation* technique which resulted in considerable improvements over the earlier study [Jaumard and Daryalal, 2017]. In this chapter, we consider the wavelength *Configuration* decomposition model that was used in [Jaumard and Daryalal, 2017], and we solve it using a *Nested Column Generation* algorithm.

4.1 Decomposition model

Given a WDM optical network, using the fixed frequency grid definition (i.e., the optical spectrum is sliced into a number of wavelengths), and given a connection demand matrix, the RWA problem consists of allocating a routing path and an optical wavelength to each connection, as to maximize the Grade-of-Service (i.e., the proportion of granted connections).

A WDM optical network is represented by a directed graph $G = (V, L)$ where V represents the nodes of the network, and L is the set of links representing the directional optical fibers. The optical spectrum is divided into a set of wavelengths with equal capacity (50 GHz), denoted by Λ. Let $W = |\Lambda|$. The traffic is defined by a $|V| \times |V|$ matrix where each element D_{sd} represents the number of wavelength units from a source node v_s to a destination node v_d. Let $\mathcal{SD}$ be the set of

node-pairs with traffic.

We propose an ILP model based on a *Configuration* decomposition. We define a *Configuration* as a set of link-disjoint lightpaths provisioning distinct connections on the same wavelength as illustrated in Figure 4.1. Links are assumed to be bi-directional. Each link is used at most once in each direction.

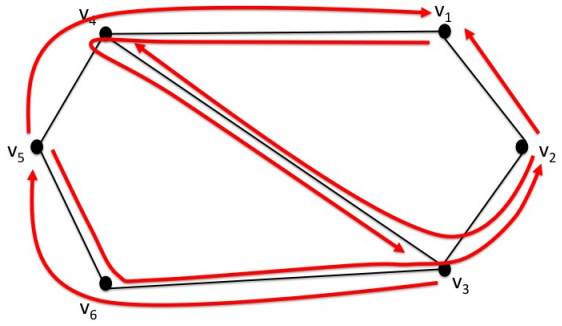


Figure 4.1: RWA wavelength *Configuration*.

Let C be the set of all possible *Configurations*. Each *Configuration* $c \in C$ is defined with the help of the following parameter:

- a_{sd}^{c}: gives the number of connections from v_s to v_d that are provisioned by *Configuration* c. Given that each *Configuration* is constructed on a single wavelength, this parameter corresponds to the number of routing paths allocated for the given node pair on the current wavelength.

The model uses two sets of decision variables:

- z_c is an integer variable that gives the number of occurrences of *Configuration* c in the solution.

- y_{sd} is an intermediate integer variable that computes the number of granted connections from v_s to v_d in the solution, and can be expressed in terms of z_c as follows:

$$y_{sd} = \sum_{c \in C} a_{sd}^{c} z_c \qquad (v_s, v_d) \in SD \tag{4.1}$$

The ILP model is the same as the one in [Jaumard and Daryalal, 2017] and reads as follows:

$$z = \max \sum_{(v_s,v_d)\in SD} y_{sd} \tag{4.2}$$

subject to:

$$\sum_{c\in C} z_c \leq W \tag{4.3}$$

$$y_{sd} \leq \sum_{c\in C} a_{sd}^c z_c \qquad (v_s, v_d) \in SD \tag{4.4}$$

$$y_{sd} \leq D_{sd} \qquad (v_s, v_d) \in SD \tag{4.5}$$

$$z_c \in \mathbb{N} \qquad c \in C \tag{4.6}$$

$$y_{sd} \geq 0 \qquad (v_s, v_d) \in SD. \tag{4.7}$$

Constraints (4.3) ensure that the accepted *Configurations* are within the number of available wavelengths. Constraints (4.4) ensure that the GoS calculated by the variables y_{sd} is backed by actual provisioning in the *Configurations* z_c. Constraints (4.5) limit the number of granted connections to the demand.

In addition to this model (with two sets of variables), Jaumard and Daryalal [2017] provided another formulation not making use of the intermediate variables y_{sd} (see Appendix C). In the process of the current research, we investigated both models by implementing both of them using *Nested Column Generation*. We have discovered that the one-variable-set formulation performs very poorly both in terms of the quality of the solution, and in terms of the computation time. The intuition behind this is that the *Configuration* columns are relatively large which makes LP relaxation take longer to converge to its optimal value ($z^\star_{LP}$). This also leads to a high contention among the *Configurations* during the ILP *Branch-and-Bound* phase, which produces an integral solution ($\tilde{z}_{ILP}$) with a relatively large accuracy gap.

On the other hand, the two-variable-sets model uses a set of intermediate variables y_{sd} which translates in relaxing the upper-bound demand constraints (C.2) on the variables z_c. In fact, the

constraints (4.4) are a relaxation of the definition (4.1). As a consequence of this relaxation, the two-variable-sets model requires to post-process the *Configurations* in the final solution, as to remove the extra provisioning that violates the demand constraints and to re-establish the equality (4.1).

In the sequel of this chapter we will describe the solution design for the two-variable-sets model which was used to produce the results in [Jaumard and Daryalal, 2017] too.

4.2 Nested Column Generation

Column Generation consists of decomposing the original problem into a *Restricted Master Problem* (RMP), i.e., with a restricted number of variables, and one or several *Pricing Problems* (PP). The RMP and the PP(s) are solved alternately. Solving the RMP consists in selecting the best columns, while solving one PP allows the generation of an improving potential column, i.e., a column such that, if added to the current RMP, improves the optimal value of its LP relaxation. The process continues until the optimality condition is satisfied, i.e., the so-called reduced cost that defines the objective function of the *Pricing Problems* is non-positive for all of them [Chvatal, 1983]. The optimal value of the linear relaxation of the so-obtained RMP is denoted by $z^{\star}_{LP}$ and represents an upper-bound on the optimal solution of the RWA problem. After the *Column Generation* phase, the next step consists of solving the ILP model associated with the last RMP, which produces an integral solution with a value, denoted as $\tilde{z}_{ILP}$, which represents a lower-bound on the optimal solution of the RWA problem. The integral solution thus-obtained is said to be ε-optimal where ε denotes the relative difference between the two bounds: $\varepsilon = (z^{\star}_{LP} - \tilde{z}_{ILP})/z^{\star}_{LP}$.

Considering the *Configuration* decomposition in [Jaumard and Daryalal, 2017], although *Column Generation* is a powerful technique, we observed that the link-based formulation of the *Pricing Problem* was not efficient enough. Given that the *Pricing Problems* need to be solved at each iteration of *Column Generation*, their performance can be an important hindrance to the over-all efficiency. In order to address this difficulty, we devised an algorithm based on the idea of solving the *Pricing Problem* itself using *Column Generation*, which approach is referred to as *Nested Column Generation*.

4.2.1 Review of Nested Decompositions

The idea of applying recursive decomposition was suggested by Dantzig [1963]. Some of the first generic implementations for *Linear Program (LP)* go back to the early 70's such as Glassey [1973] and Ho and Manne [1974]. Subsequently, many implementations for *Integer Linear Programing* have been produced.

Vanderbeck [2001] implemented a nested decomposition approach to a cutting-stock problem. First the author devise a subproblem that generates *cutting patterns* and solves it using *Column Generation*, in turn, with the help of another subproblem that generates *sections*. The author notes that the cutting-pattern generation subproblem is only solved approximately given that *Column Generation* only produces lower and upper bounds on the minimum reduced cost of a cutting-pattern, and uses the lower bounds on the reduced costs to produce a Lagrangian bound on the cutting problem. The author recognizes that this is a "heuristic based on the tools of exact optimization", given that the optimality of the Lagrangean bound is not guaranteed.

Hennig et al. [2012] proposed a *Nested Column Generation* algorithm for the *Crude Oil Tanker Routing and Scheduling* problem such that the first subproblem generates a cargo-route for each ship. This subproblem is solved using a *Branch-and-Price* algorithm with the help of a second-level subproblem which generates a route for each ship.

Closer to the applications in optical networks, Vignac et al. [2016] presented multiple models for the *Grooming, Routing and Wavelength Assignment* problem, among which, a *Column Generation* algorithm where a subproblem for each wavelength is defined to find the traffic carried by this wavelength, called a *Wavelength Routing Configuration*. This subproblem is itself decomposed into arc-disjoint grooming-patterns, thus leading to a nested decomposition. Similarly to Vanderbeck [2001], the bounds of the *Pricing Problems* are used to compute Lagrangean dual bounds on the master LP, thus resulting into a heuristic.

Caprara et al. [2016] referred to *Nested Column Generation* as *Recursive Dantzig-Wolfe Reformulation* and used this approach to design a *Branch-and-Price* algorithm for solving the *Temporal Knapsack Problem*.

Tilk et al. [2018] proposed an algorithm for solving the *Vehicle Routing Problem with Multiple*

Resource Interdependencies where both the *Master Problem* and the *Pricing Problem* are solved using Branch-and-Price. The authors also give a detailed review on the use of nested decompositions in other publications (e.g., Cordeau et al. [2001]; Dohn and Mason [2013]).

4.2.2 Processing Flow

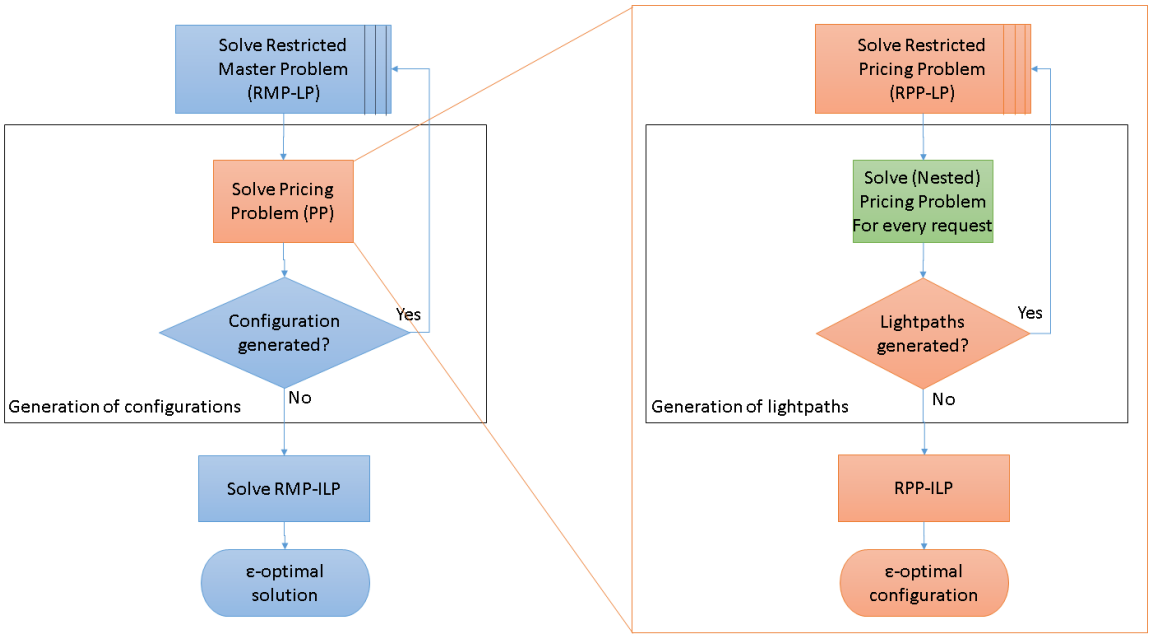


Figure 4.2: Nested Column Generation Flow for RWA

Figure 4.2 shows the processing flow of our *Nested Column Generation* algorithm as applied to the RWA problem. The *Column Generation* flow used to solve the Pricing problem, is identical to that used to solve the Master problem, thus the naming of *Nested Column Generation*. We exit the process of generating *Configurations* for the *Master Problem*, when the *Pricing Problem* cannot produce an improving *Configuration*, i.e., the optimal value of the ILP associated with the *Pricing Problem* is null ($rc^{\star}_{ILP} = 0$).

4.2.3 Optimality of the Linear Relaxation

The main outcome of *Column Generation* is to produce a strong linear relaxation bound $z^{\star}_{LP}$ on the optimal solution of the *Master Problem*. This outcome is based on the premise that the *Pricing Problem* is solved to optimality [Dantzig, 1963], in particular, when the stopping condition to exit

Column Generation is verified ($rc^{\star}_{ILP} = 0$). However, in *Nested Column Generation*, the *Pricing Problem* is solved using *Column Generation* meaning that the integer solutions that we obtain for the *Pricing Problems* are only lower-bounds (assuming *Branch-and-Price* is not employed as is the case in this study).

Nevertheless, thanks to *Nested Column Generation*, we have also an upper bound on the optimal solution of the *Pricing Problem* ($rc^{\star}_{LP}$). So when the *Pricing Problem* fails to generate a new *Configuration* ($\tilde{rc}_{ILP} = 0$), there are two possibilities:

- $rc^{\star}_{LP} = 0$: we can deduce that $rc^{\star}_{ILP} = 0$, and therefore the exit condition is satisfied. In this case, the LP relaxation of the *Master Problem* is optimal.
- $rc^{\star}_{LP} > 0$: this means there is no guarantee that $rc^{\star}_{ILP} = 0$, and therefore, the LP relaxation of the *Master Problem* might not be optimal.

To sum up, theoretically *Nested Column Generation* does not guarantee the accuracy of the LP relaxation bound of the *Master Problem* (). In fact, some studies resorted to a *Branch-and-Price* approach (with additional overhead) for solving the *Pricing Problem* in order to guarantee this accuracy [Hennig et al., 2012; Caprara et al., 2016; Tilk et al., 2018]. However, it is possible empirically to conclude whether the LP relaxation is optimal or not by verifying if the *Pricing Problem* was solved to optimality in the last iteration.

4.3 Solution Design

As shown in Figure 4.2, we apply *Nested Column Generation* where we define a *Pricing Problem* that feeds *Configurations* to the *Master Problem*, and we solve this *Pricing Problem* in turn using *Column Generation*, meaning that we define a *Nested Pricing Problem* which feeds *Lightpaths* to the first level *Pricing Problem*.

4.3.1 Pricing Problem

The *Pricing Problem* aims to generate a *Configuration* c for any wavelength. In the sequel of this section, we will drop the c index to alleviate the mathematical notation.

Let μ and μ_{sd} be the dual values corresponding to constraints (4.3) and (4.4) respectively. The reduced cost corresponding to the model ((4.2)-(4.7)) is:

$$rc \quad = \quad \max \quad \sum_{(v_s,v_d)\in SD} \mu_{sd} a_{sd} \; - \; \mu \tag{4.8}$$

Generating a *Configuration* consists of finding routing paths π for connections between different node-pairs. Let Π_{sd} be the set of all possible paths from v_s to v_d. A routing path is defined with the help of the following parameters:

- δ_π^ℓ is a binary parameter that indicates if link ℓ is part of routing path π.

We define the following decision variables:

- β_π is a boolean variable that indicates if a path π is selected in the current *Configuration*.

We can write the correspondence between β_π and the parameters of the *Master Problem* as follows:

$$a_{sd} = \sum_{\pi\in\Pi_{sd}} \beta_\pi \qquad (v_s, v_d) \in SD. \tag{4.9}$$

After a variable substitution in the reduced cost (4.8), we obtain the *Pricing Problem* model as follows:

$$rc \quad = \quad \max \quad \sum_{(v_s,v_d)\in SD} \sum_{\pi\in\Pi_{sd}} \mu_{sd}\beta_\pi \; - \; \mu \tag{4.10}$$

subject to:

$$\sum_{\pi\in\Pi_{sd}} \beta_\pi \le D_{sd} \qquad (v_s, v_d) \in SD \tag{4.11}$$

$$\sum_{(v_s,v_d)\in SD} \sum_{\pi\in\Pi_{sd}} \delta_\pi^\ell \beta_\pi \le 1 \qquad \ell \in L \tag{4.12}$$

$$\beta_\pi \in \{0, 1\} \qquad \pi \in \Pi. \tag{4.13}$$

Constraints (4.11) limit the number of paths for a given node-pair to the demand. Constraints (4.12) mean that each link is traversed by at most one path. This ensures that the paths forming the *Configuration* are link-disjoint.

4.3.2 Path Generator

The path generator (*Nested Pricing Problem*) aims to generate routing paths for each node-pair, and feeds them to the - first level - *Pricing Problem*.

Let ν_{sd} and ν_ℓ be the dual values corresponding to constraints (4.11) and (4.12) respectively. For each node-pair $(v_s, v_d) \in SD$, the reduced cost of a path $\pi \in \Pi_{sd}$ is:

$$\max \quad \mu_{sd} - \nu_{sd} - \sum_{\ell \in L} \nu_\ell \delta_\pi^\ell. \tag{4.14}$$

We get a shortest-path problem with non-negative weights for each $(v_s, v_d) \in SD$

$$\min \quad \sum_{\ell \in L} \nu_\ell \delta_\pi^\ell \tag{4.15}$$

which is solved in polynomial time.

4.4 Numerical Results

We ran this algorithm on the same dataset as in [Jaumard and Daryalal, 2017; Enoch and Jaumard, 2018a]. For the best performance, we ran CPLEX in a single thread and we turned on the *CPX_PARAM_NUMERICALEMPHASIS* [1] switch. We have also observed that the ILP phase can take too long without adding much to the quality of the integral solution. Therefore we opted to pre-terminate the ILP phase by setting the parameter *CPX_PARAM_EPGAP* [2] to 1.e-1, thus making a trade-off in the benefit of computation time. Table 4.1 shows the complete results and comparison with the previous algorithms. (We did not re-implement the reference algorithm W_CG presented by Jaumard and Daryalal [2017], and the results shown below are those published by its authors.)

[1] Numerical precision emphasis
[2] Relative MIP gap tolerance: default: 1e-04.

Table 4.1: W_NCG Algorithm Results and Comparison

	Solution						Computational performance (sec)					
	W_NCG			ϵ (%) comparative			W_NCG			Overall CPU comparative		
Data set	$z^{\star}_{LP}$	$\tilde{z}_{ILP}$	GoS (%)	W_NCG	L_CG	W_CG	# cols	LP CPU	ILP CPU	W_NCG	L_CG	W_CG
NSF30	430	395	90.6	8.9	0.2	2.1	66	1.0	0.2	1.2	0.9	9
NSF75	1242	1205	87.8	3.1	0.7	1.0	71	1.2	0.1	1.3	2.4	10
NSF115	1924	1890	86.1	1.8	0.1	0.8	68	0.8	0.1	0.9	3.7	10
USA75	1281	1170	87.6	9.5	0.5	3.1	169	19.0	0.9	19.9	5.8	141
USA125	2255	2094	86.5	7.7	0.7	2.4	174	30.3	0.2	30.5	46.6	155
USA150	3029	2895	82.5	4.6	0.7	1.8	171	25.5	0.2	25.7	29.0	176
GER100	2306	2114	89.4	9.1	1.3	2.7	211	59.1	2.0	61.1	103.9	474
GER130	2960	2738	90.0	8.1	1.1	2.4	251	88.9	0.9	89.8	130.3	521
GER150	4663	4314	76.1	8.1	1.1	3.1	282	318.7	3.6	322.3	272.4	1817
NTT42	1038	1038	100.0	0.0	0.1	0.0	15	0.1	0.1	0.2	0.9	30
NTT50	1362	1341	98.5	1.6	0.4	0.0	36	0.4	0.1	0.5	1.3	37
NTT150	5553	5271	92.7	5.4	0.8	0.9	167	9.5	0.5	10.0	14.5	231
ATT20	359	322	89.7	11.5	3.5	1.4	90	5.4	0.4	5.8	289.4	1549
ATT113	2918	2897	99.3	0.7	1.0	0.5	627	216.7	1.6	218.3	462.0	1597
Brazil48	1370	1242	90.7	10.3	0.9	3.9	132	17.3	1.6	18.9	13.8	586
Average				**6.0**	**0.9**	**1.7**				**53.8**	**91.8**	**490**

We observe for all the instances that: When the *Pricing Problem* fails to generate a new *Configuration* (i.e., $\tilde{rc}_{ILP} = 0$), we have also $rc^{\star}_{LP} = 0$, which implies that $rc^{\star}_{ILP} = 0$ meaning that the *Pricing Problem* is solved optimally. Therefore, the LP relaxation solution obtained for the *Master Problem* $z^{\star}_{LP}$ is optimal and is an upper-bound on the optimal solution $z^{\star}_{ILP}$ (see discussion at Section 4.2.3).

Due to the pre-termination of the ILP phase, the time spent in this phase is very small, which leads to an overall computation time of 53.8 seconds on average which is better than the other algorithms. However, the ILP gap of 6% on average is larger than the other algorithms. This can be partially due the large size of the *Configuration* columns as compared to the *Lightpath* columns. In fact, even if the number of columns in the *Configuration* decomposition is smaller, it intersects with more constraints which makes for an ILP with a higher density, thus more difficult to solve.

Although the W_NCG algorithm is fast, it suffers from a large accuracy gap of the integral solution. The algorithm L_CG presented in [Enoch and Jaumard, 2018a] using a *Lightpath* decomposition remains the best performing algorithm overall.

Chapter 5

Nested Column Generation Algorithm for the RSA Problem

The RSA optimization problem is key to the provisioning of *Elastic Optical Networks*. This problem has been shown to be NP-hard by Shirazipourazad et al. [2013]. Therefore, in order to solve it *exactly* (i.e., with a calculated accuracy), we need to employ *Integer Linear Programing*. While ILP compact formulations (e.g., link-based) have a polynomial number of variables, they do not scale well when the problem instances reach a real-life size. Therefore, we consider decomposition formulations.

In our previous study [Enoch and Jaumard, 2018b], we proposed a decomposition formulation based on a *Lightpath* column, and we solved it using *Column Generation* technique. This *Lightpath* decomposition proved to be very efficient compared to previous algorithms. But, as we scaled up our problem instances, we sensed the efficiency limits of the *Lightpath* decomposition. In fact, the fine-granularity of the *Lightpath* columns makes it that the number of columns in the final ILP is very high, and can take very long to be solved. Consequently, we decided to explore a different decomposition formulation based on a column definition as a set of lightpaths referred to as a *Configuration* of lightpaths.

The *Configuration* decomposition has been previously explored in [Jaumard and Daryalal, 2017] using *Column Generation* and combining two algorithms to solve the *Pricing Problem*, one of

which is a *Path-based* ILP that is restricted to a pre-computed subset of paths and generates feasible *Configurations* quickly, and the other one is an optimal *Link-based* ILP used at the end of *Column Generation* to guarantee the optimality of the *Pricing Problem*. In the current study, we introduce a new formulation for the *Configuration* decomposition, and more importantly, we implement *Nested Column Generation* technique that aims to have only one *Path-based* ILP for the *Pricing Problem*, and solve it optimally and efficiently.

5.1 RSA Decomposition Model

5.1.1 Statement of the RSA Problem

We consider an *Elastic Optical Network* and we represent its topology by an undirected graph $G = (V, L)$ with node set V and link set L. The bandwidth spectrum of the optical network is sliced into a set S of frequency slots of width 12.5GHz. The traffic is defined by a set K of optical connections where each connection $k \in K$ is defined by a source (s_k), a destination (d_k) and a bandwidth demand D_k, expressed in terms of a number of slots.

Given an *Elastic Optical Network*, and a traffic demand matrix between its nodes, the RSA problem consists of finding optical channels for the traffic requests, as to maximize the overall throughput. For a given traffic request $k \in K$ from source node s_k to destination node d_k, with spectrum demand D_k, an optical channel (or lightpath) is a combination of a routing path Π_k from node s_k to node d_k, and an optical slice that is composed of a contiguous set of spectrum slots and that is continuous throughout the routing path Π_k. An optical slice is - by definition - a contiguous set of frequency slots, thus it can be characterized (in addition to its width D_k) by its low-end spectrum slot which we refer to as a *starting slot*.

5.1.2 RSA Decomposition Model

The ILP model is based on a *Configuration* decomposition. We define a *Configuration* as a set of link-disjoint lightpaths provisioning distinct requests and having the same starting slot. Figure 5.1 shows an example of a *Configuration* of three lightpaths.

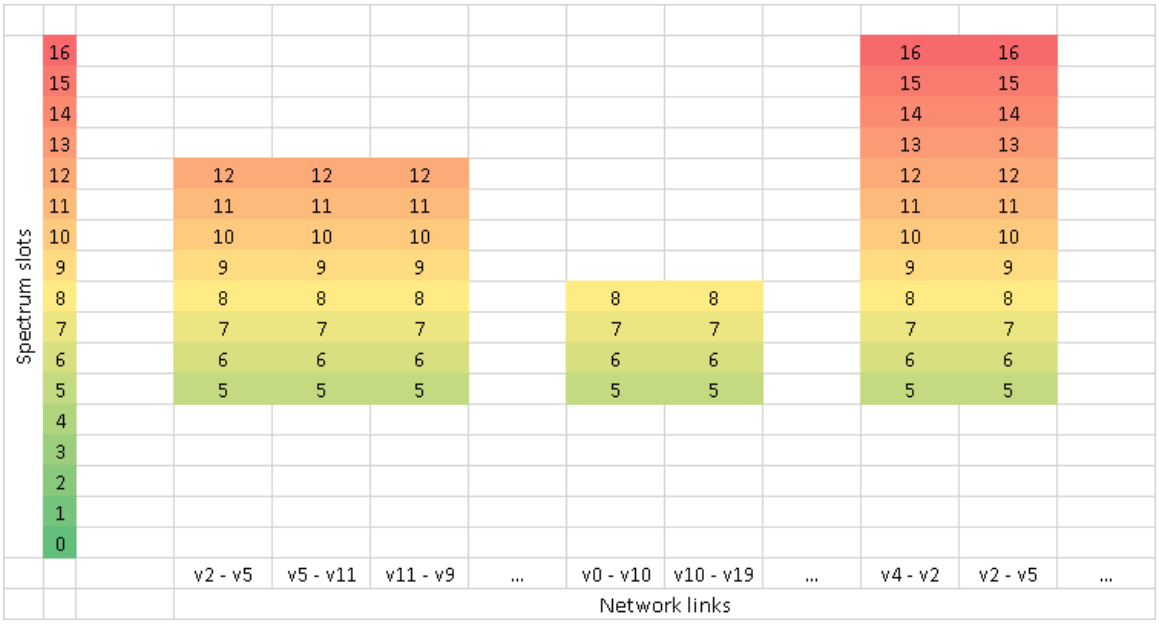


Figure 5.1: *Configuration* illustration
The first request connects node-pair $\{v_2, v_9\}$. It is routed using three links. Its demand is 8 slots. This configuration contains three lightpaths all of which start at slot 5.

Let C be the set of all possible *Configurations*. Each *Configuration* $c \in C$ is defined with the help of the following parameters:

- a_k^c: indicates if a request k is provisioned by *Configuration* c.
- b_ℓ^{sc}: indicates if one of the requests provisioned in *Configuration* c is routed on link ℓ while assigned slot s as part of its spectrum slice.

The model uses two sets of decision variables:

- z_c is a binary variable that indicates if *Configuration* c is selected in the solution.
- y_k is an intermediate binary variable that indicates if a request k is granted in the solution, and can be expressed in terms of z_c as follows:

$$y_k = \sum_{c \in C} a_k^c z_c \qquad k \in K. \tag{5.1}$$

The objective function consists of maximizing the throughput and is written in terms of y_k as follows:

$$z \quad = \quad \max \quad \sum_{k \in K} D_k y_k \tag{5.2}$$

subject to:

$$y_k \leq \sum_{c \in C} a_k^c z_c \qquad k \in K, \tag{5.3}$$

$$\sum_{c \in C} b_\ell^{sc} z_c \leq 1 \qquad \ell \in L, s \in S, \tag{5.4}$$

$$z_c \in \{0, 1\} \qquad c \in C, \tag{5.5}$$

$$0 \leq y_k \leq 1 \qquad k \in K. \tag{5.6}$$

Constraints (5.3) means that a request k is granted only if it is provisioned in at least one of the *Configurations* z_c. Constraints (5.4) make sure that each slot on each fiber link is not used by more than one *Configuration* z_c. The integrality of variables y_k is garanteed by the combination of (5.2) and (5.3), and needs not to be explicitly enforced in (5.6).

During the implementation of *Nested Column Generation* for RWA problem previously, we discussed that the two-variable-sets model making use of the intermediate variables is easier to solve than the model with only one set of variables. Following this inspiration, we designed the current model for RSA problem in the same spirit which proved to be fruitful. In fact, we have done experiments with *Nested Column Generation* using the compact model for RSA in [Jaumard and Daryalal, 2016], which we recall in Appendix B, and observed that the gap between the solution and the LP bound is very large which indicates that the solution is not close to the optimal solution.

While considering a model with two sets of variables, we have also examined a model where the intermediate variables are aggregated per node-pair ($y_{sd} = \sum_{\{v_s, v_d\} \in SD} a_{sd}^c z_c$) such that their number can be lesser. This prospect was unfruitful because the variables y_{sd} then had to be integer (whereas the variables y_k are binary), thus having a loose domain, which impacts the tightness of the model negatively.

In the sequel of this chapter, we describe the design and present the results of the model defined

above with intermediate binary variables y_k.

5.2 Solution Design

5.2.1 Nested Column Generation Processing Flow

Figure 5.2 shows the processing flow of our *Nested Column Generation* algorithm as applied to the RSA problem.

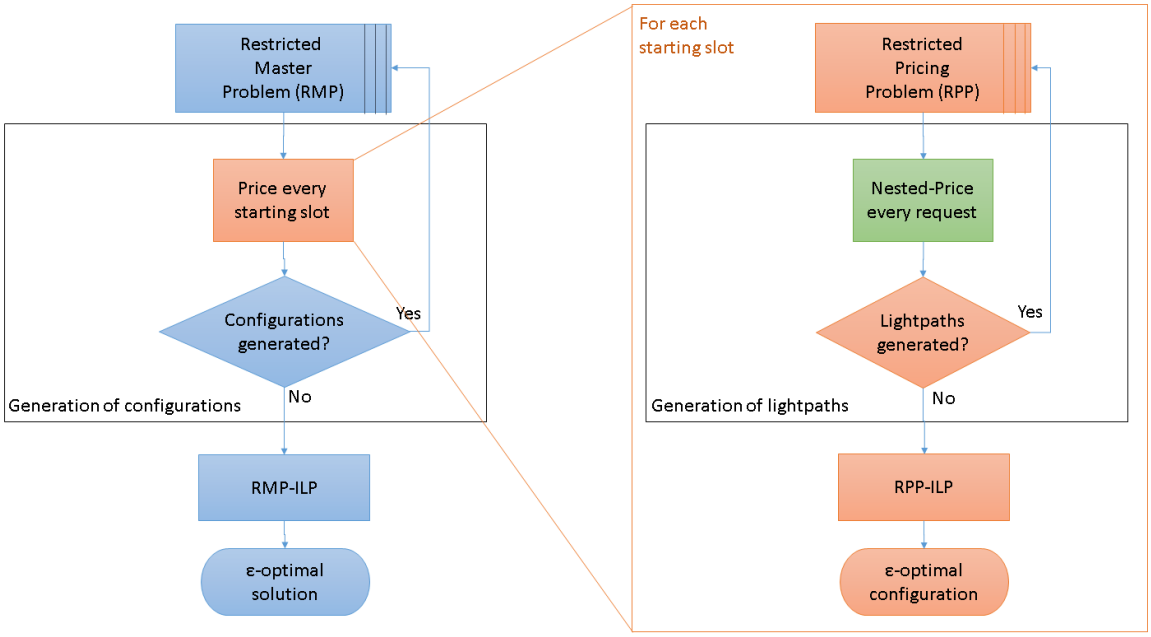


Figure 5.2: Nested Column Generation Flow for RSA

The *Column Generation* flow used to solve a Pricing problem, is identical to that used to solve the Master problem, thus the naming of *Nested Column Generation*.

The stop condition for the generation of *Configurations* is met when, after considering all the starting slots of the optical spectrum, the *Pricing Problem* cannot produce any improving *Configurations*.

5.2.2 Pricing Problem

As shown in Figure 5.2, the goal of the *Pricing Problem* is to compute an optimal *Configuration* for each starting slot s (for clear distinction from the starting slot, we will index the provisioning

slots by s'). In this section we also drop the *Configuration* index c in order to alleviate the notation.

Given that the starting slot is fixed for each *Pricing Problem*, the optical slice allocated to any request k can be deduced from the starting slot and the size of the request demand D_k. Therefore each *Pricing Problem* needs only to produce the routing paths for the lightpaths of the optimal *Configuration*. Let Π_k be the set of all possible routing paths of request k. A routing path $\pi \in \Pi_k$ of request k is defined with the help of a binary parameter δ_π^ℓ that indicates if link ℓ is part of the routing path π.

We define a set of decision variables for the *Pricing Problem* as follows:

- β_π is a binary variable that indicates if routing path π is included in the the optimal *Configuration*.

Let μ_k and $\mu_{\ell s'}$ be the dual values of constraints (5.3) and (5.4) respectively. The reduced cost for the model (5.2)-(5.6) is:

$$rc \quad = \quad \max \quad \sum_{k \in K} \mu_k a_k - \sum_{\ell \in L} \sum_{s' \in S} \mu_{\ell s'} b_\ell^{s'}. \tag{5.7}$$

The correspondence between the pricing solution and the master's parameters is as follows:

$$a_k = \sum_{\pi \in \Pi_k} \beta_\pi \qquad k \in K \tag{5.8}$$

$$b_\ell^{s'} = 0 \qquad \ell \in L, s' \notin \{s, \ldots, s + D_k^s - 1\} \tag{5.9}$$

$$b_\ell^{s'} = \sum_{k \in K} \sum_{\pi \in \Pi_k} \delta_\pi^\ell \beta_\pi \qquad \ell \in L, s' \in \{s, \ldots, s + D_k^s - 1\}. \tag{5.10}$$

Therefore, for a given starting slot s, the pricing problem reads as follows:

$$rc \quad = \quad \max \sum_{k \in K} \sum_{\pi \in \Pi_k} \left(\mu_k - \sum_{\ell \in L} (\sum_{s'=s}^{s+D_k^s-1} \mu_{\ell s'}) \delta_\pi^\ell \right) \beta_\pi \tag{5.11}$$

subject to:

$$\sum_{\pi \in \Pi_k} \beta_\pi \leq 1 \qquad k \in K \tag{5.12}$$

$$\sum_{k \in K} \sum_{\pi \in \Pi_k} \delta_\pi^\ell \beta_\pi \leq 1 \qquad \ell \in L \tag{5.13}$$

$$\beta_\pi \in \{0, 1\} \qquad k \in K, \pi \in \Pi_k. \tag{5.14}$$

Constraints (5.12) mean that each request is provisioned at most once, and constraints (5.13) mean that each link is traversed by at most one lightpath. This ensures that the lightpaths forming the *Configuration* are link-disjoint.

5.2.3 Lightpath Generator

Let ν_k and ν_ℓ be the values of the dual variables associated with constraints (5.12) and (5.13) respectively. Given a starting slot and given a request $k \in K$, the reduced cost is:

$$\max \quad \mu_k - \sum_{\ell \in L} (\sum_{s'=s}^{s+D_k^s-1} \mu_{\ell s'}) \delta^\ell - \nu_k - \sum_{\ell \in L} \nu_\ell \delta^\ell \tag{5.15}$$

which we streamline as:

$$\max \qquad (\mu_k - \nu_k) - \sum_{\ell \in L} \left((\sum_{s'=s}^{s+D_k^s-1} \mu_{\ell s'}) + \nu_\ell \right) \delta^\ell \tag{5.16}$$

We get a shortest-path problem with non-negative weights for each $k \in K$:

$$\min \qquad \sum_{\ell \in L} \left((\sum_{s'=s}^{s+D_k^s-1} \mu_{\ell s'}) + \nu_\ell \right) \delta^\ell. \tag{5.17}$$

5.3 Empirical Study

The datasets used in this empirical study use the Spain network topology with 21 nodes and 35 links, and the USA network topology with 24 nodes and 43 links.

The solution was implemented in Java. For the *Branch-and-Bound* part we used IBM CPLEX

solver 12.6 in a single thread mode. Throughout the LP phase, only the columns in the basis are retained, and the columns that drop from the basis are dropped from the current RMP, as suggested by Dantzig [1963], in order to maintain the size of the LP manageable, and force the algorithm to produce better columns in terms of low-fractionality.

It is known that linear programs' solvers (CPLEX in this paper) induce some very small numerical errors. For being so small, usually these errors are not a big hindrance in Column Generation algorithms. But in this study, these errors can be quite a problem. In fact, at the second level of our *Nested Column Generation*, the lightpath generator is not solved as a linear program but is solved as an undirected graph shortest-path problem using Dijkstra which does not allow negative link weights. But, while all dual values in this paper are supposed to be non-negative, CPLEX produces for some of them small negative values. In order to overcome this issue, we are rounding up to zero those negative dual values used in the link weights.

We observe for all the data instances that: When the *Pricing Problems* fail to generate new *Configurations* (i.e., $\tilde{rc}_{ILP} = 0$), we have also $rc^{\star}_{LP} = 0$, which implies that $rc^{\star}_{ILP} = 0$ meaning that the *Pricing Problems* are solved optimally. Therefore, the LP relaxation of the *Master Problem* $z^{\star}_{LP}$ is optimal and is an upper-bound on the optimal solution $z^{\star}_{ILP}$ (see discussion at Section 4.2.3).

5.3.1 ICTON dataset

This dataset was first introduced in [Jaumard and Daryalal, 2016] at the conference *ICTON'2016*, where it was solved using a *Column Generation* algorithm based on a *Configuration* decomposition (see Appendix B) which we will refer to as C_CG. This dataset was generated using the Spain network topology. Table 5.1 shows the results obtained using the current algorithm as well as a comparison with the results published in [Jaumard and Daryalal, 2016] and the results published in [Enoch and Jaumard, 2018b].

We did not re-implement the algorithm C_CG of Jaumard and Daryalal [2016] and the comparison is made with the results that were published by the authors of that algorithm. Considering that they produced their results on a 1.9-2.5GHz machine (in contrast with our 3.6-4.0GHz machine), it would be reasonable to divide their computation times by two for the purpose of the comparison (below) that remains approximative.

Table 5.1: ICTON dataset - Solution and Comparison

Traffic demand			Solution quality			ϵ (%) comparison			CPU comparison (sec)		
Offered load (Tbps)	|D|	|S|	$z^{\star}_{LP}$ (Tbps)	$\tilde{z}_{ILP}$ (Tbps)	GoS (%)	**C_NCG**	L_CG	C_CG	**C_NCG**	L_CG	C_CG
Spain network: demand granularities in {1, 2,, 8} slots, i.e., {25, 50, ..., 200} Gbps											
3.7	35	50	3.7	3.6	98.0	2.8	0.0	14.0	0.5	0.3	50.0
4.8	45	60	4.8	4.7	98.9	1.1	0.0	13.0	0.6	0.4	86.0
6.8	60	75	6.8	6.72	99.3	0.7	0.0	15.0	0.8	0.7	147.0
7.5	64	85	7.5	7.15	96.0	4.9	0.0	19.0	0.8	1.3	176.0
7.4	70	100	7.4	7.32	99.3	0.7	0.0	16.0	1.2	1.7	263.0
9.7	80	120	9.7	9.52	98.4	1.6	0.0	16.0	1.4	2.5	323.0
12.0	112	150	12.0	11.95	100.0	0.0	0.0	14.0	1.7	5.3	417.0
20.5	180	330	20.5	20.2	98.4	1.6	0.0	18.0	4.0	25.6	1606.0
Spain network: demand granularities in {2, 4,, 16} slots, i.e., {50, 100, ..., 400} Gbps											
7.5	35	80	7.5	7.4	99.3	0.7	0.0	10.0	0.8	0.9	134.0
9.8	45	110	9.8	9.7	99.5	0.5	0.0	10.0	0.9	2.0	177.0
10.7	60	156	10.7	10.65	99.5	0.5	0.0	12.0	0.9	3.1	261.0
15.5	64	170	15.5	15.5	100.0	0.0	0.0	16.0	1.4	4.7	630.0
15.1	70	236	15.1	15.1	100.0	0.0	0.0	13.0	1.4	7.8	1342.0
16.9	80	256	16.9	16.85	100.0	0.0	0.0	14.0	1.4	10.3	1419.0
Average						**1.1**	**0.0**	**14.3**	**1.3**	**4.8**	**502.2**

While the C_CG algorithm by Jaumard and Daryalal [2016] takes around 4 minutes on average, the new C_NCG algorithm takes only 1.3 seconds on average, beating even the L_CG algorithm by Enoch and Jaumard [2018b] at around 5 seconds on average.

Regarding the quality of the solution, while the L_CG algorithm solves this dataset to optimality, the C_NCG algorithm produces a solution accuracy around 1% on average, which is still a lot better than the accuracy of the C_CG algorithm of 14.3%.

In the next subsection, we will examine a dataset that is supposed to be much harder, and shall be more challenging for the new C_NCG algorithm.

5.3.2 INOC dataset

This dataset was first introduced in [Enoch and Jaumard, 2018b] at the conference *INOC'2017*, where it was solved using a *Column Generation* algorithm L_CG based on a *Lightpath* decomposition. This dataset was generated using both the Spain network and the USA network topologies according to the following specifications:

- Demands are drawn from granularities {4, 8, 16} in units of spectrum slots, with respective proportions {70%, 20%, 10%}, thus generating requests that are relatively large compared to those encountered in today's networks, hence producing instances with relatively a high level

of difficulty.

- Offered load is spread over all node pairs. Therefore, after aggregation, the number of requests given as input to our algorithm corresponds to the number of node pairs in the network.

In order to ease down the complexity of the algorithm and enhance its execution, we opted for the following strategies:

- Requests are aggregated per node-pair. This has the advantage of reducing the number of requests in the optimization problem and easing-down the solving process. Note that theoretically, if the problem is solved to optimality, this strategy would potentially produce a solution with a worse quality than if the requests were not aggregated.

- Given the large size of the instances of this dataset, the numerical accuracy of the LP solving becomes crucial in order for the *Column Generation* to converge. To that end, CPLEX parameter *NumericalEmphasis* [1] is turned on.

- We have noticed that the *Integer Linear Program (ILP)* phase can take a long time to cover the whole of the *Branch-and-Bound* tree, without improving by much the integral solution. Therefore we have chosen to pre-terminate the *Branch-and-Bound* by setting the CPLEX parameter *EpGap* [2] to 10^{-1}.

- We allow multi-threading during the *Branch-and-Bound* phase with up to 8 threads.

- Before the *Integer Linear Program (ILP)* phase, we set CPLEX parameter *Advance* [3] to 0 in order to enable the standard pre-solving operations instead of using the basis produced by the last iteration of *Column Generation* as a starting point for the *Branch-and-Bound*.

Table 5.2 shows the complete results obtained using the current algorithm in terms of the ϵ-optimal solution as well as the computation time. We observe that instances Spain90, USA80 and USA90 register a spike in the computation time as compared to the other instances. As we have

[1] Numerical precision emphasis

[2] Relative MIP gap tolerance: default: 1e-04.

[3] If set to 1 or 2, this parameter specifies that CPLEX should use advanced starting information when it initiates optimization.

Table 5.2: INOC dataset - Algorithm Results and Performance

Dataset				Solution quality				CPU time (sec)		
Instance	|S|	|D|	Load (Tbps)	$z^{\star}_{LP}$ (Tbps)	$\tilde{z}_{ILP}$ (Tbps)	ϵ (%)	GoS (%)	LP	ILP	Total
Spain network: Demands in {4,8, 16} slots, i.e., {100, 200, 400} Gbps										
Spain_50	400	413	50.2	50.2	42.6	17.8	84.9	30.5	0.3	31.0
Spain_60	400	495	60.0	60.0	47.2	27.1	78.7	113.6	0.2	114.1
Spain_70	400	578	70.2	70.2	52.3	34.2	74.5	177.0	1.0	178.2
Spain_80	400	660	80.0	80.0	55.5	44.1	69.4	254.1	2.2	256.6
Spain_90	400	743	**90.2**	**86.9**	56.0	55.1	62.1	1,756.9	940.1	2,697.7
USA network: Demands in {4,8, 16} slots, i.e., {100, 200, 400} Gbps										
USA_50	400	413	50.2	50.2	41.4	21.3	82.5	27.8	0.5	28.4
USA_60	400	495	60.0	60.0	45.6	31.6	76.0	350.4	3.1	353.8
USA_70	400	578	70.2	70.2	49.9	40.7	71.1	495.4	3.1	498.9
USA_80	400	660	**80.0**	**78.3**	53.2	47.2	66.5	1,441.2	204.2	1,646.3
USA_90	400	743	**90.2**	**85.7**	56.0	53.0	62.1	1,977.5	417.0	2,395.3

Table 5.3: INOC dataset - Solution and Performance Comparison

Dataset		Quality comparison						CPU comparison (sec)	
		$z^{\star}_{LP}$ (Tbps)		$\tilde{z}_{ILP}$ (Tbps)		ϵ (%)			
Instance	Load (Tbps)	**C_NCG**	L_CG	**C_NCG**	L_CG	**C_NCG**	L_CG	**C_NCG**	L_CG
Spain_50	50.2	50.2	50.2	42.6	48.0	17.8	4.6	31.0	22.4
Spain_60	60.0	60.0	60.0	47.2	57.8	27.1	3.8	114.1	28.1
Spain_70	70.2	70.2	70.2	52.3	66.6	34.2	5.4	178.2	42.2
Spain_80	80.0	80.0	80.0	55.5	73.6	44.1	8.7	256.6	71.9
Spain_90	**90.2**	**86.9**	86.9	56.0	75.0	55.1	15.8	2,697.7	1,131.1
USA_50	50.2	50.2	50.2	41.4	46.9	21.3	7.0	28.4	29.4
USA_60	60.0	60.0	60.0	45.6	55.0	31.6	9.1	353.8	55.8
USA_70	70.2	70.2	70.2	49.9	63.0	40.7	9.9	498.9	108.4
USA_80	**80.0**	**78.3**	78.3	53.2	65.5	47.2	19.6	1,646.3	1,122.0
USA_90	**90.2**	**85.7**	85.7	56.0	72.4	53.0	18.4	2,395.3	1,532.0
Average						**37.2**	**10.2**	**820.0**	**414.3**

discussed in [Enoch and Jaumard, 2018b], this is due to the fact that $z^{\star}_{LP}$ is lesser than the offered load. Consequently, the *Column Generation* needs a considerably larger number of columns to reach the $z^{\star}_{LP}$, and spends a relatively long time producing those columns. This dataset has been purposefully designed to showcase this type of difficult instances.

Table 5.3 shows a comparison with the results published in [Enoch and Jaumard, 2018b]. We can make the following observations:

- The upper-bound $z^{\star}_{LP}$ produced by the two algorithms is the same. A theoretical study of this upper-bound could be considered in a future work.
- The integral solution found by C_NCG algorithm is less good than the one found by the

L_CG algorithm and the accuracy gap is a lot larger. This is due to the fact that a *Configuration* column is a lot larger than a *Lightpath* column. Consequently, in the ILP phase, if the *Configuration* column has even a small overlapping with another column in the integral solution, it is rejected as a whole. To get an intuition of this phenomenon, we could make an analogy with the *Knapsack* problem. If we suppose that we have a fractional *Knapsack* solution and we want to get the integral solution from it: The smaller the items in the fractional solution, the lesser the contention among them, and the better the integral solution.

- The computation time of the C_NCG algorithm is less good than that of the L_CG algorithm (double on average). This is also due to the disparity of size of the columns in each decomposition. That said, the computation time of the C_NCG algorithm remains reasonable considering the level of difficulty of this dataset.

While the C_NCG algorithm is more scalable than the previous study, it can be inefficient in terms of solution quality for the largest data instances. We conclude that the algorithm L_CG, based on the *Lightpath* decomposition, presented in [Enoch and Jaumard, 2018b] remains the most efficient in terms of both solution quality and scalability.

Chapter 6

Conclusion

6.1 Contributions

Considering the provisioning of Optical Networks, we have explored the RWA optimization problem which is defined for the coarse fixed WDM frequency grid, and its successor the RSA problem which is defined for the fine flexible frequency grid. We have reviewed the literature for the two problems and observed the tendency to depart from compact formulations toward decomposition formulations most notably the *Configuration* decomposition. We aimed to explore ways to improve the latest results published for the two problems in [Jaumard and Daryalal, 2016, 2017]. In doing so, we explored two possibilities:

- A *Lightpath* decomposition that proved to produce the best results in terms of the accuracy of the integral solution as well as the efficiency in terms of computation time (up to two orders of magnitude faster than the latest existing study). This decomposition enabled us to publish two papers [Enoch and Jaumard, 2018b,a] for RSA and RWA problems respectively.

- When considering the same *Configuration* decomposition as the previous studies, we designed a *Nested Column Generation* algorithm. The idea behind this approach is to tame the difficulty of the *Pricing Problem* thus improving the overall performance. We applied this algorithm to the RWA problem which resulted in a considerable improvement of the computation time (9 times faster) but with a less good solution quality. We also applied this

algorithm to RSA problem which produced an accuracy near optimal for the reference dataset and improved the computation time by a factor of two orders of magnitude on average. We are currently in the process of preparing a journal publication with our findings about *Nested Column Generation*.

As an empirical contribution, in addition to the reference dataset for the RSA problem introduced in [Jaumard and Daryalal, 2016], we introduced a new dataset considerably larger and more difficult to solve, and ran our two approaches on it.

The overall observation is that the *Lightpath* decomposition produces the best solution with the best accuracy, and is the most efficient one in terms of computation time.

6.2 Future Work

Equipped with the two approaches that we presented in this thesis, the next studies could consider increasing the scope of the optimization problems by adding other dimensions that are useful for real-life implementations such as varying the modulation formats, or minimizing the number of optical regenerators needed in the optical network.

Appendix A

Guard-band Extension of the RSA Problem

When provisioning traffic demand in an *Elastic Optical Network*, a guard band of one slot is reserved on top of the spectrum slice of each request.

If we consider the case where traffic requests are not aggregated, this means that if there are multiple requests for the same node pair, these requests can be provisioned independently, thus, eventually taking different routing paths, and using spectrum slices that are not neighbors. If such requests happen to be routed on the same path using neighbors spectrum slices, they need not to be separated by a guard-band.

In order to account for this possibility, we extend our model as follows:

Given a node pair $w = \{u, v\}$, let $K_w = \{k_1, k_2, ..., k_n\}$ be the set of requests between u and v. We will refer to these requests as *atomic*. For $m \in \{1, 2, ..., n\}$, we denote ${K'}_w^m$ the set of combinations of m requests from K_w. We have

$$|{K'}_w^m| = \binom{n}{m}. \tag{A.1}$$

This quantity is the m^{th} binomial coefficient for power n. Let ${K'}_w = \cup_{m=1}^{m=n} {K'}_w^m$. We have

$$|{K'}_w| = \sum_{m=1}^{m=n} \binom{n}{m} = 2^n - 1. \tag{A.2}$$

Each combination of atomic requests produces a new derived request by summing up their demands. ${K'}_w$ is referred to as the set of *derived* requests. Note that $K_w \subset {K'}_w$ given that $K_w = {K'}_w^1$. The derived requests obtained for $m \in \{2, ..., n\}$ are referred to as *composite* requests. Let $K' = \cup_{w \in W} {K'}_w$. If a request k' is derived from an atomic request k, we note $k' \leftarrow k$.

For each node pair, we enumerate the derived requests using a binomial tree as in Figure A.1. Given this data structure's properties, this operation is done recursively as we go through the atomic requests of a given node pair. For each derived request, we keep track of the atomic requests used in its making.

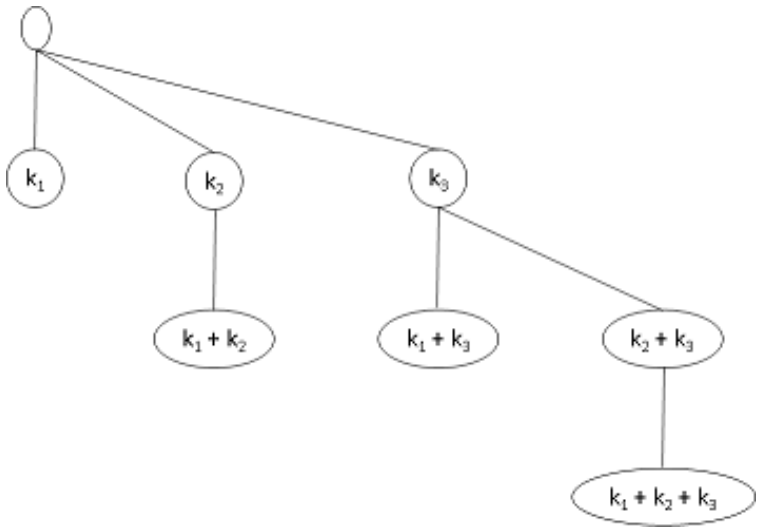


Figure A.1: A binomial tree of order n = 3 for a node pair with 3 atomic requests

In order to account for all possible derived requests, the ILP model in Enoch and Jaumard [2018b] (for example) is slightly modified as follows:

$$z = \max \sum_{c \in C} \left(\sum_{k' \leftarrow c} \sum_{k \rightarrow k'} D_k \right) z_c \tag{A.3}$$

Subject to:

$$\sum_{c \in C} \sum_{k' \leftarrow k} a^c_{k'} z_c \leq 1 \qquad k \in K \tag{A.4}$$

$$\sum_{c \in C} b^{sc}_{\ell} z_c \leq 1 \qquad \ell \in L, s \in S \tag{A.5}$$

$$z_c \in \{0, 1\} \qquad c \in C. \tag{A.6}$$

This modification means that the *Pricing Problem* needs to include all the *derived* requests. As calculated earlier, the number of *derived* requests is much larger than the number of *atomic* requests. Therefore, while this extension offers an elegant way to enumerate *derived* requests and modify the model accordingly, it remains unscalable for a practical dataset such as the one introduced in Enoch and Jaumard [2018b].

Appendix B

Previous RSA Configuration Decomposition Model

The model below was implemented in [Jaumard and Daryalal, 2016].

A configuration c is characterized by: a_k^c that is equal to 1 if request k is provisioned in c, 0 otherwise, and $a_{k,l}^{s,c}$ that is equal to one if k is provisioned with a lightpath going through link ℓ and slot s, 0 otherwise. The model is written as follows:

$$z = \max \sum_{c \in C} \left(\sum_{k \in K} D_k a_k^c \right) z_c \tag{B.1}$$

subject to:

$$\sum_{c \in C_s} z_c \leq 1 \qquad s \in S \tag{B.2}$$

$$\sum_{c \in C} a_k^c z_c \leq 1 \qquad k \in K \tag{B.3}$$

$$\sum_{c \in C} a_{k,l}^{s,c} z_c \leq 1 \qquad \ell \in L, s \in S \tag{B.4}$$

$$z_c \in \{0, 1\} \qquad c \in C. \tag{B.5}$$

Constraints (B.2) allow at most one configuration per starting slot for the consecutive slot blocks.

Constraints (B.3) ensure that each request is accepted at most once. Constraints (B.4) do not allow more than one lightpath going through a given link ℓ for a given slot s.

Appendix C

RWA Configuration Decomposition - One-variable-set Model

The model below was described in [Jaumard and Daryalal, 2017]. Each *Configuration* $c \in C$ is defined with the help of parameter a_{sd}^c which gives the number of connections from v_s to v_d that are provisioned by *Configuration* c.

The model uses a set of decision variables z_c which is an integer variable that gives the number of occurrences of *Configuration* c in the solution.

The ILP formulation reads as follows:

$$z = \max \sum_{c \in C} \left(\sum_{(v_s, v_d) \in SD} a_{sd}^c \right) z_c \tag{C.1}$$

subject to:

$$\sum_{c \in C} a_{sd}^c z_c \leq D_{sd} \qquad (v_s, v_d) \in SD \tag{C.2}$$

$$\sum_{c \in C} z_c \leq W \tag{C.3}$$

$$z_c \in \mathbb{Z}^+ \qquad c \in C. \tag{C.4}$$

where constraints (C.2) limit the number of granted connections to the demand, and constraints (C.3) ensure that the accepted *Configurations* are within the number of available wavelengths.

Appendix D

RWA network topologies

The data set used for RWA correspond to the network topologies: NSFNET, USANET, GERMANY, NTT, ATT, and BRAZIL. They are shown in the figures below where undirected lines represent bidirectional links.

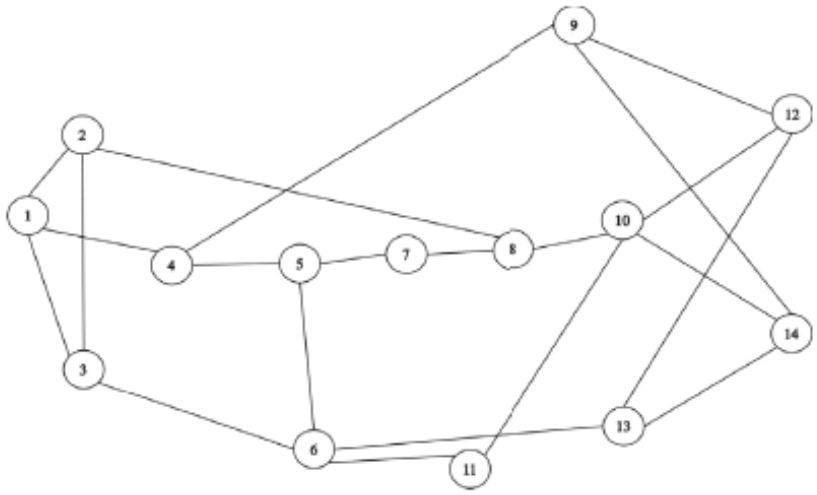


Figure D.1: NSF network topology

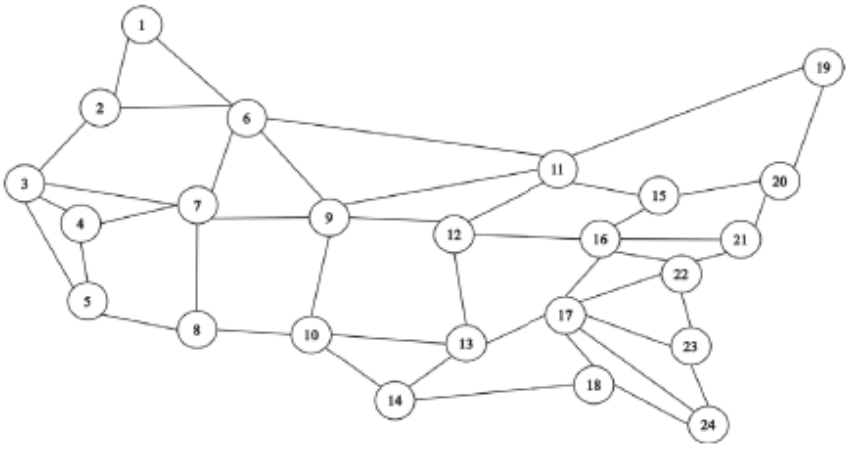


Figure D.2: USA network topology

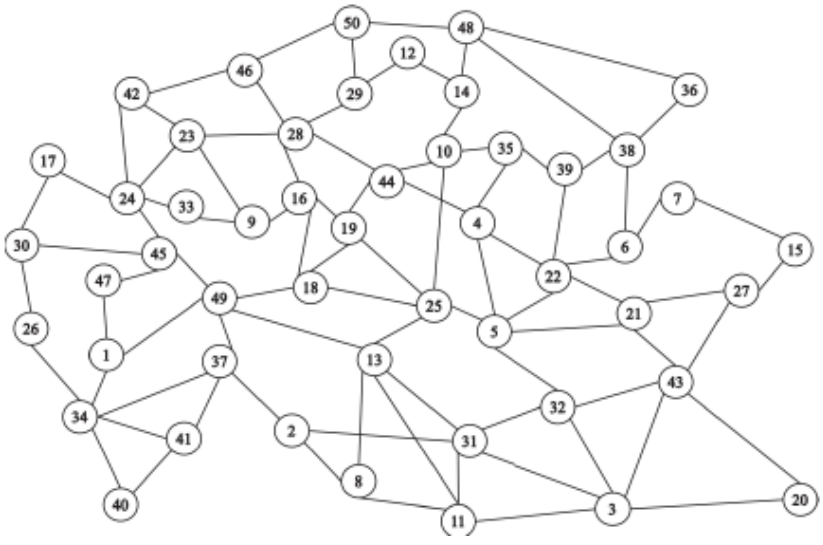


Figure D.3: NSF network topology

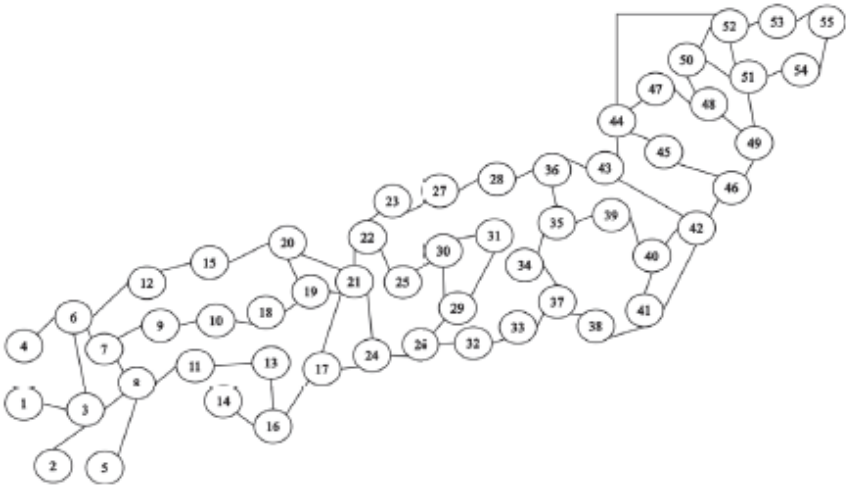


Figure D.4: NTT network topology

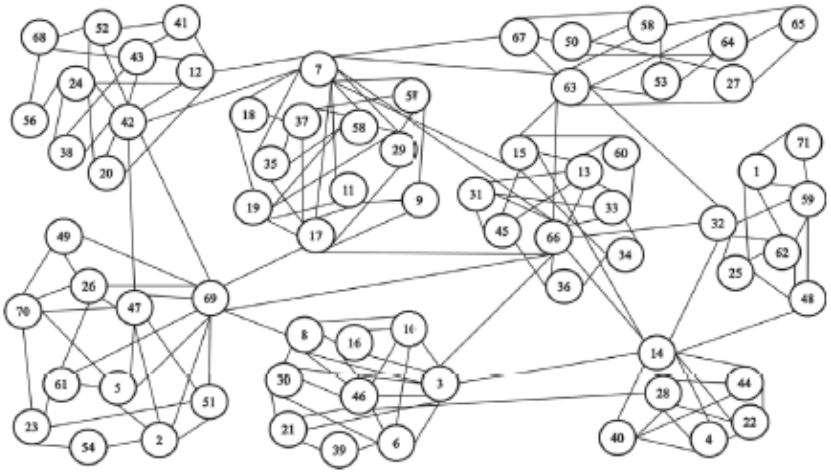


Figure D.5: ATT network topology

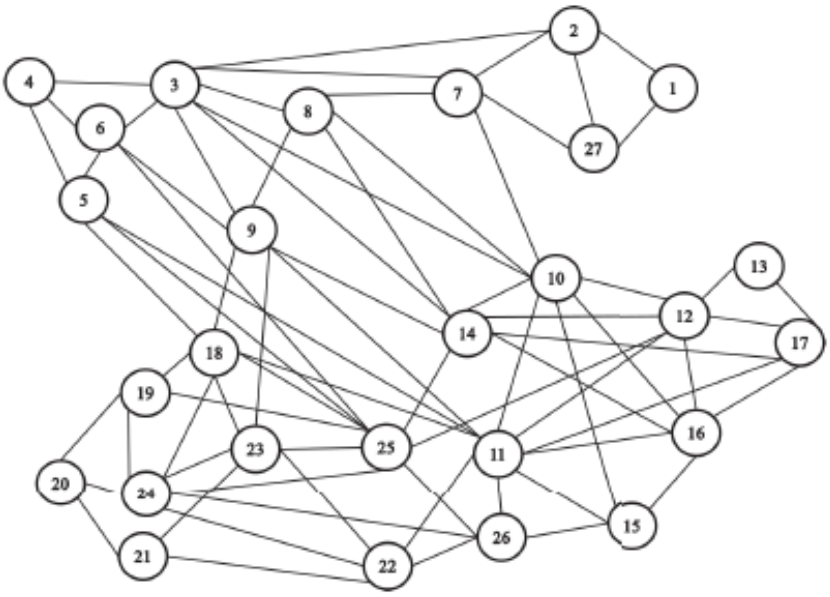


Figure D.6: Brazil network topology

Bibliography

Ahuja, R., Magnanti, T., and Orlin, J. (1993). *Network Flows: Theory, Algorithms and Applications*. Prentice Hall. 14

Barnhart, C., Johnson, E., Nemhauser, G., Savelsbergh, M., and Vance, P. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329. 11

Caprara, A., Furini, F., Malaguti, E., and Traversi, E. (2016). Solving the temporal knapsack problem via recursive dantzig–wolfe reformulation. *Information Processing Letters*, 116(5):379–386. 33, 35

Christodoulopoulos, K., Manousakis, K., and Varvarigos, E. (2010). Offline routing and wavelength assignment in transparent WDM networks. *IEEE/ACM Transactions on Networking*, 18:1557 – 1570. 4

Chvatal, V. (1983). *Linear Programming*. Freeman. 11, 13, 22, 32

Cisco (2017). 2017 annual report. White Paper. 1

Cordeau, J.-F., StojkoviÄĞ, G., Soumis, F., and Desrosiers, J. (2001). Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science*, 35(4):375–388. 34

Dantzig, G. (1963). *Linear Programming and Extensions*. Princeton University Press, Princeton. 33, 34, 46

Dohn, A. and Mason, A. (2013). Branch-and-price for staff rostering: An efficient implementation using generic programming and nested column generation. *European Journal of Operational Research*, 230(1):157 – 169. 34

Duhamel, C., Mahey, P., Martins, A., Saldanha, R., and de Souza, M. (2016). Model-hierarchical column generation and heuristic for the routing and wavelength assignment problem. *4OR*, pages 1 – 20. 8

Enoch, J. and Jaumard, B. (2018a). Enhanced rwa exact solution with a new lightpath decomposition algorithm. In *IEEE International Conference on Computing, Networking and Communications - ICNC*. 3, 7, 29, 37, 38, 51

Enoch, J. and Jaumard, B. (2018b). Towards optimal and scalable solution for routing and spectrum allocation. *Electronic Notes in Discrete Mathematics*, 64:335 – 344. 8th International Network Optimization Conference - INOC 2017. 3, 5, 19, 20, 39, 46, 47, 49, 50, 51, 54, 55

Gilmore, P. C. and Gomory, R. E. (1961). A linear programming approach to the cutting stock problem. *Operations Research*, 9:849–859. 10

Glassey, C. R. (1973). Nested decomposition and multi-stage linear programs. *Management Science*, 20(3):282–292. 33

Hennig, F., Nygreen, B., and Lübbecke, M. (2012). Nested column generation applied to the crude oil tanker routing and scheduling problem with split pickup and split delivery. *Naval Research Logistics*, 59:298–310. 33, 35

Ho, J. K. and Manne, A. S. (1974). Nested decomposition for dynamic models. *Mathematical Programming*, 6(1):121–140. 33

IBM (2016a). Cplex optimization studio, version 12 release 7. https://www.ibm.com/bn-en/marketplace/ibm-ilog-cplex. 16

IBM (2016b). *CPLEX User's Manual, version 12 Release 7*. IBM Corporation. 16

IUT (2012). Spectral grids for WDM applications: DWDM frequency grid. White Paper. 1

Jaumard, B. and Daryalal, M. (2016). Scalable elastic optical path networking models. In *International Conference on Transparent Optical Networks - ICTON*, pages 1–4. 3, 25, 26, 42, 46, 47, 51, 52, 56

Jaumard, B. and Daryalal, M. (2017). Efficient spectrum utilization in large scale RWA problems. *IEEE/ACM Transactions on Networking*, pages 1 – 16. 3, 5, 6, 8, 15, 16, 19, 29, 31, 32, 37, 39, 51, 58

Jaumard, B., Meyer, C., and Thiongane, B. (2007). Comparison of ILP formulations for the RWA problem. *Optical Switching and Networking*, 4(3-4):157–172. 4, 8

Jaumard, B., Meyer, C., and Thiongane, B. (2009). On column generation formulations for the RWA problem. *Discrete Applied Mathematics*, 157:1291–1308. 4, 5, 9

Jaumard, B., Meyer, C., and Yu, X. (2006). How much wavelength conversion allows a reduction in the blocking rate ? *Journal of Optical Networking*, 5(12):881–900. 8

Klinkowski, M., Pioro, M., Zotkiewicz, M., Ruiz, M., and Velasco, L. (2014). Valid inequalities for the routing and spectrum allocation problem in elastic optical networks. In *International Conference on Transparent Optical Networks - ICTON*, pages 1–5. 6

Klinkowski, M. and Walkowiak, K. (2015). A simulated annealing heuristic for a branch and price-based routing and spectrum allocation algorithm in elastic optical networks. In *International Conference on Intelligent Data Engineering and Automated Learning (IDEAL)*, pages 290–299. 6

Klinkowski, M., Zotkiewicz, M., Walkowiak, K., Pioro, M., Ruiz, M., and Velasco, L. (2016). Solving large instances of the RSA problem in flexgrid elastic optical networks. *IEEE/OSA Journal of Optical Communications and Networking*, 8:320 – 330. 6

Lee, K., Kang, K., Lee, T., and Park, S. (2002). An optimization approach to routing and wavelength assignment in WDM all-optical mesh networks without wavelength conversion. *ETRI Journal*, 24(2):131–141. 5, 9

Lee, T., Lee, K., and Park, S. (2000). Optimal routing and wavelength assignment in WDM ring networks. *IEEE Journal on Selected Areas in Communications*, 18(10):2146–2154. 5

Liu, Z. and Rouskas, G. (2012). A fast path-based ILP formulation for offline RWA in mesh optical

networks. In *IEEE Global Telecommunications Conference - GLOBECOM*, pages 3014–3019. 4, 9

Margot, F. (2010). *Symmetry in Integer Linear Programming*, pages 647–686. Springer, Berlin, Heidelberg. 5

Muciaccia, T. and Passaro, V. (2017). Future scenarios for software-defined metro and access networks and software-defined photonics. *Photonics*, 4:1 – 27. 7

Naveh, B. (2016). Jgrapht - a free java graph library. http://jgrapht.org/. 16

Noronha, T. and Ribeiro, C. (2006). Routing and wavelength assignment by partition coloring. *European Journal of Operational Research*, 171(3):797–810. 8

Ramaswami, R. and Sivarajan, K. (1995). Routing and wavelength assignment in all-optical networks. *IEEE/ACM Transactions on Networking*, 5(3):489–501. 5

Ruiz, M., Pioro, M., Zotkiewicz, M., Klinkowski, M., and Velasco, L. (2013a). Column generation algorithm for RSA problems in flexgrid optical networks. *Photonic Network Communications*, 26:53–64. 24

Ruiz, M., Zotkiewicz, M., Velasco, L., and Comellas, J. (2013b). A column generation approach for large-scale RSA-based network planning. In *International Conference on Transparent Optical Networks - ICTON*, pages 53–64. 6

Saad, M. and Luo, Z.-Q. (2002). A Lagrangean decomposition approach for the routing and wavelength assignment in multifiber WDM networks. In *IEEE Global Telecommunications Conference - GLOBECOM*, pages 2818–2822. 4

Saleh, A. and Simmons, J. (2011). Technology and architecture to enable the explosive growth of the internet. *IEEE Communications Magazine*, 49:126 –132. 7

Saleh, A. and Simmons, J. (2012). All-optical networking - evolution, benefits, challenges, and future vision. *Proceedings of the IEEE*, 100(5):1105 –1117. 7

Shirazipourazad, S., Zhou, C., Derakhshandeh, Z., and Sen, A. (2013). On routing and spectrum allocation in spectrum-sliced optical networks. In *Proceedings IEEE INFOCOM*, pages 385–389. 39

Simmons, J. M. (2017). Technology and architectural approaches to address continued explosive growth in network traffic. In *IEEE International Conference on Computing, Networking and Communications - ICNC*. 7, 8

Tilk, C., Drexl, M., Irnich, S., et al. (2018). Nested branch-and-price-and-cut for vehicle routing problems with multiple resource interdependencies. Technical report. 33, 35

Vanderbeck, F. (2001). A nested decomposition approach to a three-stage, two-dimensional cutting-stock problem. *Management Science*, 47(6):864–879. 33

Vignac, B., Vanderbeck, F., and Jaumard, B. (2016). Reformulation and decomposition approaches for traffic routing in optical networks. *Networks*, 67(4):277–298. 33

Zang, H., Jue, J., and Mukherjee, B. (2000). A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks. *Optical Networks Magazine*, 1(1):47–60. 8