A Machine Learning-based Optimization Algorithm for the Survivable Constrained-Routing and Spectrum Assignment Problem

Ibrahima Diarrassouba[1], Youssouf Hadhbi[2], A. Ridha Mahjoub[3,4], and Suja Abou Khamseen[3]

[1] LMAH, FR CNRS 3335, Le Havre Normandie University, 76600 Le Havre, France

diarrasi@univ-lehavre.fr

[2] Orange Innovation, Châtillon, France

youssouf.hadhbi@orange.com

[3] Departement of Statistics and Operations Research, College of Science, Kuwait University, Kuwait,

suja.aboukhamseen@ku.edu.kw, ridha.mahjoub@ku.edu.kw

[4] LAMSADE, UMR CNRS 7243, Paris-Dauphine PSL University, 75775 Paris Cedex 16, France

ridha.mahjoub@lamsade.dauphine.fr

Abstract. This paper deals with a variant of the well known Routing and Spectrum Assignment problem (RSA), namely the Survivable Constrained-Routing and Spectrum Assignment (SC-RSA). The SC-RSA is NP-hard and more challenging than the RSA which is already known to be NP-hard. The main purpose of this paper is to devise an exact cutting plane and column generation based algorithm for solving the problem. To this end, we first give an extended formulation called *path formulation*. This is based on a new integer linear programming formulation which contains a huge number of variables. For this, we use a column generation technique to approach its linear relaxation. We propose a pseudo-polynomial time dynamic programming based algorithm for solving the related pricing problem which is shown to be equivalent to the well known resource constrained shortest path problem. Moreover, we identify various families of valid inequalities for the associated polytope and devise separation routines. Using these results, we develop a cutting planes based algorithm for the problem along with an extensive computational study is presented.

Furthermore, we discuss a machine learning approach to learn columns and cuts selection strategies via reinforcement learning and supervised learning. The proposed approach has shown to be able to solve more instances to optimality and accelerate the computational time of the algorithm.

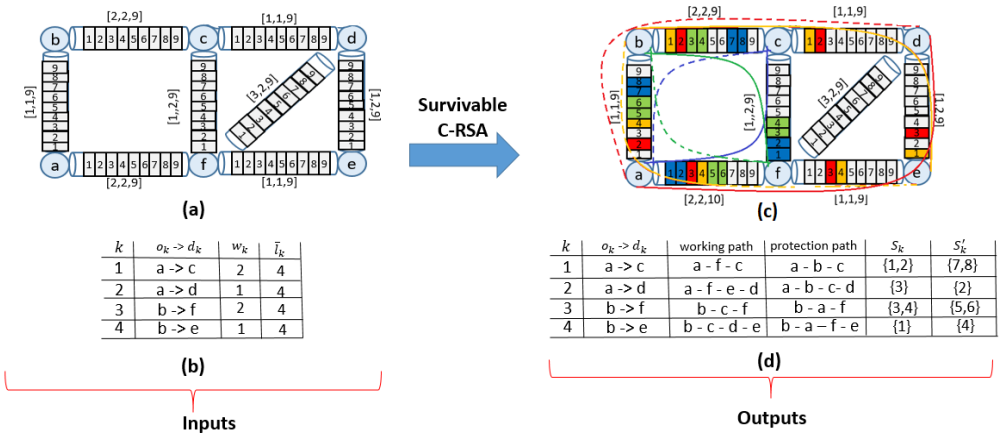
Keywords: 5*G* Network design, routing and spectrum assignment, survivability, dedicated-protection, valid inequality, separation, column generation, branch-and-cut-and-price, reinforcement learning.

1 Introduction

Nowadays, optical transport networks are facing a significant challenge posed by the exponential growth of data and networking. In response, Spectrally Flexible Optical Networks (SFON) have emerged as a promising architecture due to their flexibility, scalability, efficiency, reliability, and survivability, offering advantages over traditional Fixed-Grid Optical Wavelength Division Multiplexing (WDM) networks. Addressing the problem of survivability in SFONs is of great interest. Indeed survivability becomes an issue in the network design of telecommunication networks with the introduction of optical fiber technology. Telecommunication networks became sparse (like trees), and having very high capacity. So an increasing amount of information is going through the network, and any failure may yield a big loss of traffic. That is why having networks sufficiently survivable is now the first concern in network design. Survivability is the hability to restore the network in the event of a failure. This paper deals with the Survivable Constrained-Routing and Spectrum Assignment (SC-RSA) problem, which is a variant of the Routing and Spectrum Assignment (RSA) problem. In this context, we consider the dedicated protection routing scheme, where a protection path and capacity are provisioned in advance for each traffic demand. This variant is harder than the RSA problem, which is already known to be NP-hard [47] [49]. For the SC-RSA problem, an optical spectrum of $\bar{s} \in \mathbb{Z}_+$ available contiguous frequency slots is considered, denoted by $\mathbb{S} = \{1, \ldots, \bar{s}\}$. A SFON topology is represented by an undirected, loopless, and two-edge-connected graph $G = (V, E)$, where each edge e has an associated length $\ell_e \in \mathbb{R}_+$ (in kilometers) and cost $c_e \in \mathbb{R}+$. There are $\bar{s} \in \mathbb{N}_+$ contiguous slots numbered from 1 to $\bar{s}$ over each edge $e \in E$. Additionally, a set of non-splittable traffic demands, denoted by K, is given. Each demand $k \in K$ is characterized by its pair origin and destination (o_k, d_k), a slot-width $w_k \in \mathbb{Z}_+$, and a delay $\bar{\ell_k} \in \mathbb{R}_+$ (in kms). The SC-RSA problem consists in determining for each demand $k \in K$ two edge-disjoint (o_k,d_k)-paths, respectively called *working* path P_k and *protection* path P'_k (*2-connectivity and non-splittable demands constraints*) whose total length does not exceed $\bar{\ell_k}$ (*delay constraint*). Additionally, for each demand $k \in K$, two intervals S_k and S'_k of w_k consecutive frequency slots should be selected along its working and protection paths respectively (*contiguity and continuity constraints*). If two demands have their working (protection) paths non disjoint, then they cannot share any slot over the common edges (*non-overlapping constraints*). The problem aims at minimizing the total cost of the edges selected for routing the demands.

Fig. 1 illustrates a feasible solution for an instance of the SC-RSA problem. The instance involves 4 demands routed in a graph G with 6 nodes and 8 edges. Each edge e is characterized by a triplet

$[l_e, c_e, \bar{s}]$, with $\bar{s} = 9$. Figures 1(c) and 1(d) display the selected routes (working and protection paths) and the corresponding spectrums for each demand.



k	o_k -> d_k	w_k	$\bar{l}_k$
1	a -> c	2	4
2	a -> d	1	4
3	b -> f	2	4
4	b -> e	1	4

k	o_k -> d_k	working path	protection path	S_k	S'_k
1	a -> c	a - f - c	a - b - c	{1,2}	{7,8}
2	a -> d	a - f - e - d	a - b - c- d	{3}	{2}
3	b -> f	b - c - f	b - a - f	{3,4}	{5,6}
4	b -> e	b - c - d - e	b - a – f - e	{1}	{4}

Fig. 1. The set of established working and protection paths (see Fig. 1(c) and 1(d)), along with their spectrums, in graph G (see Fig. 1(a)) for the demands k_1, k_2, k_3, k_4 as defined in Figure 1(b).

2 Related works

The Routing and Spectrum Assignment (RSA) problem is known to be crucial in the design and the dimensioning of spectrally flexible optical networks. The RSA problem has been shown to be NP-hard [47] [49] and more complex than the historical Routing and Wavelength Assignment (RWA) problem [22]. Over the years, various algorithms and (mixed) integer linear programming (ILP) formulations have been proposed to solve this problem. There are two main classes of ILP formulations used to solve the RSA problem: *edge-path* and *edge-node* formulations. The edge-path formulation, which is commonly used in the literature, involves variables associated with the possible physical optical paths between each origin and destination pair in the graph. However, this approach leads to an exponential growth in the number of variables and constraints as the instance size increases, including the number of demands, the number of slots [20]. Notice that several papers have used the edge-path formulation to solve the RSA problem, but they often rely on precomputed paths without guaranteeing the optimality as pointed out by Christodoulopoulos et al. [7], Klinkowski et al. [38], Velasco et al. [50], Zotkiewicz et al. [54], and Salameh et al. [44]. On the other hand, column generation approach has been employed by Klinkowski et al. [43], and Jaumard et al. [24]. To further enhance the linear relaxation bounds, Klinkowsky et al. proposed [39] a class of valid inequalities induced by cliques. They used a Branch-and-Bound (B&B) approach for separating them.

In addition, Klinkowski et al. [40] introduced a Branch-and-Cut-and-Price (B&C&P) method based on an edge-path formulation for the RSA problem. More recently, decomposition approaches have been utilized by Fayez et al. [15], and Xuan et al. [51], where the RSA problem was solved in two steps. First, the routing and then the spectrum assignment (R+SA) are considered, using a recursive algorithm based on an ILP edge-path formulation.

On the other hand, and to deal with the limitation of the edge-path formulation, an alternative compact edge-node formulation has been introduced [3], [50], [54]. That formulation has been the subject of comparative studies by Bertero et al. [1], who proposed new ILP formulations.

Due to the NP-hardness of the RSA problem, several heuristics [23][31], greedy algorithms [30], metaheuristics [21][22][40][27], and hybrid meta-heuristic approaches [42] have been proposed to handle large-sized instances of the RSA problem. Furthermore, recent studies have started incorporating artificial intelligence [41] [28][29], deep-learning [5] and machine-learning [19][45][52][53] to improve performance.

In this paper, we are interested in the resolution of a new complex variant of the so-called Constrained-Routing and Spectrum Assignment (C-RSA) problem which is known as a hard variant of the RSA problem. This problem can also be viewed as a new variant of the so-called Routing, Modulation, and Spectrum Assignment (RMSA) problem [29], where the modulation formats (also known as transponder configurations) are preselected for each traffic demand. The C-RSA problem uses additional constraints, such as the delay constraint, which ensures that the chosen path for each traffic demand does not exceed a certain length. However, only a few works have taken that constraint into account. Hadhbi et al. [20] developed a novel cut-based model for the C-RSA problem. This exhibits a polynomial number of variables and an exponential number of constraints separable in polynomial time using network flows. Computational results have shown that the proposed cut formulation can handle larger instances compared to other approaches as those which have been proposed by Velasco et al. [50] and Cai et al. [3]. Note that Velasco et al. [50], Cai et al. [3] and Bertero et al. [1] have not taken into account the delay constraint.

Recently, Da Silva et al. [48] proposed a new framework for the C-RSA based on an edge-path formulation combined with a Multi-Commodity Flow approach. Additionally, Diarrassouba et al. [10][12] proposed an extendted formulation and a Branch-and-Cut-and-Price algorithm for solving the C-RSA problem. Their approach has shown to outperform a Branch-and-Cut algorithm previously presented by Diarrassouba and Hadhbi [11].

On the other hand, and as mentionned before, in recent years, telecommunication networks have experienced significant growth in demand for bandwidth. Sometimes the bandwidth demand is huge

in such a way that if there is a failure in certain resources in the network, then this might cause a serious interruption in data transmission. All the traffic demands using these failed resources will be affected and will cause a significant amount of data loss. This highlights the importance of implementing efficient protection mechanisms to minimize the impact of such failures on data transmission. For this, network operators have had to adapt their network architecture and management to ensure continuous data transmission, especially when faced to physical issues that may yield failures. There are two main types of physical failures that can occur in these networks: node failures and fiber-link failures. Understanding these types of failures is important for maintaining the survivability of telecommunication networks and ensuring uninterrupted data transmission. Protection and restoration are two commonly used techniques for recovering from failure in optical networks. We destinguish four schemes:

- *proactive scheme*: in this scheme, protection paths are precalculated and reserved in advance. In the event of a failure, the affected demands are rerouted through their preplanned protection paths. This approach allows a quick and efficient recovery as the protection paths are already available.
- *reactive scheme*: in contrast to the proactive scheme, protection paths are calculated after a failure occurs. When a failure is detected, new protection paths are determined and the affected demands are rerouted through these paths. This scheme may take more time to recover as the calculation of protection paths is done in response to the failure.
- *link based scheme*: in this scheme, when a failed link is detected, the traffic is rerouted through alternative paths that bypass the failed link. This approach minimizes the impact of the failure on the affected demands.
- *path based scheme*: along this scheme, protection paths are reserved for every group of demands sharing the same link. If the link fails, the demands of the group are rerouted through the reserved protection paths.

In this study, we focus on the proactive and path based schemes. For this, we ensure that both the working paths (the original paths) and the protection paths (reserved paths for rerouting in case of failures) are edge-disjoint. That is to say these paths do not share any edge, which enhances the network's resilience against fiber-link failures. There are two main categories of protection scheme: *dedicated backup path protection* (DBPP) and *shared backup path protection* (SBPP) [6]. In the DBPP scheme, each working path has a dedicated alternate path (called protection path) having no link in common. This ensures that if the working path fails, the traffic can be rerouted through the dedicated backup one. However, in the SBPP scheme, traffic demands that share common

links can utilize shared resources (slots) on their protection paths. This allows for more efficient utilization of resources, particularly in terms of spectrum utilization, but it may result in less safety, compared to the DBPP scheme. Researchers in the field of elastic optical networks and optimization have studied both, the DBPP [18] and the SBPP [4]. They have explored various aspects of these schemes, proposed some ILP formulations, and devised efficient heuristics and metaheuristics.

It has been observed that the exact algorithms proposed in the literature have faced challenges in effectively solving large-sized instances of the RSA and C-RSA problems with and without survivability. We believe that a cutting-plane-based approach could be a powerful tool as already shown in [10][11][12] when solving the C-RSA without considering the survivability aspect. To the best of our knowledge, such an approach has not been explored yet for the survivable C-RSA problem. Therefore, the main objective of this work is to thoroughly investigate the theoretical properties of the survivable C-RSA problem. Our aim is to provide a comprehensive polyhedral analysis of the problem and, based on this, develop a cutting-plane and column generation algorithm, to solve it. To this end, we first give an extended formulation based on an ILP called *path formulation*. We then identify several classes of valid inequalities for the associated polytope and propose separation routines. Moreover, we devise a column generation algorithm to solve its linear relaxation. Based on this, we develop a Branch-and-Cut-and-Price algorithm for the problem, and present detailed computational results. Furthermore, we incorporate some reinforcement learning techniques to improve the performance of our approach. The reinforcement learning is applied to select and generate promising columns and cuts based on the current state of the problem. This approach has the potential to significantly accelerate the resolution and find high-quality solutions.

In what follows we give more definitions and notations. A *path* P is a sequence of edges in G. If u and v are respectively the first and last nodes of P, then u and v are called the *extremities* of P. A path P is said to be *elementary* if no node of P is met more than once. The set of edges of path P is denoted by $E(p)$. For an edge $e \in E$, $P(e)$ represents the set of paths that pass through e in G. A subset of nodes C in G is called a *clique* if every pair of nodes in C are linked by an edge. In other words, the induced subgraph on C is complete. Given two nodes s and t in G, a $(s,t)-$*path* is a path between s and t. Given a subset of nodes $W \subset V$, the subgraph of G induced by W is the graph $(W, E(W))$ where $E(W)$ is the set of edges having both endnodes in W. A cycle in G is a path whose extremities coincide. An interval $I = [s_i, s_j]$ of slots is a set of contiguous frequency slots, numbered from s_i to s_j, where $j \geq i+1$ and $s_j \leq \bar{s}$. For $X \subset V$ with $X \neq \emptyset$, we let $\delta(X)$ denote the *cut* induced by X, that is the set of edges having one extremity in X and the other in

$\overline{X} = V \setminus X$.

The remainder of this paper is organized as follows. In Section 3, we introduce the path formulation. In Section 4, we provide several valid inequalities for the problem. In Section 5, the column generation algorithm is dicussed. In Section 6, the Branch-and-Cut-and-Price algorithm is described. In Section 7, we present a reinforcement learning approach for the problem. Computational results are presented in Section 8 to evaluate our approach. Finally, some concluding remarks and future outlook are given in Section 9.

3 Extended formulation

In this section, we introduce an extended formulation for the problem based on the so-called *path* variables. For this, we consider for each $k \in K$, $P \in P^k$ and $s \in \mathbb{S}$, a binary variable $y^k_{p,s}$ (resp. $f^k_{p,s}$) which takes 1 if slot s is the last-slot allocated along the working path (resp. protection path) P of demand k and 0 if not. Here, P^k denotes the set of all feasible (o_k, d_k)-paths in G, i.e., P^k contains each (o_k, d_k)-path in G with a total length less than $\bar{\ell}_k$. For $k \in K$ and $e \in E$, let x^k_e (resp. b^k_e) be a binary variable which takes 1 if the working path (resp. protection path) of demand k uses the edge e and 0 if not. And for each $k \in K$, $e \in E$ and $s \in \mathbb{S}$, let $t^k_{e,s}$ be a binary variable which takes 1 if the slot s is the last-slot assigned to demand k over edge e by its working or protection path and 0 if not.

The SC-RSA problem is equivalent to the following integer program

$$\min \sum_{k\in K}\sum_{e\in E} c_e x^k_e + \sum_{k\in K}\sum_{e\in E} c_e b^k_e, \tag{1}$$

$$\sum_{e\in E}\sum_{s=1}^{w_k-1} t^k_{e,s} + \sum_{p\in P^k}\sum_{s=1}^{w_k-1} (y^k_{p,s} + f^k_{p,s}) = 0, \forall k \in K, \tag{2}$$

$$\sum_{p\in P^k}\sum_{s=w_k}^{\bar{s}} y^k_{p,s} = 1, \forall k \in K, \tag{3}$$

$$\sum_{p\in P^k}\sum_{s=w_k}^{\bar{s}} f^k_{p,s} = 1, \forall k \in K, \tag{4}$$

$$\sum_{p\in P^k(e)}\sum_{s=w_k}^{\bar{s}} y^k_{p,s} \leq x^k_e, \forall k \in K, \forall e \in E, \tag{5}$$

$$\sum_{p\in P^k(e)}\sum_{s=w_k}^{\bar{s}} f^k_{p,s} \leq b^k_e, \forall k \in K, \forall e \in E, \tag{6}$$

$$\sum_{p\in P^k(e)} (y^k_{p,s} + f^k_{p,s}) \leq t^k_{e,s}, \forall k \in K, \forall e \in E, \forall s \in \mathbb{S}, \tag{7}$$

$$x_e^k + b_e^k \leq 1, \forall k \in K, \forall e \in E, \tag{8}$$

$$\sum_{s=w_k}^{\bar{s}} t_{e,s}^k = x_e^k + b_e^k, \forall k \in K, \forall e \in E, \tag{9}$$

$$\sum_{e \in E} l_e x_e^k \leq \bar{\ell}_k, \forall k \in K, \tag{10}$$

$$\sum_{e \in E} l_e b_e^k \leq \bar{\ell}_k, \forall k \in K, \tag{11}$$

$$\sum_{k \in K} \sum_{s'=s}^{\min(s+w_k-1,\bar{s})} t_{e,s'}^k \leq 1, \forall e \in E, \forall s \in \mathbb{S}, \tag{12}$$

$$x_e^k, b_e^k \geq 0, \forall k \in K, \forall e \in E, \tag{13}$$

$$y_{p,s}^k \geq 0, \forall k \in K, \forall p \in P^k, \forall s \in \mathbb{S}, \tag{14}$$

$$f_{p,s}^k \geq 0, \forall k \in K, \forall p \in P^k, \forall s \in \mathbb{S}, \tag{15}$$

$$y_{p,s}^k \in \{0,1\}, \forall k \in K, \forall p \in P^k, \forall s \in \mathbb{S}. \tag{16}$$

$$f_{p,s}^k \in \{0,1\}, \forall k \in K, \forall p \in P^k, \forall s \in \mathbb{S}, \tag{17}$$

where $P^k(e)$ denotes the set of all (o_k,d_k) paths of demand k going through edge e in G.

Equations (2) force each slot $s \in \{1, ..., w_k - 1\}$ to not be used as last-slot for demand k along its working and protection paths. Inequalities (3) (resp. (4)) express the fact that exactly one slot $s \in \{w_k, \ldots, \bar{s}\}$ is assigned as last-slot for demand k, and exactly one single working (resp. protection) path from P^k is allocated to each demand $k \in K$. Inequalities (5), (6) and (7) indicate the use of an edge $e \in E$ by demand k, and the assignment of slot s as last-slot to demand k over e. Inequalities (8) express the 2-connectivity constraints. Equations (9) ensure that exactly one slot $s \in \{w_k, \ldots, \bar{s}\}$ is assigned as last-slot over edge e for demand k if demand k uses edge e. Inequalities (10) and (11) impose that the length of the working and protection paths to be no more than $\bar{\ell}_k$ for each demand k. Inequalities (12) express the non-overlapping constraints. They ensure that every slot s over edge e can be assigned to at most one demand $k \in K$. Inequalities (13)-(15) are the trivial inequalities, and constraints (16) and (17) are the integrality constraints. We will denote by $\mathcal{P}(G, K, \mathbb{S})$ the polytope, convex hull of the solutions of (2)-(17).

4 Valid inequalities

In what follows, we describe various classes of valid inequalities for the polytope $\mathcal{P}(G, K, \mathbb{S})$. These will be used to strengthen the linear relaxation of the problem. Some of these inequalities are

obtained by using which will be called a conflict graph. Further inequalities are induced by covers related to some capacity constraints and also to the delay constraint.

4.1 Cutset-based inequalities

We first introduce some valid inequalities related to the 2-connectivity of the solutions.

Proposition 1. *Let e be an edge of E. Then the inequality*

$$\sum_{k\in K} w_k(x_e^k + b_e^k) \leq \bar{s} \tag{18}$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$.

Inequalities (18) express the *capacity constraint* on the edges. They capture the fact that at most $\bar{s}$ slots can be used over each edge e.

By using non overlapping inequalities (12) and capacity constraints (18), we introduce the following inequalities.

Consider a subset of nodes W in V. We denote by $K(W)$ the set of all the demands of K whose origin and destination nodes are separated by the cut $\delta(W)$, i.e., $|W \cap \{o_k, d_k\}| = 1$ for each $k \in K(W)$ with $K(W) \neq \emptyset$.

Proposition 2. *Let W be a subset of nodes in V. Then the inequality*

$$\sum_{e\in\delta(W)} \sum_{k\in K} \sum_{s=1}^{\bar{s}} \sum_{s'=s}^{\min(s+w_k-1,\bar{s})} t_{e,s'}^k \geq 2 \sum_{k\in K(W)} w_k \tag{19}$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$.

Proof. Consider a subset of nodes W in V. By summing inequalities (18) for each edge $e \in \delta(W)$, we obtain

$$\sum_{e\in\delta(W)} \sum_{k\in K} \sum_{s=1}^{\bar{s}} \sum_{s'=s}^{\min(s+w_k-1,\bar{s})} t_{e,s'}^k \geq \sum_{e\in\delta(W)} \sum_{k\in K} w_k(x_e^k + b_e^k).$$

This inequality is still valid when considering only the set of demands $K(W)$. That is to say when we replace the sum $\sum_{e\in\delta(W)} \sum_{k\in K}$ by $\sum_{e\in\delta(W)} \sum_{k\in K(W)}$ as follows

$$\sum_{e\in\delta(W)} \sum_{k\in K(W)} \sum_{s=1}^{\bar{s}} \sum_{s'=s}^{\min(s+w_k-1,\bar{s})} t_{e,s'}^k \geq \sum_{e\in\delta(W)} \sum_{k\in K(W)} w_k(x_e^k + b_e^k).$$

From inequalities (3) and (4), each demand $k \in K(W)$ uses two different edges in each cut $\delta(W)$ separating its origin and destination nodes. This implies that

$$\sum_{e \in \delta(W)} x_e^k + b_e^k \geq 2, \text{ for all } k \in K(W),$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$, and the statement follows.

This ensures that the total number of slots used in the cut must be greater than the total number of slots required by the set of demands in $K(W)$. Inequalities (19) will be called *cut inequalities*.

In the next section, we introduce some valid cover-based inequalities.

4.2 Cover inequalities

Using inequalities (18), we introduce some cover inequalities related to the capacity constraint. A subset of demands $\tilde{K} \subseteq K$ is said to be a *cover* for spectrum $\mathbb{S}$ if $\sum_{k \in \tilde{K}} w_k > \bar{s}$. A cover $\tilde{K}$ is said to be *minimal* if $\sum_{k' \in \tilde{K} \setminus \{k\}} w_{k'} \leq \bar{s}$ for each $k \in \tilde{K}$.

Proposition 3. *Consider an edge $e \in E$. Let $\tilde{K}$ be a minimal cover for spectrum $\mathbb{S}$. Then the inequality*

$$\sum_{k \in \tilde{K}} x_e^k + b_e^k \leq |\tilde{K}| - 1 \tag{20}$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$.

This shows that there exists at least one demand k from $\tilde{K}$ that cannot share edge e with the rest of demands in $\tilde{K}$ due to the capacity constraint (18). Inequalities (20) will be called *cover inequalities*.

Based on inequalities (20) and the contiguity constraint, we introduce a further class of cover inequalities related to the interval of slots over the edges. An interval of slots $I = [s_i, s_j]$ represents an ordered set of contiguous slots situated between the two slots s_i and s_j with $j \geq i + 1$ and $s_j \leq \bar{s}$ (e.g., interval $I = [1, 6]$ contains all slots situated between the slots $s_i = 1$ and $s_j = 6$).

Consider an interval of slots $I = [s_i, s_j]$ in $\mathbb{S}$. A subset of demands $\tilde{K}$ of K is called a *cover* of interval I if $\sum_{k \in \tilde{K}} w_k > |I|$ and $w_k < |I|$ for each $k \in \tilde{K}$. A cover for an interval of slots I is said to be *minimal* for I if $\sum_{k' \in \tilde{K} \setminus \{k\}} w_{k'} \leq |I|$ for each $k \in \tilde{K}$. Based on this, we introduce the following inequalities.

Proposition 4. *Consider an edge $e \in E$. Let $I = [s_i, s_j]$ be an interval in $\mathbb{S}$, and $\tilde{K}$ a minimal cover for the interval I. Then the inequality*

$$\sum_{k \in \tilde{K}} \sum_{s=s_i+w_k-1}^{s_j} t_{e,s}^k \leq |\tilde{K}| - 1 \tag{21}$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$.

Inequalities (21) show that interval I can cover at most $|\tilde{K}| - 1$ demands over edge e given that $\tilde{K}$ is a minimal cover for interval I over edge e. Otherwise, the non-overlapping constraint is violated given that there exists at least one slot $s \in I$ which is shared between at least two demands.

Note that inequalities (20) and (21) are equivalent when $I = \{1, ..., \bar{s}\}$. Inequalities (21) will be also called *cover inequalities*.

On the other hand, we propose further cover-based inequalities to strengthen the delay constraints. Consider a demand $k \in K$. A subset of edges C in E is called a *cover* for demand k, related to the delay constraint, if $\sum_{e \in C} \ell_e > \bar{l}_k$. Furthermore, it is called *minimal* for demand k if for each $e \in C$ we have $\sum_{e' \in C \setminus \{e\}} \ell_{e'} \leq \bar{l}_k$.

Proposition 5. *Let C be a minimal cover related to the delay constraint for a demand $k \in K$. Then, the following inequalities*

$$\sum_{e \in C} x_e^k \leq |C| - 1, \tag{22}$$

$$\sum_{e \in C} b_e^k \leq |C| - 1, \tag{23}$$

are valid for $\mathcal{P}(G, K, \mathbb{S})$.

In what follows, we present further classes of valid inequalities induced by cliques and odd-cycles of some conflict graphs.

4.3 Clique inequalities

Consider an edge $e \in E$. Let $\tilde{H}_S^e$ be the graph defined as follows. For each slot $s \in \{w_k, ..., \bar{s}\}$ and demand $k \in K$, consider a node n_s^k in $\tilde{H}_S^e$. Two nodes n_s^k and $n_{s'}^{k'}$ are linked by an edge in $\tilde{H}_S^e$ if either $k = k'$ and (4), or $\{s - w_k + 1, ..., s\} \cap \{s' - w_{k'} + 1, ..., s'\} \neq \emptyset$ (if $k \neq k'$). Graph $\tilde{H}_S^e$ will be called *conflict graph*.

Proposition 6. *Consider an edge $e \in E$. Let C be a clique in the conflict graph $\tilde{H}_S^e$ with $|C| \geq 3$. Then the inequality*

$$\sum_{n_s^k \in C} t_{e,s}^k \leq 1 \tag{24}$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$.

Now we present some clique-based inequalities related to the delay constraints. For this, we need to consider the following conflict graph.
Given a demand $k \in K$. We introduce the conflict graph $\tilde{H}_E^k$ as follows. For each edge $e \in E$, consider a node v_e in $\tilde{H}_E^k$. Two nodes v_e and $v_{e'}$ are linked by an edge in $\tilde{H}_E^k$ if and only if the two edges e and e' cannot be used together in a feasible path for routing demand k. These two edges (e, e') are said to be *non compatible edges* for demand k.

Proposition 7. *Consider a demand $k \in K$. Let C be a clique in $\tilde{H}_E^k$. Then, the following inequalities*

$$\sum_{v_e \in C} x_e^k \leq 1, \tag{25}$$

$$\sum_{v_e \in C} b_e^k \leq 1, \tag{26}$$

are valid for $\mathcal{P}(G, K, \mathbb{S})$.

Notice that the set of non compatible edges can be determined in polynomial time for each demand k.

4.4 Odd-cycle inequalities

In this section we describe some valid inequalities induced by odd cycles obtained from conflict graphs.

Proposition 8. *Consider an edge $e \in E$. Let H be an odd-cycle in the conflict graph $\tilde{H}_S^e$ with $|H| \geq 5$. Then the inequality*

$$\sum_{v_s^k \in H} t_{e,s}^k \leq \frac{|H| - 1}{2} \tag{27}$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$.

Proof. For each pair of nodes $(n_{k,s}, n_{k',s'})$ linked in H by an edge, we have $t^k_{e,s} + t^{k'}_{e,s'} \leq 1$. By summing these inequalities for all edges of H, it follows that

$$\sum_{n^k_s \in H} 2t^k_{e,s} \leq |H|.$$

Since $|H|$ is odd, by dividing by 2 and rounding down the right hand side, it follows that

$$\sum_{n^k_s \in H} t^k_{e,s} \leq \left\lfloor \frac{|H|}{2} \right\rfloor = \frac{|H|-1}{2}.$$

Inequality (27) can be strengthened by combining inequalities (24) and (27).

Proposition 9. *Consider an edge $e \in E$. Let H be an odd-cycle, and C a clique in the conflict graph $\tilde{H}^e_S$ with $|H| \geq 5$, $|C| \geq 3$, $H \cap C = \emptyset$, and the nodes $v^k_s, v^{k'}_{s'}$ are linked in $\tilde{H}^e_S$ for all $v^k_s \in H$ and $v^{k'}_{s'} \in C$. Then the inequality*

$$\sum_{v^k_s \in H} t^k_{e,s} + \frac{|H|-1}{2} \sum_{v^{k'}_{s'} \in C} t^{k'}_{e,s'} \leq \frac{|H|-1}{2} \tag{28}$$

is valid for $\mathcal{P}(G, K, \mathbb{S})$.

Proof. If $t^{k'}_{e,s'} = 1$ for some $v^{k'}_{s'} \in C$, then $\sum_{v^k_s \in H} t^k_{e,s} = 0$. Otherwise, the statement follows from inequality (27).

In a similar way, we propose a set of odd-cycle inequalities that are related to the delay constraints. These inequalities, whose validity follows as for inequalities (28), can be described as follows.

Proposition 10. *Consider a demand $k \in K$. Let H be an odd-cycle in the conflict graph $\tilde{H}^k_E$ with $|H| \geq 5$. Then the following inequalities*

$$\sum_{v_e \in H} x^k_e \leq \frac{|H|-1}{2}, \tag{29}$$

$$\sum_{v_e \in H} b^k_e \leq \frac{|H|-1}{2} \tag{30}$$

are valid for $\mathcal{P}(G, K, \mathbb{S})$.

Inequality (29) and (30) can also be strengthened without modifying its right-hand side by combining it with inequalities (25) and (26) respectively.

Proposition 11. *Consider a demand $k \in K$. Let H be an odd-cycle, and C a clique in the conflict graph $\tilde{H}_E^k$ with $|H| \geq 5$, $|C| \geq 3$, $H \cap C = \emptyset$, and the nodes $v_e, v_{e'}$ are linked in $\tilde{H}_E^k$ for all $v_e \in H$ and $v_{e'} \in C$. Then, the inequalities*

$$\sum_{v_e \in H} x_e^k + \frac{|H|-1}{2} \sum_{v_{e'} \in C} x_{e'}^k \leq \frac{|H|-1}{2}, \tag{31}$$

$$\sum_{v_e \in H} b_e^k + \frac{|H|-1}{2} \sum_{v_{e'} \in C} b_{e'}^k \leq \frac{|H|-1}{2}, \tag{32}$$

are valid for $\mathcal{P}(G, K, \mathbb{S})$.

5 Column Generation algorithm

In order to handle the large number of variables in the path formulation, we discuss in this section a column generation algorithm for solving its linear relaxation. The main ingredient of a column generation algorithm is the *pricing algorithm* which is used for solving which is called the *pricing problem*. This consists in generating new columns (variables) to add to the current linear relaxation of the problem. The pricing problem here involves finding a working path P and a protection path P' for each demand k and slot s with a negative reduced cost, using the optimal solution of the dual variables associated with the optimal solution of the linear relaxation of the problem. To accomplish this, the following dual variables are considered,

- α dual variables of equations (2) such that $\alpha^k \in \mathbb{R}$ for all $k \in K$,
- β dual variables of equations (3) such that $\beta^k \in \mathbb{R}$ for all $k \in K$,
- ζ dual variables of equations (4) such that $\zeta^k \in \mathbb{R}$ for all $k \in K$,
- η dual variables of inequalities (5) such that $\eta_e^k \leq 0$ for all $k \in K$ and $e \in E$,
- θ dual variables of inequalities (6) such that $\theta_e^k \leq 0$ for all $k \in K$ and $e \in E$,
- λ dual variables of inequalities (7) such that $\lambda_{e,s}^k \leq 0$ for all $k \in K$ and $e \in E$ and $s \in \mathbb{S}$,
- φ dual variables of inequalities (8) such that $\varphi_e^k \leq 0$ for all $k \in K$ and $e \in E$,
- ρ dual variables of equations (9) such that $\rho_e^k \in \mathbb{R}$ for all $k \in K$ and $e \in E$,
- σ dual variables of inequalities (10) such that $\sigma^k \leq 0$ for all $k \in K$,
- τ dual variables of inequalities (11) such that $\tau^k \leq 0$ for all $k \in K$,
- μ dual variables of inequalities (12) such that $\mu_s^e \leq 0$ for all $e \in E$ and $s \in \mathbb{S}$,

The dual problem of the linear relaxation of the path formulation is given by

$$\max - \sum_{k \in K} \beta^k - \sum_{k \in K} \zeta^k + \sum_{k \in K} \sigma^k \bar{l}_k + \sum_{k \in K} \tau^k \bar{l}_k, \tag{33}$$

subject to

$$\beta^k + \sum_{e \in E(p)} (-\lambda^k_{e,s} - \eta^k_e - \lambda^k_{e,s}) \geq 0, \forall k \in K, \forall p \in P^k, \forall s \in \{w_k, ..., \bar{s}\}, \tag{34}$$

$$\zeta^k + \sum_{e \in E(p)} (-\lambda^k_{e,s} - \theta^k_e - \lambda^k_{e,s}) \geq 0, \forall k \in K, \forall p \in P^k, \forall s \in \{w_k, ..., \bar{s}\}, \tag{35}$$

$$\alpha^k - \sum_{e \in E(p)} \lambda^k_{e,s} \geq 0, \forall k \in K, \forall p \in P^k, \forall s \in \{1, ..., w_k - 1\}, \tag{36}$$

$$\lambda^k_{e,s} - \sum_{s'=\max(0,s-w_k+1)}^{s} \mu^e_s - \rho^k_e \geq 0, \forall k \in K, \forall e \in E, \forall s \in \mathbb{S}, \tag{37}$$

$$\eta^k_e - \sigma^k - \varphi^k_e + \rho^k_e \geq 0, \forall k \in K, \forall e \in E, \tag{38}$$

$$\theta^k_e - \tau^k - \varphi^k_e + \rho^k_e \geq 0, \forall k \in K, \forall e \in E, \tag{39}$$

$$\varphi^k_e \leq 0, \forall k \in K, \forall e \in E, \tag{40}$$

$$\sigma^k, \tau^k \leq 0, \forall k \in K, \tag{41}$$

$$\lambda^k_{e,s} \leq 0, \forall k \in K, \forall e \in E, \forall s \in \mathbb{S}, \tag{42}$$

$$\mu^e_s \leq 0, \forall e \in E, \forall s \in \mathbb{S}, \tag{43}$$

$$\eta^k_e, \theta^k_e \leq 0, \forall k \in K, \forall e \in E. \tag{44}$$

The reduced-cost $c^k_s(P)$ related to the generation of a new working path $P \in P^k$ for each demand $k \in K$ and slot $s \in \{w_k, ..., \bar{s}\}$, is given by

$$c^k_s(P) = \beta^k + \sum_{e \in E(P)} (-\lambda^k_{e,s} - \eta^k_e - \lambda^k_{e,s}) \geq 0. \tag{45}$$

Therefore, the pricing problem for each demand $k \in K$ and slot $s \in \{w_k, ..., \bar{s}\}$, consists in finding a working path P^* of P^k such that

$$c^k_s(P^*) = \beta^k + \min_{\tilde{p} \in P^k} [\sum_{e \in E(\tilde{p})} -\lambda^k_{e,s} - \eta^k_e - \lambda^k_{e,s}] < 0.$$

In consequence, taking into account that $-\lambda^k_{e,s} - \eta^k_e - \lambda^k_{e,s} \geq 0$ for each $k \in K$ and $e \in E$, the pricing problem reduces to the so-called *Resource Constrained Shortest Path* (RCSP) Problem. The RCSP is NP-hard [13]. For this, we propose a pseudo-polynomial time algorithm using dynamic programming [14] to compute the minimum-cost path for each demand k while satisfying the delay constraint.

In a similar way, the reduced-cost $c'^k_s(P')$ related to the generation of a new protection path

$P' \in P^k$ for a demand $k \in K$ and slot $s \in \{w_k, ..., \bar{s}\}$, is given by

$$c'^k_s(P') = \zeta^k + \sum_{e \in E(P')} (-\lambda^k_{e,s} - \theta^k_e - \lambda^k_{e,s}). \tag{46}$$

The pricing problems for these paths is also equivalent to the RCSP.

6 Branch-and-Cut-and-Price algorithm

In this section, we describe a Branch-and-Cut-and-Price algorithm for the problem. This aims at using the results obtained in the previous sections and show their utility in solving the problem. The Branch-and-Cut-and-Price algorithm combines a column generation algorithm, a cutting-plane algorithm, and a Branch-and-Bound algorithm. The main objective of this algorithm is to solve a series of linear programs using the column generation algorithm.

The column generation algorithm begins with a restricted linear program called *restricted master problem* (RMP) that contains a small subset of variables, providing a feasible basis for the master problem. During each iteration of the column generation, the algorithm checks for the existence of new variable $y^k_{p,s}$ or $f^k_{p,s}$ not present in the current RMP, with a negative reduced cost, by solving the pricing problem. If such a variable is found, it is added to the current RMP, which is then solved again. This process continues until no new path with negative reduced cost is found. Then, the cutting-plane phase generates additional valid inequalities that are violated by the current solution. This is achieved by solving the so-called *separation problem* associated with each class of valid inequalities we consider.

It is important to note that the cutting-plane algorithm provides an optimal solution only for the linear relaxation of the problem, which may not be feasible for the original problem if it does not satisfy the integrality constraints. In such cases, the Branch-and-Cut-and-Price algorithm proceeds to the branching step, where the problem is divided into subproblems by branching on selected integer variables that have fractional values. The column generation and cutting-plane algorithms are then applied to each sub-problem. This process continues until an optimal solution is obtained for the problem.

6.1 Initial columns

As mentionned before, we first provide a subset of variables defining a feasible basis for the master problem. For this, we first compute initial columns that define a feasible solution for the master

problem using a brute force procedure. This procedure involves creating a search tree that encompasses several paths between (o_k, d_k) of each demand $k \in K$. Using this, we generate an initial set of precomputed paths $B_k \subseteq P_k$ for each demand k such that there exist at least two paths $P, P' \in B_k$ with $E(P) \cap E(P') = \emptyset$. Then, we introduce a light path variable $y^k_{p,s}$ and $f^k_{p,s}$ for each demand k, path $P \in B_k$, and slot $s \in \{1, ..., \bar{s}\}$ with $\bar{s} = 320$. Additionally, we include all variables x^k_e, b^k_e for each demand $k \in K$ and edge $e \in E$, and variables $t^k_{e,s}$ for each demand $k \in K$, edge $e \in E$ and slot $s \in \{1, ..., \bar{s}\}$. By incorporating these variables, we establish a feasible basis for the RMP.

As mentionned before, it is necessary to devise separation routines for the different classes of valid inequalities. This will be discussed in the following section.

6.2 Separation routines

To enhance the efficiency of the Branch-and-Cut-and-Price algorithm, we have introduced various classes of valid inequalities to obtain tighter LP bounds. In each iteration and each node of the Branch-and-Cut-and-Price tree, it is possible to identify one or more inequalities violated by the current fractional solution.
In what follows, we consider a fractional solution $(\bar{x}, \bar{b}, \bar{y}, \bar{f}, \bar{t})$.

6.2.1 Separation of cut inequalities

In this section, we discuss the separation problem associated with the cut inequalities (19). Identifying cut-set inequalities violated by $(\bar{x}, \bar{b}, \bar{y}, \bar{f}, \bar{t})$ is known to be NP-hard, as already proven by Bienstock et al. [2]. We have developed a heuristic approach for separating these inequalities. For this, we use the Goldberg-Tarjan max-flow algorithm [16] to identify violated cut-set inequalities (19) by determining a minimum (o_k, d_k)-cut for each demand $k \in K$. This can be done exactly and very effectively in $\mathcal{O}(|V|^2 * \sqrt{|E|})$ time for each demand k using an efficient implementation of the minimum cut algorithm based on the preflow push-relabel algorithm of Goldberg and Tarjan [16]. For this, we use a C++ library provided by the LEMON GRAPH library (Lemon) [26]. We first assign a capacity of $\bar{x}^k_e + \bar{b}^k_e$ to each edge $e \in E$. Then, we determine the minimum (o_k, d_k)-cut in the graph G, denoted by $\delta(W^*)$ where $W^* \subset V$. This allows us to identify the subset of demands $\tilde{K}$ that pass through this cut $\delta(W^*)$. Finally, we add the cut-set inequality (19) induced by the

cut $\delta^(W^*)$ and demands of $\tilde{K}$

$$\sum_{e\in\delta(W^*)}\sum_{k\in\tilde{K}}\sum_{s=1}^{\bar{s}}\sum_{s'=s}^{\min(s+w_k-1,\bar{s})} t^k_{e,s'} \geq 2\sum_{k\in\tilde{K}} w_k$$

to the current linear program if it is violated. This heuristic is in $\mathcal{O}(|V|^2 * \sqrt{|E|} * |K|)$ time in the worst case.

6.2.2 Separation of cover inequalities

The separation problem for the cover inequalities was first considered by Crowder et al. [9] for the knapsack problem. It is well-known that the problem is NP-hard [37][32]. To address this, Nemhauser and Sigismondi [33] proposed a greedy algorithm that provides an approximate solution for this problem.

In this section, we discuss the separation problem of inequalities (21). Given an edge $e \in E$ and an interval $I = [s_i, s_j]$ of contiguous slots in $\mathbb{S}$, the objective here is to find a minimal cover $\tilde{K}^*$ for the interval I over the edge e, satisfying

$$\sum_{k\in\tilde{K}^*}\sum_{s'=s_i+w_k-1}^{s_j} \bar{t}^k_{e,s'} > |\tilde{K}^*| - 1.$$

For this, we propose the following heuristic [33]. First, we choose a demand $k \in K$ with the highest number of slots w_k such that $\sum_{s'=s_i+w_k-1}^{s_j} \bar{t}^k_{e,s'} > 0$. Then, we iteratively add a demand $k' \in K\setminus\tilde{K}^*$ to $\tilde{K}^*$ if $\sum_{s'=s_i+w_{k'}-1}^{s_j} \bar{t}^{k'}_{e,s'} > 0$ until $\tilde{K}^*$ becomes a cover for the interval I over edge e, that is $\sum_{k\in\tilde{K}^*} w_k > |I|$. A minimal cover is then obtained from $\tilde{K}^*$ by removing each demand $k \in \tilde{K}^*$ with $\sum_{k'\in\tilde{K}^*\setminus\{k\}} w_{k'} > |I|$. If the corresponding inequality (21) is violated, then it should be added to the current LP.

This procedure can also be used for separating inequalities (20). Consider an edge $e \in E$. The procedure focuses on finding a minimal cover $\tilde{K}^*$ for the edge e such that $\sum_{k\in\tilde{K}^*} w_k(\bar{x}^k_e + \bar{b}^k_e) > |\tilde{K}^*| - 1$. Initially, a demand $k \in K$ with the largest number of requested slots w_k and $\bar{x}^k_e + \bar{b}^k_e > 0$ is selected, and $\tilde{K}^*$ is set to $\{k\}$. Then, each demand $k' \in K \setminus \tilde{K}^*$ is added to $\tilde{K}^*$ if $\bar{x}^{k'}_e + \bar{b}^{k'}_e > 0$ until $\tilde{K}^*$ becomes a cover for edge e. Once this is achieved, a minimal cover is derived from $\tilde{K}^*$ by removing each demand $k \in \tilde{K}^*$ if $\sum_{k'\in\tilde{K}^*\setminus\{k\}} w_{k'} > \bar{s}$. If the inequality (20) induced by $\tilde{K}^*$ is violated, then it is added to the current LP.

In a similar way, we use the same procedure of Nemhauser and Sigismondi [33] for the separation of inequalities (22) and (23).

6.2.3 Separation of clique inequalities

Let us turn now to the separation of the clique inequalities. This is well known to be NP-Hard [35][36]. For our purpose, we adapt the greedy algorithm of Nemhauser and Sigismondi [33] for the separation of cover inequalities. Our algorithm for inequalities (24) involves identifying for each edge e a maximum clique C^* such that

$$\sum_{v_{k,s} \in C^*} \bar{t}^k_{e,s} > 1.$$

To this end, we adapt a greedy algorithm of [33] to find a maximum clique C^* in the conflict graph $\tilde{H}^e_S$. This works as follows. We first assign weight $\bar{t}^k_{e,s}$ to each node $v_{k,s}$ in $\tilde{H}^e_S$. Then, we select a node $v_{k,s}$ with the highest weight and set $C^* = \{v_{k,s}\}$. Then, iteratively add to C^* each node $v_{k',s'}$ with $\bar{t}^{k'}_{e,s'} > 0$, if it is adjacent to all the nodes $v_{k,s}$ already in C^*. If the inequality induced by C^* is violated, then it is added to the current LP.

Now, we describe the separation problem for inequalities (25). This reduces in finding a maximum clique C^* such that

$$\sum_{v_e \in C^*} \bar{x}^k_e > 1.$$

To address this, we can apply the algorithm proposed in [33] to identify for each demand k a maximum clique C^* in $\tilde{H}^k_E$. This can be presented as follows. First, we assign the weight $\bar{x}^k_e$ to each node v_e in the conflict graph $\tilde{H}^k_E$. We then select a node v_e in the conflict graph $\tilde{H}^k_E$ with the highest weight, and set $C^* = \{v_e\}$. After that, we add each node $v_{e'}$ to the current C^* if $\bar{x}^k_{e'} > 0$ and it is adjacent to all the nodes already assigned to the current clique C^*. Finally, if the inequality (25) induced by clique C^* is violated, it is added to the current LP. We apply the same procedure for each demand $k \in K$. This approach has also been used for the separation of inequalities (26).

6.2.4 Separation of odd-cycle inequalities

Now, we turn our attention to the separation of inequalities (27). Recall that the related separation problem reduces to identifying an odd-cycle H^* in $\tilde{H}^e_S$ such that

$$\sum_{v_{k,s} \in H^*} \bar{t}^k_{e,s} > \frac{|H^*| - 1}{2}.$$

This problem can be solved exactly in polynomial time [17]. The resolution consists in finding a minimum weighted odd-cycle in an auxiliary graph. Let $\tilde{H}'^{e}_{S}$ be the graph obtained from $\tilde{H}'^{e}_{S}$ as follows. Duplicate each node $v_{k,s}$ in $\tilde{H}^{e}_{S}$. Two nodes form an edge in $\tilde{H}'^{e}_{S}$ if the corresponding nodes are adjacent in $\tilde{H}^{e}_{S}$. Each link $(\tilde{v}_{k,s}, \tilde{v}_{k',s'})$ in $\tilde{H}'^{e}_{S}$ is assigned a non-negative weight equal to $\frac{1-\bar{t}^{k}_{e,s}-\bar{t}^{k'}_{e,s'}}{2}$. The shortest path $P_{v_{k,s},v'_{k,s}}$ between $v_{k,s}$ and its copy in $\tilde{H}'^{e}_{S}$ is computed for each node $v_{k,s}$. The sum of weights over edges belonging to this path is then checked whether it is smaller than $\frac{1}{2}$, that is

$$\sum_{(v_{a,s'},v_{b,s"})\in E(P_{v_{k,s},v'_{k,s}})} \frac{1-\bar{t}^{a}_{e,s'}-\bar{t}^{b}_{e,s"}}{2} < \frac{1}{2}.$$

If this is the case, then the odd cycle H^* will consist of all the initial nodes corresponding to the ones in this path. And consequently, the inequality (27) induced by H^* is added to the current LP. This has been adapted for the separation of inequalities (29) and (30).

7 Reinforcement Learning

In what follows, we present a Reinforcement Learning (RL) approach that will be integrated with our Branch-and-Cut-and-Price algorithm. This combination aims to enhance the efficiency and effectiveness of the overall Branch-and-Cut-and-Price algorithm for solving the survivable CRSA problem. In this context, the RL approach can be employed to train an RL agent to make best decisions regarding column generation and cutting-planes during each iteration of the algorithm. The RL agent learns and determines which columns and cuts to add to the so-called restricted master problem.

7.1 Learning to parameterize the pricing algorithm

In this section, a supervised learning approach is introduced to study the best setting for solving the pricing problem using the dynamic programming algorithm. The objective of this approach is to reduce the size of the input graph by selecting only the most promising edges for each demand k. Such an edge is defined as an edge that has been used at least once to generate new columns for demand k during any iteration of the column generation algorithm. For this, and for each demand k, a set of promising edges E_k is defined, initially empty. Note that the promising edge selection procedure can be executed either before starting (i.e., given by an expert or known in advance) the problem-solving or after the second iteration of the column generation algorithm. At each iteration, the set of promising edges is updated for each demand k by only adding those edges that belong

to the new working or protection path associated with the additional column (having a negative reduced cost) of demand k. It is important to note that when dealing with large-sized instances, the percentage of edges that are used at least once during the resolution of the linear relaxation by column generation is very small. Therefore, we believe that this approach has the potential to improve the resolution of the pricing problem and speed up the dynamic programming algorithm.

7.2 Learning for solving the pricing problem

The RL approach also enhances the column generation algorithm by providing a framework for learning optimal strategies in solving the pricing problem. The objective is to improve the efficiency and effectiveness of the column generation algorithm. In our study, the RL agent is guided by a score function that evaluates the quality of each solving method: Dijkstra or dynamic programming algorithm. For this, we assign a score, denoted by $score\_algo$, to each method. This score is calculated for each method by dividing the total number of generated new columns by each method by the total number of generated columns during the algorithm. Notice that the score is updated at each iteration of the algorithm.

During each iteration of the column generation algorithm, we generate a random real value denoted by a between 0 and 1. We then select the method $algo^*$ using a $Q-$ learning mechanism which consists in finding the optimal action-selection based on the different scores. Next, we only solve the pricing problem using $algo^*$. If $algo^* = Dijkstra$ is used and no new column is generated, we solve again the pricing problem using the dynamic programming algorithm given that Dijkstra is not always able to solve the pricing problem to optimality. This procedure is applied at each iteration of the column generation algorithm. Notice that at the first iteration, we prioritize the use of Dijkstra to solve the pricing problem by giving a higher score.

7.3 Full learning for the pricing algorithm

Based on what we have mentionned in section 7.1 and 7.2, at each iteration of the column generation, we follow the following steps:

1. for each demand $k \in K$ and slot $s \in \{w_k, ..., \bar{s}\}$, we use $algo^*$ to solve the pricing problem while considering only a subgraph G' as mentioned in Section 7.1. If no new column is generated, we go to step 2. Otherwise, we add all the generated variables and resolve the new RMP.
2. for each demand $k \in K$ and slot $s \in \{w_k, ..., \bar{s}\}$, we use again $algo^*$ to solve the pricing problem while considering the original graph G. If no new column is generated, we pass to step 3. Otherwise, we add all the generated variables and solve the new RMP.

3. We distinguish two cases:
   - if $algo^* = Dijkstra$: for each demand $k \in K$ and slot $s \in \{w_k, ..., \bar{s}\}$, we use dynamic programming algorithm to solve the pricing and only consider a subgraph G' as mentioned in Section 7.1. If no new column is generated, we go to step 4. Otherwise, we resolve the new RMP while adding the new generated columns.
   - if $algo^* \neq Dijkstra$: the current linear relaxation solution is optimal.
4. for each demand $k \in K$ and slot $s \in \{w_k, ..., \bar{s}\}$, we solve the pricing problem using a dynamic programming algorithm considering the original graph G. If no new column is generated, the current linear relaxation solution is optimal. Otherwise, we add all the generated variables and solve the new RMP.

Notice that we only use a dynamic programming algorithm for solving the pricing problem when the learning mechanism is desactivated.

7.4 Learning for the cutting plane phase

We also use a RL approach to enhance the cutting plane algorithm we have described before. The aim is to explore different separation strategies for the various classes of inequalities we have introduced.

Similary to the column generation algorithm, a RL approach provides a framework for selecting classes of inequalities which may lead to better performance. In this context, we assign a score, denoted by *score_ineq*, to each class of valid inequalities. This score is calculated by dividing the total number of generated inequalities from this class divided by the total number of generated inequalities during the algorithm. The score is updated at each iteration of the cutting plane algorithm. Our RL approach then selects a class of valid inequalities to be separated according to a $Q-$learning mechanism. The $Q-$learning mechanism relies on the scores that have been previously calculated for each class of valid inequalities. This process continues until either a new cut is found or no new cut is found for any of the inequalities.

By applying the RL this way, we can explore different classes of valid inequalities and learn which ones are most effective in improving the linear relaxation.

Notice that the scoring mechanism and random selection allow the RL agent to explore different strategies during the column generation and cutting-plane algorithms.

8 Computational study

The B&C&P algorithm was implemented using the C++ programming language on a Linux system. The implementation utilized the "Solving Constraint Integer Programs" (Scip 7.0) framework [46] and Cplex 12.9 [8] as the LP solver. The algorithm was tested on a high-performance server with a memory size of 64 GB and 8 activated threads for parallelism. The CPU time was limited to 5 hours (18000 seconds).

The computational results were obtained using two realistic graphs from SND-Lib [34] as shown in Figure 2. The total number of nodes ($|V|$) in the graphs ranged up to 25, and the total number of edges ($|E|$) ranged up to 44. The demands K were randomly generated with $|K|$ values ranging from 10 to 50, and the number of slots $\bar{s}$ ranged up to 320.

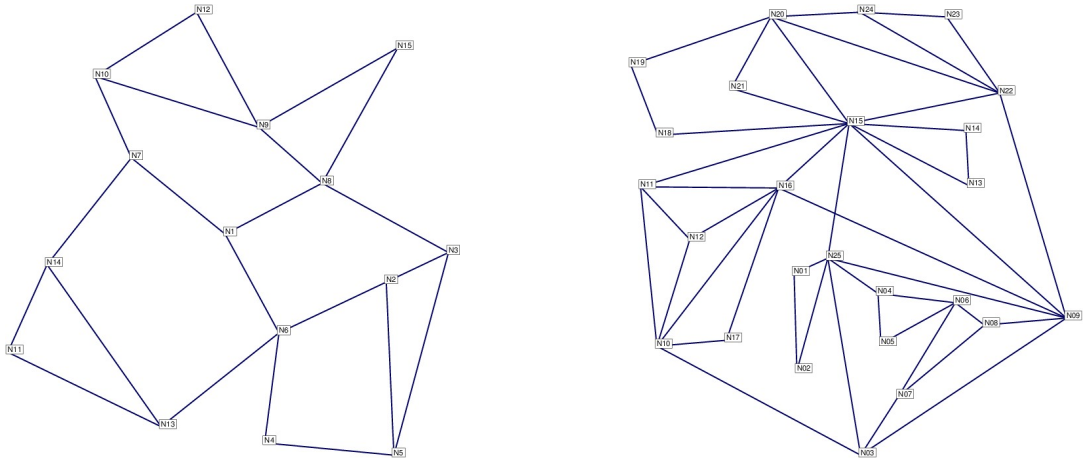


Fig. 2. SND-Lib graphs used in our computational study (Atlanta in left hand side and France in right hand side).

We consider 7 criteria in the computational study:

- the number of nodes in the B&C&P tree (Nd),
- the final lower bound (LB),
- the final upper bound (UB),
- the optimality gap (Gap) which represents the relative error between the lower bound gotten at the end of the resolution and best upper bound,
- the number of generated new columns (Cols),
- the number of violated inequalities added during the algorithm (Cuts),
- and the total CPU computational time (TT).

Note that $LB = 0$ means that the linear relaxation is not solved for the problem. Also, $Gap = INF$ means that the algorithm was not able to provide a feasible solution for the problem (when $UB = INF$) or the linear relaxation is not solved (when $LB = 0$). Also, $Gap = 0\%$ means that the problem was solved to optimality and $LB = UB$.
On the other hand, the order of separation in the B&C&P (without learning) is determined based on preliminary experiments and is as follows.

- Cover inequalities: (20), (21), (22),
- Clique inequalities: (24), (25) and (26),
- Odd-cycle inequalities: (27), (29) and (30),
- Cutset inequalities: (19).

In what follows, we report a comparative study between

- Branch-and-Price (denoted by B&P) Vs Learned Branch-and-Price (denoted by Learned_B&P) in Table 1,
- Branch-and-Cut-and-Price (denoted by B&C&P) Vs Learned Branch-and-Cut-and-Price (denoted by Learned_B&C&P) in Table 2,
- Branch-and-Price Vs Branch-and-Cut-and-Price in Table 3,
- Branch-and-Price Vs Learned Branch-and-Cut-and-Price in Table 4,
- Learned Branch-and-Price Vs Branch-and-Cut-and-Price in Table 5,
- Learned Branch-and-Price Vs Learned Branch-and-Cut-and-Price in Table 6.

Instances						B&P						Learned_B&P					
Name	$\|V\|$	$\|E\|$	$\|K\|$	$\|S\|$	**SumWk**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**
Atlanta_10_1	15	22	10	320	36	1	3608	3608	0	352,35	3056	1	3608	3608	0	691,57	4039
Atlanta_10_2	15	22	10	320	40	1	3974	3974	0	550,83	4099	1	3974	3974	0	926,70	4552
Atlanta_10_3	15	22	10	320	32	1	4438	4438	0	534,28	3801	1	4438	4438	0	1075,57	4650
Atlanta_10_4	15	22	10	320	38	1	4446	4446	0	1542,21	6239	1	4446	4446	0	1169,49	5918
Atlanta_20_1	15	22	20	320	75	1	7972	7972	0	752,93	3839	1	7972	7972	0	1972,87	6092
Atlanta_20_2	15	22	20	320	70	1	7864	7864	0	1201,85	4865	1	7864	7864	0	2116,77	5921
Atlanta_20_3	15	22	20	320	82	1	8036	8036	0	1553,75	5614	1	8036	8036	0	2921,92	7230
Atlanta_20_4	15	22	20	320	70	2	8026	8026	0	999,34	4825	1	8026	8026	0	1236,22	4930
Atlanta_30_1	15	22	30	320	110	1	11188	11188	0	2263,65	6812	1	11188	11188	0	6017,35	10412
Atlanta_30_2	15	22	30	320	106	1	13602	13602	0	3359,67	7616	1	13602	13602	0	6337,66	10640
Atlanta_30_3	15	22	30	320	114	7	11748	11748	0	3440,12	7821	5	11748	11748	0	6030,61	10091
Atlanta_30_4	15	22	30	320	120	1	11248	11248	0	1096,13	6135	1	11248	11248	0	1502,82	7433
Atlanta_40_1	15	22	40	320	150	2	16456	16456	0	2912,13	8258	11	16456	16456	0	4751,02	11856
Atlanta_40_2	15	22	40	320	150	3	16324	16324	0	2870,33	9378	1	16324	16324	0	4451,20	13929
Atlanta_40_3	15	22	40	320	150	1	16650	16650	0	4069,14	10838	1	16650	16650	0	4357,06	12206
Atlanta_40_4	15	22	40	320	154	1	14990	14990	0	2898,91	9029	1	14990	14990	0	3707,70	12903
Atlanta_50_1	15	22	50	320	193	11	18922	19083	0,85	18000	14364	1	18922	18922	0	5138,30	17479
Atlanta_50_2	15	22	50	320	199	1	18692	18692	0	3222,87	9593	5	18692	18692	0	3888,97	11504
Atlanta_50_3	15	22	50	320	195	13	18934	18934	0	5769,38	11498	1	18934	18934	0	4327,25	15384
Atlanta_50_4	15	22	50	320	184	21	19608	19800	0,98	18000	11904	5	19608	19608	0	2829,69	11382
France_10_1	25	44	10	36	320	1	25918	25918	0	1316,38	3845	1	25918	25918	0	552,39	6336
France_10_2	25	44	10	40	320	3	33364	33364	0	1240,51	4520	11	33364	33364	0	890,57	6174
France_10_3	25	44	10	32	320	1	35694	35694	0	753,72	4672	1	35694	35694	0	683,37	6710
France_10_4	25	44	10	38	320	1	32105	32105	0	558,13	4835	1	32105	32105	0	647,67	6424
France_20_1	25	44	20	75	320	1	57991	57991	0	3011,14	6809	1	57991	57991	0	992,23	8791
France_20_2	25	44	20	70	320	1	65609	65609	0	1550,95	6381	1	65609	65609	0	905,46	8918
France_20_3	25	44	20	82	320	1	55432	55432	0	3794,91	5460	19	55432	55432	0	3931,66	9349
France_20_4	25	44	20	70	320	7	62668	62668	0	1614,27	6704	1	62668	62668	0	1696,09	10235
France_30_1	25	44	30	110	320	14	93439	93439	0	11236,74	8716	1	93439	93439	0	3882,17	14310
France_30_2	25	44	30	106	320	5	110248	110248	0	5622,95	11680	1	110248	110248	0	4150,55	14645
France_30_3	25	44	30	114	320	1	101356	101356	0	2425,3	7477	5	101356	101356	0	4284,27	13909
France_30_4	25	44	30	120	320	5	91264	91264	0	9505,11	12259	1	91264	91264	0	2237,30	14476
France_40_1	25	44	40	150	320	1	0	124321	INF	18000	17298	1	123210	123210	0	4306,91	18752
France_40_2	25	44	40	150	320	1	0	133835	INF	18000	13034	11	131954	132203	0,19	18000	25401
France_40_3	25	44	40	150	320	1	0	125606	INF	18000	15433	3	114926	114926	0	11646,78	16640
France_40_4	25	44	40	154	320	1	0	INF	INF	18000	11827	1	124739	125452	0,57	18000	20552
France_50_1	25	44	50	193	320	1	0	INF	INF	18000	9537	1	0	153554	INF	18000	21756
France_50_2	25	44	50	199	320	1	0	INF	INF	18000	11036	1	0	154452	INF	18000	21510
France_50_3	25	44	50	195	320	1	0	INF	INF	18000	9474	1	0	152886	INF	18000	21503
France_50_4	25	44	50	184	320	1	0	INF	INF	18000	12192	1	0	176742	INF	18000	23747

Table 1. Table of comparison for the B&P vs the Learned_B&P using SND-Lib graphs.

Table 1 shows the results obtained by the B&P algorithm without learning compared with the B&P algorithm with learning. As it appears, the number of instances solved to optimality increases when using the Learned_B&P. Also, the number of nodes in the branching tree is less for certain instances when using the Learned_B&P. Moreover, the Learned_B&P was able to solve some instances to optimality for which the linear relaxation has not been solved when using the B&P. We also observe that the CPU time over some instances decreases when using the Learned_B&P even if we generate more new columns when using the Learned_B&P. The remaining instances, they are solved with a small optimality gap compared to when using the B&P. Also, the Learned_B&P has shown to be able to provide some feasible solutions for some instances for which the linear relaxation has not been solved by the B&P, which has not been observed when using only the B&P.

In what follows, we compare in Table 2 between the B&C&P and the Learned_B&C&P.

Instances						B&C&P							Learned_B&C&P						
Name	$\|V\|$	$\|E\|$	$\|K\|$	$\|S\|$	**SumWk**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Cuts**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Cuts**
Atlanta_10_1	15	22	10	320	36	1	3608	3608	0	169,87	3056	0	1	3608	3608	0	230,28	4039	0
Atlanta_10_2	15	22	10	320	40	1	3974	3974	0	230,50	4099	0	1	3974	3974	0	245,38	4552	0
Atlanta_10_3	15	22	10	320	32	1	4438	4438	0	373,26	4627	0	1	4438	4438	0	300,63	4650	0
Atlanta_10_4	15	22	10	320	38	1	4446	4446	0	594,71	6239	0	1	4446	4446	0	324,28	5918	0
Atlanta_20_1	15	22	20	320	75	1	7972	7972	0	383,57	4587	0	1	7972	7972	0	453	6092	0
Atlanta_20_2	15	22	20	320	70	1	7864	7864	0	400,50	4865	0	1	7864	7864	0	718,96	5921	0
Atlanta_20_3	15	22	20	320	82	1	8036	8036	0	495,03	5188	0	1	8036	8036	0	986,09	7230	0
Atlanta_20_4	15	22	20	320	70	2	8026	8026	0	1125,76	4825	0	1	8026	8026	0	417,69	4930	0
Atlanta_30_1	15	22	30	320	110	1	11188	11188	0	2426,22	6812	0	1	11188	11188	0	2452,68	10412	56
Atlanta_30_2	15	22	30	320	106	1	13602	13602	0	3532,76	7616	30	1	13602	13602	0	1682,21	10049	0
Atlanta_30_3	15	22	30	320	114	5	11748	11748	0	3657	7795	55	3	11748	11748	0	2315,83	10720	4
Atlanta_30_4	15	22	30	320	120	1	11248	11248	0	2805,91	6450	136	1	11248	11248	0	1200,33	7433	0
Atlanta_40_1	15	22	40	320	150	1	16456	16456	0	751,19	4684	0	11	16456	16456	0	4189,60	11856	0
Atlanta_40_2	15	22	40	320	150	13	16324	16324	0	7167,04	9877	64	1	16324	16324	0	2538,70	13929	0
Atlanta_40_3	15	22	40	320	150	15	16650	16650	0	2267,31	4589	8	7	16650	16650	0	13611,14	16935	33
Atlanta_40_4	15	22	40	320	154	1	14990	14990	0	6056,70	10063	132	1	14990	14990	0	2251,98	12903	0
Atlanta_50_1	15	22	50	320	193	2	18922	19202	1,48	18000	11931	92	1	18922	18922	0	5815,76	17479	24
Atlanta_50_2	15	22	50	320	199	11	18692	18692	0	2585,75	9721	45	1	18692	18692	0	3479,28	11709	55
Atlanta_50_3	15	22	50	320	195	7	18934	19078	0,76	18000	12748	77	1	18934	18934	0	3612,93	15384	0
Atlanta_50_4	15	22	50	320	184	15	19608	19608	0	3565	9926	19	1	19608	19608	0	2185,87	11382	14
France_10_1	25	44	10	36	320	1	25918	25918	0	1956,63	3845	0	1	25918	25918	0	572,27	6336	0
France_10_2	25	44	10	40	320	1	33364	33364	0	1933,99	4520	7	11	33364	33364	0	932,72	6174	0
France_10_3	25	44	10	32	320	1	35694	35694	0	1188,21	4672	0	1	35694	35694	0	691,56	6710	0
France_10_4	25	44	10	38	320	1	32105	32105	0	927,38	4835	0	1	32105	32105	0	568,85	6424	0
France_20_1	25	44	20	75	320	1	57991	57991	0	4638,58	6809	0	1	57991	57991	0	1016,91	8791	0
France_20_2	25	44	20	70	320	1	65609	65609	0	2416,44	6381	0	1	65609	65609	0	958,47	8918	0
France_20_3	25	44	20	82	320	1	55432	55432	0	5371,12	5460	0	8	55432	55432	0	3829,65	9247	37
France_20_4	25	44	20	70	320	1	62668	62668	0	2457,92	6704	0	1	62668	62668	0	1520,54	10235	0
France_30_1	25	44	30	110	320	1	93439	93439	0	14503,67	8716	0	1	93439	93439	0	8376,95	13868	23
France_30_2	25	44	30	106	320	1	110248	110248	0	8195,66	11154	0	1	110248	110248	0	3102,85	14645	0
France_30_3	25	44	30	114	320	1	101356	101356	0	5561,17	8791	0	1	101356	101356	0	6101,83	11714	0
France_30_4	25	44	30	120	320	2	91264	91264	0	10680,01	13162	5	1	91264	91264	0	2294,50	14476	0
France_40_1	25	44	40	150	320	5	123210	123210	0	11787,3	13351	102	1	123210	123210	0	3593,02	18752	0
France_40_2	25	44	40	150	320	1	0	131954	INF	18000	15787	0	5	131954	133551	1,21	18000	23153	0
France_40_3	25	44	40	150	320	1	114926	114926	0	11093,36	10216	22	3	114926	114926	0	8204,39	18432	94
France_40_4	25	44	40	154	320	1	0	INF	INF	18000	13554	0	3	124739	125452	0,57	18000	20552	25
France_50_1	25	44	50	193	320	1	0	INF	INF	18000	12086	0	3	146265	148259	1,36	18000	21858	7
France_50_2	25	44	50	199	320	7	0	INF	INF	18000	10932	15	2	153581	154007	0,28	18000	23845	32
France_50_3	25	44	50	195	320	1	0	INF	INF	18000	11275	0	1	150759	151915	0,77	18000	24162	54
France_50_4	25	44	50	184	320	3	174775	177583	1,61	18000	13193	53	1	174775	176742	1,13	18000	26678	28

Table 2. Table of comparison for the B&C&P Vs the Learned_B&C&P using SND-Lib graphs.

The results in Table 2 indicate that using the Learned_B&C&P also leads to an increase in the number of instances solved to optimality. Additionally, for certain instances, the number of nodes in the branching tree is reduced when using the Learned_B&C&P. Furthermore, the Learned_B&C&P was capable to solve some instances to optimality for which even the linear relaxation could not be solved when using the B&C&P. We can also observe that the CPU time and the number of Cuts decreases for some instances when using the Learned_B&C&P. Other instances are solved with a small optimality gap compared to when using the B&C&P, and the Learned_B&C&P is able to provide feasible solutions for instances where the linear relaxation could not be solved when using the B&C&P.

The improvement observed in Tables 1 and 2 can be attributed to the utilization of the learning approach incorporated in the column generation algorithm.

Next, we compare in Table 3 between the B&P and the B&C&P to show the influence of the additional valid inequalities.

Instances						B&P						B&C&P						
Name	$\|V\|$	$\|E\|$	$\|K\|$	$\|S\|$	**SumWk**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Cuts**
Atlanta_10_1	15	22	10	320	36	1	3608	3608	0	352,35	3056	1	3608	3608	0	169,87	3056	0
Atlanta_10_2	15	22	10	320	40	1	3974	3974	0	550,83	4099	1	3974	3974	0	230,50	4099	0
Atlanta_10_3	15	22	10	320	32	1	4438	4438	0	534,28	3801	1	4438	4438	0	373,26	4627	0
Atlanta_10_4	15	22	10	320	38	1	4446	4446	0	1542,21	6239	1	4446	4446	0	594,71	6239	0
Atlanta_20_1	15	22	20	320	75	1	7972	7972	0	752,93	3839	1	7972	7972	0	383,57	4587	0
Atlanta_20_2	15	22	20	320	70	1	7864	7864	0	1201,85	4865	1	7864	7864	0	400,50	4865	0
Atlanta_20_3	15	22	20	320	82	1	8036	8036	0	1553,75	5614	1	8036	8036	0	495,03	5188	0
Atlanta_20_4	15	22	20	320	70	2	8026	8026	0	999,34	4825	2	8026	8026	0	1125,76	4825	0
Atlanta_30_1	15	22	30	320	110	1	11188	11188	0	2263,65	6812	1	11188	11188	0	2426,22	6812	0
Atlanta_30_2	15	22	30	320	106	1	13602	13602	0	3359,67	7616	1	13602	13602	0	3532,76	7616	30
Atlanta_30_3	15	22	30	320	114	7	11748	11748	0	3440,12	7821	5	11748	11748	0	3657	7795	55
Atlanta_30_4	15	22	30	320	120	1	11248	11248	0	1096,13	6135	1	11248	11248	0	2805,91	6450	136
Atlanta_40_1	15	22	40	320	150	2	16456	16456	0	2912,13	8258	1	16456	16456	0	751,19	4684	0
Atlanta_40_2	15	22	40	320	150	3	16324	16324	0	2870,33	9378	13	16324	16324	0	7167,04	9877	64
Atlanta_40_3	15	22	40	320	150	1	16650	16650	0	4069,14	10838	15	16650	16650	0	2267,31	4589	8
Atlanta_40_4	15	22	40	320	154	1	14990	14990	0	2898,91	9029	1	14990	14990	0	6056,70	10063	132
Atlanta_50_1	15	22	50	320	193	11	18922	19083	0,85	18000	14364	2	18922	19202	1,48	18000	11931	92
Atlanta_50_2	15	22	50	320	199	1	18692	18692	0	3222,87	9593	11	18692	18692	0	2585,75	9721	45
Atlanta_50_3	15	22	50	320	195	13	18934	18934	0	5769,38	11498	7	18934	19078	0,76	18000	12748	77
Atlanta_50_4	15	22	50	320	184	21	19608	19800	0,98	18000	11904	15	19608	19608	0	3565	9926	19
France_10_1	25	44	10	36	320	1	25918	25918	0	1316,38	3845	1	25918	25918	0	1956,63	3845	0
France_10_2	25	44	10	40	320	3	33364	33364	0	1240,51	4520	1	33364	33364	0	1933,99	4520	7
France_10_3	25	44	10	32	320	1	35694	35694	0	753,72	4672	1	35694	35694	0	1188,21	4672	0
France_10_4	25	44	10	38	320	1	32105	32105	0	558,13	4835	1	32105	32105	0	927,38	4835	0
France_20_1	25	44	20	75	320	1	57991	57991	0	3011,14	6809	1	57991	57991	0	4638,58	6809	0
France_20_2	25	44	20	70	320	1	65609	65609	0	1550,95	6381	1	65609	65609	0	2416,44	6381	0
France_20_3	25	44	20	82	320	1	55432	55432	0	3794,91	5460	1	55432	55432	0	5371,12	5460	0
France_20_4	25	44	20	70	320	7	62668	62668	0	1614,27	6704	1	62668	62668	0	2457,92	6704	0
France_30_1	25	44	30	110	320	14	93439	93439	0	11236,74	8716	1	93439	93439	0	14503,67	8716	0
France_30_2	25	44	30	106	320	5	110248	110248	0	5622,95	11680	1	110248	110248	0	8195,66	11154	0
France_30_3	25	44	30	114	320	1	101356	101356	0	2425,3	7477	1	101356	101356	0	5561,17	8791	0
France_30_4	25	44	30	120	320	5	91264	91264	0	9505,11	12259	2	91264	91264	0	10680,01	13162	5
France_40_1	25	44	40	150	320	1	0	124321	INF	18000	17298	5	123210	123210	0	11787,3	13351	102
France_40_2	25	44	40	150	320	1	0	133835	INF	18000	13034	1	0	131954	INF	18000	15787	0
France_40_3	25	44	40	150	320	1	0	125606	INF	18000	15433	1	114926	114926	0	11093,36	10216	22
France_40_4	25	44	40	154	320	1	0	INF	INF	18000	11827	1	0	INF	INF	18000	13554	0
France_50_1	25	44	50	193	320	1	0	INF	INF	18000	9537	1	0	INF	INF	18000	12086	0
France_50_2	25	44	50	199	320	1	0	INF	INF	18000	11036	7	0	INF	INF	18000	10932	15
France_50_3	25	44	50	195	320	1	0	INF	INF	18000	9474	1	0	INF	INF	18000	11275	0
France_50_4	25	44	50	184	320	1	0	INF	INF	18000	12192	3	174775	177583	1,61	18000	13193	53

Table 3. Table of comparison for the B&P Vs the B&C&P using SND-Lib graphs.

Table 3 shows that the two algorithms B&P and B&C&P are very similar when considering only the Atlanta graph. However, as it appears the B&C&P was able to solve some instances to optimality that could not be solved to optimality when using the B&P. Moreover, the number of nodes in the branching tree and the number of generated columns decrease for some instances when using the B&C&P. The CPU time has also decreased for only a few instances when using the B&C&P. We also observe that the B&C&P solves some instances to optimality for which the linear relaxation has not been solved when using only B&P. Based on these results, we can attribute this improvement to our cutting-plane algorithm, specifically to the utilization of the valid inequalities.

Also, we propose a comparison between the B&P and the Learned_B&C&P in Table 4.

Instances						B&P						Learned_B&C&P						
Name	$\|V\|$	$\|E\|$	$\|K\|$	$\|S\|$	**SumWk**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Cuts**
Atlanta_10_1	15	22	10	320	36	1	3608	3608	0	352,35	3056	1	3608	3608	0	230,28	4039	0
Atlanta_10_2	15	22	10	320	40	1	3974	3974	0	550,83	4099	1	3974	3974	0	245,38	4552	0
Atlanta_10_3	15	22	10	320	32	1	4438	4438	0	534,28	3801	1	4438	4438	0	300,63	4650	0
Atlanta_10_4	15	22	10	320	38	1	4446	4446	0	1542,21	6239	1	4446	4446	0	324,28	5918	0
Atlanta_20_1	15	22	20	320	75	1	7972	7972	0	752,93	3839	1	7972	7972	0	453	6092	0
Atlanta_20_2	15	22	20	320	70	1	7864	7864	0	1201,85	4865	1	7864	7864	0	718,96	5921	0
Atlanta_20_3	15	22	20	320	82	1	8036	8036	0	1553,75	5614	1	8036	8036	0	986,09	7230	0
Atlanta_20_4	15	22	20	320	70	2	8026	8026	0	999,34	4825	1	8026	8026	0	417,69	4930	0
Atlanta_30_1	15	22	30	320	110	1	11188	11188	0	2263,65	6812	1	11188	11188	0	2452,68	10412	56
Atlanta_30_2	15	22	30	320	106	1	13602	13602	0	3359,67	7616	1	13602	13602	0	1682,21	10049	0
Atlanta_30_3	15	22	30	320	114	7	11748	11748	0	3440,12	7821	3	11748	11748	0	2315,83	10720	4
Atlanta_30_4	15	22	30	320	120	1	11248	11248	0	1096,13	6135	1	11248	11248	0	1200,33	7433	0
Atlanta_40_1	15	22	40	320	150	2	16456	16456	0	2912,13	8258	11	16456	16456	0	4189,60	11856	0
Atlanta_40_2	15	22	40	320	150	3	16324	16324	0	2870,33	9378	1	16324	16324	0	2538,70	13929	0
Atlanta_40_3	15	22	40	320	150	1	16650	16650	0	4069,14	10838	7	16650	16650	0	13611,14	16935	33
Atlanta_40_4	15	22	40	320	154	1	14990	14990	0	2898,91	9029	1	14990	14990	0	2251,98	12903	0
Atlanta_50_1	15	22	50	320	193	11	18922	19083	0,85	18000	14364	1	18922	18922	0	5815,76	17479	24
Atlanta_50_2	15	22	50	320	199	1	18692	18692	0	3222,87	9593	1	18692	18692	0	3479,28	11709	55
Atlanta_50_3	15	22	50	320	195	13	18934	18934	0	5769,38	11498	1	18934	18934	0	3612,93	15384	0
Atlanta_50_4	15	22	50	320	184	21	19608	19800	0,98	18000	11904	1	19608	19608	0	2185,87	11382	14
France_10_1	25	44	10	36	320	1	25918	25918	0	1316,38	3845	1	25918	25918	0	572,27	6336	0
France_10_2	25	44	10	40	320	3	33364	33364	0	1240,51	4520	11	33364	33364	0	932,72	6174	0
France_10_3	25	44	10	32	320	1	35694	35694	0	753,72	4672	1	35694	35694	0	691,56	6710	0
France_10_4	25	44	10	38	320	1	32105	32105	0	558,13	4835	1	32105	32105	0	568,85	6424	0
France_20_1	25	44	20	75	320	1	57991	57991	0	3011,14	6809	1	57991	57991	0	1016,91	8791	0
France_20_2	25	44	20	70	320	1	65609	65609	0	1550,95	6381	1	65609	65609	0	958,47	8918	0
France_20_3	25	44	20	82	320	1	55432	55432	0	3794,91	5460	8	55432	55432	0	3829,65	9247	37
France_20_4	25	44	20	70	320	7	62668	62668	0	1614,27	6704	1	62668	62668	0	1520,54	10235	0
France_30_1	25	44	30	110	320	14	93439	93439	0	11236,74	8716	1	93439	93439	0	8376,95	13868	23
France_30_2	25	44	30	106	320	5	110248	110248	0	5622,95	11680	1	110248	110248	0	3102,85	14645	0
France_30_3	25	44	30	114	320	1	101356	101356	0	2425,3	7477	1	101356	101356	0	6101,83	11714	0
France_30_4	25	44	30	120	320	5	91264	91264	0	9505,11	12259	1	91264	91264	0	2294,50	14476	0
France_40_1	25	44	40	150	320	1	0	124321	INF	18000	17298	1	123210	123210	0	3593,02	18752	0
France_40_2	25	44	40	150	320	1	0	133835	INF	18000	13034	5	131954	133551	1,21	18000	23153	0
France_40_3	25	44	40	150	320	1	0	125606	INF	18000	15433	3	114926	114926	0	8204,39	18432	94
France_40_4	25	44	40	154	320	1	0	INF	INF	18000	11827	3	124739	125452	0,57	18000	20552	25
France_50_1	25	44	50	193	320	1	0	INF	INF	18000	9537	3	146265	148259	1,36	18000	21858	7
France_50_2	25	44	50	199	320	1	0	INF	INF	18000	11036	2	153581	154007	0,28	18000	23845	32
France_50_3	25	44	50	195	320	1	0	INF	INF	18000	9474	1	150759	151915	0,77	18000	24162	54
France_50_4	25	44	50	184	320	1	0	INF	INF	18000	12192	1	174775	176742	1,13	18000	26678	28

Table 4. Table of comparison for the B&P Vs the Learned B&C&P using SND-Lib graphs.

The results presented in Table 4 show that the Learned_B&C&P outperforms the B&P algorithm. This improvement can be attributed to the utilization of our learning mechanism in the column generation and cutting plane algorithms, along with some additional valid inequalities have been generated. It can be observed that the Learned_B&C&P achieves a higher number of instances solved to optimality compared to the B&P algorithm. Furthermore, the Learned_B&C&P is capable to solve some instances for which the linear relaxation has not been solved when using the B&P algorithm. For the remaining instances, the results indicate a decrease in CPU time and optimality gap when using the Learned_B&C&P.

The observed improvement in Table 4 can be attributed to the utilization of the learning approach incorporated in both, the column generation and cutting-plane algorithms.

In Table 5 below, we discuss a comparison between the Learned_B&P and the B&C&P.

Instances						Learned_B&P						B&C&P						
Name	$\|V\|$	$\|E\|$	$\|K\|$	$\|S\|$	**SumWk**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Cuts**
Atlanta_10_1	15	22	10	320	36	1	3608	3608	0	691,57	4039	1	3608	3608	0	169,87	3056	0
Atlanta_10_2	15	22	10	320	40	1	3974	3974	0	926,70	4552	1	3974	3974	0	230,50	4099	0
Atlanta_10_3	15	22	10	320	32	1	4438	4438	0	1075,57	4650	1	4438	4438	0	373,26	4627	0
Atlanta_10_4	15	22	10	320	38	1	4446	4446	0	1169,49	5918	1	4446	4446	0	594,71	6239	0
Atlanta_20_1	15	22	20	320	75	1	7972	7972	0	1972,87	6092	1	7972	7972	0	383,57	4587	0
Atlanta_20_2	15	22	20	320	70	1	7864	7864	0	2116,77	5921	1	7864	7864	0	400,50	4865	0
Atlanta_20_3	15	22	20	320	82	1	8036	8036	0	2921,92	7230	1	8036	8036	0	495,03	5188	0
Atlanta_20_4	15	22	20	320	70	1	8026	8026	0	1236,22	4930	2	8026	8026	0	1125,76	4825	0
Atlanta_30_1	15	22	30	320	110	1	11188	11188	0	6017,35	10412	1	11188	11188	0	2426,22	6812	0
Atlanta_30_2	15	22	30	320	106	1	13602	13602	0	6337,66	10640	1	13602	13602	0	3532,76	7616	30
Atlanta_30_3	15	22	30	320	114	5	11748	11748	0	6030,61	10091	5	11748	11748	0	3657	7795	55
Atlanta_30_4	15	22	30	320	120	1	11248	11248	0	1502,82	7433	1	11248	11248	0	2805,91	6450	136
Atlanta_40_1	15	22	40	320	150	11	16456	16456	0	4751,02	11856	1	16456	16456	0	751,19	4684	0
Atlanta_40_2	15	22	40	320	150	1	16324	16324	0	4451,20	13929	13	16324	16324	0	7167,04	9877	64
Atlanta_40_3	15	22	40	320	150	1	16650	16650	0	4357,06	12206	15	16650	16650	0	2267,31	4589	8
Atlanta_40_4	15	22	40	320	154	1	14990	14990	0	3707,70	12903	1	14990	14990	0	6056,70	10063	132
Atlanta_50_1	15	22	50	320	193	1	18922	18922	0	5138,30	17479	2	18922	19202	1,48	18000	11931	92
Atlanta_50_2	15	22	50	320	199	5	18692	18692	0	3888,97	11504	11	18692	18692	0	2585,75	9721	45
Atlanta_50_3	15	22	50	320	195	1	18934	18934	0	4327,25	15384	7	18934	19078	0,76	18000	12748	77
Atlanta_50_4	15	22	50	320	184	5	19608	19608	0	2829,69	11382	15	19608	19608	0	3565	9926	19
France_10_1	25	44	10	36	320	1	25918	25918	0	552,39	6336	1	25918	25918	0	1956,63	3845	0
France_10_2	25	44	10	40	320	11	33364	33364	0	890,57	6174	1	33364	33364	0	1933,99	4520	7
France_10_3	25	44	10	32	320	1	35694	35694	0	683,37	6710	1	35694	35694	0	1188,21	4672	0
France_10_4	25	44	10	38	320	1	32105	32105	0	647,67	6424	1	32105	32105	0	927,38	4835	0
France_20_1	25	44	20	75	320	1	57991	57991	0	992,23	8791	1	57991	57991	0	4638,58	6809	0
France_20_2	25	44	20	70	320	1	65609	65609	0	905,46	8918	1	65609	65609	0	2416,44	6381	0
France_20_3	25	44	20	82	320	19	55432	55432	0	3931,66	9349	1	55432	55432	0	5371,12	5460	0
France_20_4	25	44	20	70	320	1	62668	62668	0	1696,09	10235	1	62668	62668	0	2457,92	6704	0
France_30_1	25	44	30	110	320	1	93439	93439	0	3882,17	14310	1	93439	93439	0	14503,67	8716	0
France_30_2	25	44	30	106	320	1	110248	110248	0	4150,55	14645	1	110248	110248	0	8195,66	11154	0
France_30_3	25	44	30	114	320	5	101356	101356	0	4284,27	13909	1	101356	101356	0	5561,17	8791	0
France_30_4	25	44	30	120	320	1	91264	91264	0	2237,30	14476	2	91264	91264	0	10680,01	13162	5
France_40_1	25	44	40	150	320	1	123210	123210	0	4306,91	18752	5	123210	123210	0	11787,3	13351	102
France_40_2	25	44	40	150	320	11	131954	132203	0,19	18000	25401	1	0	131954	INF	18000	15787	0
France_40_3	25	44	40	150	320	3	114926	114926	0	11646,78	16640	1	114926	114926	0	11093,36	10216	22
France_40_4	25	44	40	154	320	1	124739	125452	0,57	18000	20552	1	0	INF	INF	18000	13554	0
France_50_1	25	44	50	193	320	1	0	153554	INF	18000	21756	1	0	INF	INF	18000	12086	0
France_50_2	25	44	50	199	320	1	0	154452	INF	18000	21510	7	0	INF	INF	18000	10932	15
France_50_3	25	44	50	195	320	1	0	152886	INF	18000	21503	1	0	INF	INF	18000	11275	0
France_50_4	25	44	50	184	320	1	0	176742	INF	18000	23747	3	174775	177583	1,61	18000	13193	53

Table 5. Table of comparison for the Learned_B&P Vs the B&C&P using SND-Lib graphs.

Table 5 shows that the Learned_B&P performers better than the B&C&P which has not been the case when using only B&P. This is due to the usage of the learning mechanism in the column generation. We observe that the Learned_B&P solves more instances to optimality compared with the B&C&P. Moreover, we can note that the Learned_B&P was able to solve some instances to optimality that have not been solved to optimality when using the B&C&P. The results show also that the CPU time and optimality gap have decreased when using the Learned_B&P. There exists only one instance "France_50_4" for which the linear relaxation has not been solved when using the Learned_B&P.

In the rest of this section, we compare the Learned_B&P with the Learned_B&C&P.

Instances						Learned_B&P						Learned_B&C&P						
Name	$\|V\|$	$\|E\|$	$\|K\|$	$\|S\|$	**SumWk**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Nd**	**LB**	**UB**	**Gap**	**TT**	**Cols**	**Cuts**
Atlanta_10_1	15	22	10	320	36	1	3608	3608	0	691,57	4039	1	3608	3608	0	230,28	4039	0
Atlanta_10_2	15	22	10	320	40	1	3974	3974	0	926,70	4552	1	3974	3974	0	245,38	4552	0
Atlanta_10_3	15	22	10	320	32	1	4438	4438	0	1075,57	4650	1	4438	4438	0	300,63	4650	0
Atlanta_10_4	15	22	10	320	38	1	4446	4446	0	1169,49	5918	1	4446	4446	0	324,28	5918	0
Atlanta_20_1	15	22	20	320	75	1	7972	7972	0	1972,87	6092	1	7972	7972	0	453	6092	0
Atlanta_20_2	15	22	20	320	70	1	7864	7864	0	2116,77	5921	1	7864	7864	0	718,96	5921	0
Atlanta_20_3	15	22	20	320	82	1	8036	8036	0	2921,92	7230	1	8036	8036	0	986,09	7230	0
Atlanta_20_4	15	22	20	320	70	1	8026	8026	0	1236,22	4930	1	8026	8026	0	417,69	4930	0
Atlanta_30_1	15	22	30	320	110	1	11188	11188	0	6017,35	10412	1	11188	11188	0	2452,68	10412	56
Atlanta_30_2	15	22	30	320	106	1	13602	13602	0	6337,66	10640	1	13602	13602	0	1682,21	10049	0
Atlanta_30_3	15	22	30	320	114	5	11748	11748	0	6030,61	10091	3	11748	11748	0	2315,83	10720	4
Atlanta_30_4	15	22	30	320	120	1	11248	11248	0	1502,82	7433	1	11248	11248	0	1200,33	7433	0
Atlanta_40_1	15	22	40	320	150	11	16456	16456	0	4751,02	11856	11	16456	16456	0	4189,60	11856	0
Atlanta_40_2	15	22	40	320	150	1	16324	16324	0	4451,20	13929	1	16324	16324	0	2538,70	13929	0
Atlanta_40_3	15	22	40	320	150	1	16650	16650	0	4357,06	12206	7	16650	16650	0	13611,14	16935	33
Atlanta_40_4	15	22	40	320	154	1	14990	14990	0	3707,70	12903	1	14990	14990	0	2251,98	12903	0
Atlanta_50_1	15	22	50	320	193	1	18922	18922	0	5138,30	17479	1	18922	18922	0	5815,76	17479	24
Atlanta_50_2	15	22	50	320	199	5	18692	18692	0	3888,97	11504	1	18692	18692	0	3479,28	11709	55
Atlanta_50_3	15	22	50	320	195	1	18934	18934	0	4327,25	15384	1	18934	18934	0	3612,93	15384	0
Atlanta_50_4	15	22	50	320	184	5	19608	19608	0	2829,69	11382	1	19608	19608	0	2185,87	11382	14
France_10_1	25	44	10	36	320	1	25918	25918	0	552,39	6336	1	25918	25918	0	572,27	6336	0
France_10_2	25	44	10	40	320	11	33364	33364	0	890,57	6174	11	33364	33364	0	932,72	6174	0
France_10_3	25	44	10	32	320	1	35694	35694	0	683,37	6710	1	35694	35694	0	691,56	6710	0
France_10_4	25	44	10	38	320	1	32105	32105	0	647,67	6424	1	32105	32105	0	568,85	6424	0
France_20_1	25	44	20	75	320	1	57991	57991	0	992,23	8791	1	57991	57991	0	1016,91	8791	0
France_20_2	25	44	20	70	320	1	65609	65609	0	905,46	8918	1	65609	65609	0	958,47	8918	0
France_20_3	25	44	20	82	320	19	55432	55432	0	3931,66	9349	8	55432	55432	0	3829,65	9247	37
France_20_4	25	44	20	70	320	1	62668	62668	0	1696,09	10235	1	62668	62668	0	1520,54	10235	0
France_30_1	25	44	30	110	320	1	93439	93439	0	3882,17	14310	1	93439	93439	0	8376,95	13868	23
France_30_2	25	44	30	106	320	1	110248	110248	0	4150,55	14645	1	110248	110248	0	3102,85	14645	0
France_30_3	25	44	30	114	320	5	101356	101356	0	4284,27	13909	1	101356	101356	0	6101,83	11714	0
France_30_4	25	44	30	120	320	1	91264	91264	0	2237,30	14476	1	91264	91264	0	2294,50	14476	0
France_40_1	25	44	40	150	320	1	123210	123210	0	4306,91	18752	1	123210	123210	0	3593,02	18752	0
France_40_2	25	44	40	150	320	11	131954	132203	0,19	18000	25401	5	131954	133551	1,21	18000	23153	0
France_40_3	25	44	40	150	320	3	114926	114926	0	11646,78	16640	3	114926	114926	0	8204,39	18432	94
France_40_4	25	44	40	154	320	1	124739	125452	0,57	18000	20552	3	124739	125452	0,57	18000	20552	25
France_50_1	25	44	50	193	320	1	0	153554	INF	18000	21756	3	146265	148259	1,36	18000	21858	7
France_50_2	25	44	50	199	320	1	0	154452	INF	18000	21510	2	153581	154007	0,28	18000	23845	32
France_50_3	25	44	50	195	320	1	0	152886	INF	18000	21503	1	150759	151915	0,77	18000	24162	54
France_50_4	25	44	50	184	320	1	0	176742	INF	18000	23747	1	174775	176742	1,13	18000	26678	28

Table 6. Table of comparison for the Learned_B&P Vs the Learned_B&C&P using SND-Lib graphs.

Based on the previous results and the results presented in Table 6, it is evident to see that both, the Learned_B&P and Learned_B&C&P algorithms yield superior results compared to the B&P and B&C&P algorithms. Moreover, the results in Table 6 indicate that the Learned_B&C&P outperforms the Learned_B&P algorithm thanks to our learned cutting-plane algorithm and specifically to our additional valid inequalities. The Learned_B&C&P algorithm demonstrates a decrease in the number of nodes in the branching tree, CPU time, and gap. Additionally, it is capable to solve instances with a small gap for which the linear relaxation remained unsolved when using the Learned_B&P algorithm.

All the results presented in this section highlight the effectiveness of the learning and cutting plane algorithms in enhancing the performance of the Branch-and-Cut-and-Price algorithm.

9 Conclusion

In this paper, we have addressed the Survivable Constrained-Routing and Spectrum Assignment problem. The consideration of survivability is crucial in optical networks to ensure continuous service and robustness against failures, making the development of efficient algorithms for survivable routing and spectrum assignment highly significant. For this, we have first introduced an extended formulation to model the problem. We described various classes of valid inequalities to strengthen the formulation and developed separation routines to efficiently identify violated inequalities. Based on these results, we have develop a Branch-and-Cut-and-Price algorithm to solve the problem. An extensive computational study is conducted to evaluate the performance of the algorithm. A machine learning approach has been used to improve the performance of the Branch-and-Cut-and-Price. Although our approach has shown effectiveness, it would be valuable to compare it with a Branch-and-Cut algorithm that utilizes a compact formulation.

Additionally, we plan to extend these approaches to shared protection schemes, which are beneficial in large-scale networks by allowing multiple demands to share backup resources when failures are independent and working paths do not share edges. This can significantly reduce backup resource needs compared to dedicated protection. Comparing shared and dedicated protection schemes would offer valuable insights into their relative efficiencies.

Furthermore, exploring the same approaches in the context of the basic Constrained Routing and Spectrum Assignment (CRSA) problem, without survivability considerations, could yield valuable insights into their strengths and limitations for large instances. Such a comparison would help identify the most suitable approaches for different scenarios.

Looking ahead, we aim to extend this methodology to related problems such as the Routing, Modulation, and Spectrum Assignment (RMSA) problem, considering both scenarios with and without survivability constraints. This extension involves incorporating modulation formats and their impact on spectrum efficiency, as well as addressing the challenges of joint routing, modulation level selection, and spectrum allocation to optimize network performance. Moreover, we plan to apply our approach to the Routing and Spectrum Assignment (RSA) problem, focusing on resource optimization and resilience, with particular attention to large-scale network scenarios where efficient resource utilization and survivability are critical.

This future research will evaluate the performance and scalability of our approaches across various problem settings, thereby contributing to the development of more efficient, resilient, and adaptable optical network design strategies.

Acknowledgement

We are grateful to the anonymous referees for their constructive comments that permitted to correct some errors introduced in the initial version, and significantly enhance the clarity and quality of the presentation.

References

1. Bertero, F., Bianchetti, M., and Marenco, J.: Integer programming models for the routing and spectrum allocation problem. In: Official Journal of the Spanish Society of Statistics and Operations Research 2018, pp. 465-488.
2. Bienstock, D., Chopra, S., Günlük, O. and Tsai, C-Y.: Minimum cost capacity installation for multicommodity network flows. In: Math. Program., 1995, 81, pp. 177–199.
3. Cai, A., Shen, G., Peng, L., and Zukerman, M.: Novel Node-Arc Model and Multiiteration Heuristics for Static Routing and Spectrum Assignment in Elastic Optical Networks. In: J. of Lightwave Technology 2013, pp. 3402-3413.
4. Cai, A., Guo, J., Lin, R., Shen, G., and Zukerman, M.:"Multicast routing and distance-adaptive spectrum allocation in elastic optical networks with shared protection. In: J. of Lightwave Technology, 34(17), 2016, pp. 4076–4088.
5. Chen, X., Guo, J., Zhu, Z., Proietti, R., Castro, A., and Yoo, S.J.B.: Deep-RMSA: A Deep-Reinforcement-Learning Routing, Modulation and Spectrum Assignment Agent for Elastic Optical Networks. In: Optical Fiber Communications Conference and Exposition (OFC) 2018, pp. 1-3.
6. Choudhury, P. D., Bhadra, S. and De, T.: "A Brief Review of Protection based Routing and Spectrum Assignment in Elastic Optical Networks and a novel P-cycle based protection approach for Multicast Traffic Demands". In: Optical Switching and Networking, 2019, pp. 1-21.
7. Christodoulopoulos, K., Tomkos, I., and Varvarigos, E.A.: Elastic Bandwidth Allocation in Flexible OFDM-Based Optical Networks. In: Lightwave Technology 2011, pp. 1354-1366.
8. Cplex, I.I., 2020. V12. 9: User's Manual for CPLEX. International Business Machines Corporation, 46(53), pp. 157.
9. Crowder, H., Johnson, E.L., and Padberg, M.W.: Solving large scale zero-one linear programming problems. In: Operations Research 1983, 31, pp. 803-834. 1983;31:803–34.
10. Diarrassouba, I., Hadhbi, Y. and Mahjoub, A. R.: The Constrained-Routing and Spectrum Assignment Problem: Extended Formulation and Branch-and-Cut-and-Price Algorithm. In: CoDIT 2022, pp. 926-931.
11. Diarrassouba, I., and Hadhbi, Y.: The Constrained-Routing and Spectrum Assignment Problem: Valid Inequalities and Branch-and-Cut Algorithm. In: Ljubić, I., Barahona, F., Dey, S.S., Mahjoub, A.R. (eds) Combinatorial Optimization, ISCO 2022, Lecture Notes in Computer Science, vol 13526, pp. 35-47.

12. Diarrassouba, I., Hadhbi, Y. and Mahjoub, A. R.: Branch-and-cut-and-price algorithm for the constrained-routing and spectrum assignment problem. In: J. of Combinatorial Optimization, 47, 56, 2024.
13. Dror, M. : Note on the Complexity of the Shortest Path Models for Column Generation in VRPTW. In: J. of Operations Research, 1994, pp. 977-978.
14. Dumitrescu, I., and Boland, N.: Algorithms for the weight constrained shortest path problem. In: International Transactions in Operational Research, 2001, pp. 15-29.
15. Fayez, M., Katib, I., George, N.R., Gharib, T.F., Khaleed H., and Faheem, H. M.: Recursive algorithm for selecting optimum routing tables to solve offline routing and spectrum assignment problem. In: Ain Shams Engineering Journal 2020, pp. 273-280.
16. Goldberg, A.V., and Tarjan, R. E.: A New Approach to the Maximum Flow Problem. In: Proceedings of the Eighteenth Annual Association for Computing Machinery Symposium on Theory of Computing 1986, pp. 136-146.
17. Grötschel, M., Lovász, L., Schrijver, A.: Stable Sets in Graphs. In: Geometric Algorithms and Combinatorial Optimization. Algorithms and Combinatorics, 1988, vol 2. Springer, Berlin, Heidelberg.
18. Goscien, R., Walkowiak, K. and Klinkowski, M.: "Joint anycast and unicast routing and spectrum allocation with dedicated path protection in elastic optical networks". In: Design of Reliable Communication Networks (DRCN), 2014 10th International Conference, IEEE, 2014, pp. 1–8.
19. Gu, R., Yang, Z., and Ji, Y.: Machine Learning for Intelligent Optical Networks: A Comprehensive Survey. In : Journal CoRR 2020, pp. 1-42.
20. Hadhbi, Y., Kerivin, H., and Wagler, A.: A novel integer linear programming model for routing and spectrum assignment in optical networks. In: Federated Conference on Computer Science and Information Systems (FedCSIS) 2019, pp. 127-134.
21. Hai, D.H., and Hoang, K. M.: An efficient genetic algorithm approach for solving routing and spectrum assignment problem. In: J. of Recent Advances in Signal Processing 2017.
22. Hai, D.H., Morvan, M., and Gravey, P.: Combining heuristic and exact approaches for solving the routing and spectrum assignment problem. In: J. of Iet Optoelectronics 2017, pp. 65-72.
23. He, S., Qiu, Y., and Xu, J.: Invalid-Resource-Aware Spectrum Assignment for Advanced-Reservation Traffic in Elastic Optical Network. In: Sensors 2020.
24. Jaumard, B., and Daryalal, M.: Scalable elastic optical path networking models. In: 18th International Conference Transparent Optical Networks (ICTON) 2016, pp. 1-4.
25. Klinkowski, M., and Walkowiak, K.: Routing and Spectrum Assignment in Spectrum Sliced Elastic Optical Path Network. In: IEEE Communications Letters 2011, pp. 884-886.
26. https://lemon.cs.elte.hu/trac/lemon.
27. Lezama, F., Martinez-Herrera, A. F., Castanon, G., Del-Valle-Soto, C., Sarmiento, A. M., Munoz de Cote, A.: Solving routing and spectrum allocation problems in flexgrid optical networks using precomputing strategies. In: J. of Photon Netw Commun 41, pp. 17-35.

28. Liu, L., Yin, S., Zhang, Z., Chu, Y., and Huang, S.: A Monte Carlo Based Routing and Spectrum Assignment Agent for Elastic Optical Networks. In: Asia Communications and Photonics Conference (ACP) 2019, pp. 1-3.
29. Lohani, V., Sharma, A., and Singh, Y. N.: Routing, Modulation and Spectrum Assignment using an AI based Algorithm. In: 11th International Conference on Communication Systems & Networks (COMSNETS) 2019, pp. 266-271.
30. Mahala, N., and Thangaraj, J.: Spectrum assignment technique with first-random fit in elastic optical networks. In : 4th International Conference on Recent Advances in Information Technology (RAIT) 2018, pp. 1-4.
31. Mesquita, L.A.J., Assis, K., Santos, A. F., Alencar, M., and Almeida, R.C.: A Routing and Spectrum Assignment Heuristic for Elastic Optical Networks under Incremental Traffic. In: SBFoton International Optics and Photonics Conference (SBFoton IOPC) 2018, pp. 1-5.
32. Nemhauser, G.L., and Wolsey, L.A.: Integer and Combinatorial Optimization. In: John Wiley 1988.
33. Nemhauser, G. L., and Sigismondi, G.: A Strong Cutting Plane/Branch-and-Bound Algorithm for Node Packing. In: The Journal of the Operational Research Society 1992, pp. 443-457.
34. Orlowski, S., Pióro, M., Tomaszewski, A. , and Wessäly, R.: SND-Lib 1.0-Survivable Network Design Library. In: Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium, http://www.zib.de/orlowski/Paper/OrlowskiPioroTomaszewskiWessaely2007-SND-Lib-INOC.pdf.gz.
35. Padberg, M. W.: On the facial structure of set packing polyhedra. In: J. of Mathematical Programming 1973, pp. 199-215.
36. Karp, R. M.: Reducibility among Combinatorial Problems. In: Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center 1972, pp. 85-94.
37. Klabjan, D., Nemhauser, G.L., and Tovey, C.: The complexity of cover inequality separation. In: J. of Operations Research Letters 1998, pp. 35-40.
38. Klinkowski, M., and Walkowiak, K.: Routing and Spectrum Assignment in Spectrum Sliced Elastic Optical Path Network. In: IEEE Communications Letters 2011, pp. 884-886.
39. Klinkowski, M., Pioro, M., Zotkiewicz, M., Ruiz, M., and Velasco, L.: Valid inequalities for the routing and spectrum allocation problem in elastic optical networks. In: 16th International Conference on Transparent Optical Networks (ICTON) 2014, pp. 1-5.
40. Klinkowski, M., Pioro, M., Zotkiewicz, M., Ruiz, M., and Velasco, L.: A Simulated Annealing Heuristic for a Branch and Price-Based Routing and Spectrum Allocation Algorithm in Elastic Optical Networks. In: Intelligent Data Engineering and Automated Learning – IDEAL 2015, Springer International Publishing, pp. 290-299.
41. Reihani, A.Z, Behdadfar, M., and Sebghati, M.: Artificial neural network-based adaptive modulation for elastic optical networks. In: Internet Technology Letters Journal, 11 April 2021, pp. 1-6.

42. Ruiz, M., Pioro, M., Zotkiewicz, M., Klinkowski, M., and Velasco, L. : A hybrid meta-heuristic approach for optimization of routing and spectrum assignment in Elastic Optical Network (EON). In: J. of Enterprise Information Systems 2020, pp. 11-24.

43. Ruiz, M., Pioro, M., Zotkiewicz, M., Klinkowski, M., and Velasco, L.: Column generation algorithm for RSA problems in flexgrid optical networks. In: Photonic Network Communications 2013, pp. 53-64.

44. Salameh, B.H., Qawasmeh, R., and Al-Ajlouni, A.F.: Routing With Intelligent Spectrum Assignment in Full-Duplex Cognitive Networks Under Varying Channel Conditions. In: J. of IEEE Communications Letters 2020, pp. 872-876.

45. Salani, M., Rottondi, C., and Tornatore, M.: Routing and Spectrum Assignment Integrating Machine-Learning-Based QoT Estimation in Elastic Optical Networks. In: IEEE INFOCOM - IEEE Conference on Computer Communications 2019, pp. 173846.

46. Gamrath, G., Anderson, D., Bestuzheva, K., Chen, W.K., Eifler, L., Gasse, M., Gemander, P., Gleixner, A., Gottwald, L., Halbig, K., and Hendel, G., and Hojny, C., Koch, T., Bodic, L., Maher, P. J., Matter, F., Miltenberger, M., Mühmer, E., Müller, B., Pfetsch, M.E., Schlösser, F., Serrano, F., Shinano, Y., Tawfik, C., Vigerske, S., Wegscheider, F., Weninger, D., and Witzig, J.: The SCIP Optimization Suite 7.0. In: http://www.optimization-online.org/DB_HTML/2020/03/7705.html, March 2020.

47. Shirazipourazad, S., Zhou, C., Derakhshandeh, Z., and Sen, A.: On routing and spectrum allocation in spectrum-sliced optical networks. In: Proceedings IEEE INFOCOM 2013, pp. 385-389.

48. Pedro H. Fernandes da Silva, Kerivin, H., Nant, J. P., and Wagler, A.: A framework for routing and spectrum assignment in optical networks, driven by combinatorial properties. In: the International Network Optimization Conference (INOC), 2022, pp. 1-6.

49. Talebi, S., Alam, F., Katib, I., Khamis, M., Salama, R., and Rouskas, G. N.: Spectrum management techniques for elastic optical networks: A survey. In: Optical Switching and Networking 2014.

50. Velasco, L., Klinkowski, M., Ruiz, M., and Comellas, J.: Modeling the routing and spectrum allocation problem for flexgrid optical networks. In: Photonic Network Communications 2012, pp. 177-186.

51. Xuan, H., Wang, Y., Xu, Z., Hao, S., and Wang, X.: New bi-level programming model for routing and spectrum assignment in elastic optical network. In: Opt Quant Electron 49-2017, pp. 1-16.

52. Zhang, Y., Xin, J., and Li, X., and Huang, S.: Overview on routing and resource allocation based machine learning in optical networks. In: J. of Optical Fiber Technology, pp. 1-21.

53. Ziazet, J.M., and Jaumard, B.: Reinforcement Learning for Routing, Modulation And Spectrum Assignment Problem in Elastic Optical Networks. In: Partial Fulfillment of a Cooperative Masters Degree in Industrial Mathematics at AIMS-Cameroon, December 20, 2019, pp. 1-56.

54. Zotkiewicz, M., Pioro, M., Ruiz, M., Klinkowski, M., and Velasco, L.: Optimization models for flexgrid elastic optical networks. In: 15th International Conference on Transparent Optical Networks (ICTON) 2013, pp. 1-4.