Crew Scheduling and Routing Problem in Road Restoration via Branch-and-Price Algorithms

Alfredo Moreno
Department of Logistics and Operations Management, HEC Montréal, Montreal, Canada
alfredo-daniel.moreno-arteaga@hec.ca

Pedro Munari
Production Engineering Department, Federal University of São Carlos, São Carlos/SP, Brazil
munari@dep.ufscar.br

Douglas Alem
University of Edinburgh Business School, Management Science and Business Economics Group, Edinburgh, UK
douglas.alem@ed.ac.uk

This paper addresses the single crew scheduling and routing problem in the context of road network repair and restoration, which is critical in assisting complex post-disaster decisions in humanitarian logistics settings. We present three novel formulations for this problem, which are the first suitable for column generation and branch-and-price (BP) algorithms. Specifically, our first formulation is based on enumerating crew schedules and routes while explicitly defining the relief paths. The second formulation relies on enumerating the schedules, routes, and relief paths. Finally, the third formulation builds upon the second one by including additional constraints and variables related to relief path decisions. Considering each formulation, we propose BP algorithms that rely on several enhancements, including a new dynamic programming labeling algorithm to efficiently solve the subproblems. Extensive computational results based on 648 benchmark instances reveal that our BP algorithms significantly outperform existing exact approaches, solving 450 instances to optimality, and remarkably 118 instances for the first time. Our framework is also very effective in improving the lower bounds, upper bounds, and optimality gaps that have been reported in the literature.

Key words: road restoration; network repair; humanitarian logistics; column generation; branch-and-price.

1. Introduction

Large-scale disasters such as hurricanes, earthquakes, floods, and landslides are examples of events that usually have devastating effects on transportation network systems such as roads, bridges, and tunnels. As a consequence, affected areas may become inaccessible, causing delays in the relief distribution as well as in evacuation and rescue operations (Almeida, Goerlandt, and Pelot 2022), which often results in victim suffering and loss of life (Moreno et al. 2020). The 2008 Wenchuan earthquake in China, for example, caused destruction to the transport system of 53,289 km, which includes 24 highways and 161 state or local roads, leaving many villages and towns isolated from cities, distribution centers, and/or supply depots (Li, Ma, and Teo 2020). Similarly, during the 2010 Port-au-Prince earthquake in Haiti, affected areas had difficulty accessing essential services such as the airport and seaport because of the large amount of debris blocking the roads (Bono and Gutiérrez 2011). The heavy road damage associated with the 2010 Hurricane Igor in Atlantic Canada isolated communities for up to eleven days (Masson 2014). These and many other real-world examples including hurricanes, earthquakes, rainfalls, and landslides highlight the importance of restoring the transportation infrastructure to effectively respond to disasters (Moreno et al. 2018, Caunhye, Aydin, and Duzgun 2020).

Road restoration activities include removing debris, rearranging power lines, draining flooded sections, installing temporary bridges, applying temporary road patches, and installing emergency signage, among others. Within the context of humanitarian logistics, these activities belong to the so-called *disaster recovery phase*, and their main goal is to "restore the system to the greatest extent possible and stabilize the community involved" (Çelik et al. 2012). The restoration of the infrastructure network span from hours to several days, depending on the number of damaged locations that need to be repaired, the complexity of the repair operations, and the distance between the locations to be repaired. The time spent by the restoration crew to reach out to the damaged locations is also a key aspect of the problem when the repair operations must be done at different locations over a vast area. For instance, according to a case-study based on the 2011 megadisaster in the Serrana Region of Brazil (Moreno et al. 2020), a crew can spend more than half of the total restoration time travelling to reach the damaged locations.

In this paper, we are particularly interested in an important variant of road restoration problems, the so-called *single crew scheduling and routing problem* (SCSRP). The SCSRP is typically defined on an undirected graph in which the set of vertices is split into a subset of damaged vertices, representing the damaged locations over the road network; a subset of demand vertices, which represent the affected areas that must become accessible in order to get humanitarian assistance; and a subset of intersection vertices, which represent the connection of multiple road segments or highways, but there is no demand and no damage associated to them. It considers that a highway

or road segment can have one or more damaged locations (damaged vertices), as may occur in real cases, especially on long highways. For instance, in the 2011 megadisaster in the Serrana Region of Brazil, various roads longer than 30 kilometers suffered damage, but only specific points or locations needed repair intervention rather than the entire highway. The road network also comprises a source vertex or depot to be connected to the demand vertices. A single crew is available to perform restoration activities over the damaged network.

The SCSRP aims to find out the relief paths, defined as the sequence of edges and vertices necessary to connect the depot to affected areas; and the schedule or order in which the damaged vertices must be repaired, which also entails the route that must be followed by the crew to perform the restoration and return to the depot at the end of the operation. Damaged vertices cannot be traversed unless they have been repaired. The goal is to restate the accessibility of the affected areas as soon as possible. An affected area is called *accessible* when there exists a path connecting it to a central supply depot using only undamaged and/or repaired vertices. Therefore, a critical subset of damaged vertices must be repaired in order to restate the system accessibility.

Notably, the SCSRP involves a complex interdependence between crew scheduling and routing decisions. The design of crew routes is challenging because the vertices and edges available at a given moment in the time horizon depend on which vertices have been repaired, which in turn depend on scheduling decisions. On the other hand, scheduling decisions also depend on routing. Defining the schedule without considering routing decisions can lead to infeasible solutions in practice, since damaged vertices that are not accessible (i.e., there is no feasible route to reach them) at a given moment might be selected first in the schedule. Furthermore, the shortest paths between damaged vertices change dynamically during restoration according to the schedule.

Because of the computational complexity of the SCSRP and related road restoration problems, as well as the fact that practical optimization instances of those problems are usually large-scale, most solution approaches are heuristics. To the best of our knowledge, only Maya-Duque, Dolinskaya, and Sörensen (2016), Moreno, Munari, and Alem (2019) and Moreno, Munari, and Alem (2020) have developed exact methods. The dynamic programming (DP) algorithm proposed by Maya-Duque, Dolinskaya, and Sörensen (2016) has not been able to solve even some small instances within 24-hour time limit. Additionally, although the BBC algorithms proposed by Moreno, Munari, and Alem (2019, 2020) provide feasible solutions and valid lower bounds for all their tested instances, the corresponding optimality gaps are relatively high even for some small instances, probably because of poor lower bounds provided by the weak linear programming (LP) relaxations of the formulations available so far. Therefore, this paper presents three new mathematical formulations for the SCSRP, which are based on the enumeration of the crew scheduling, crew routing, and relief path decisions. The main advantage of these formulations is that they have stronger LP relaxations than existing

formulations. Since the complete enumeration of the schedules, routes, and relief paths may be impractical for large-scale instances, we devise branch-and-price (BP) algorithms to effectively solve the proposed formulations. BP is a branch-and-bound scheme that solves the linear relaxation at each node of the branching tree with a column generation (CG) algorithm. This strategy has yielded successful results in many related applications such as vehicle routing (Costa, Contardo, and Desaulniers 2019, Munari et al. 2019) and scheduling (Legrain, Omer, and Rosat 2020, Lin, Juan, and Chang 2020).

The overall contributions of this paper are threefold:

(i) We introduce for the first time in the authors' best knowledge three mathematical formulations for the SCSRP that are suitable to be solved by means of BP algorithms. The first formulation is based on the enumeration of the crew schedules and routes and explicitly defines the relief paths. Hence, in a CG setting, there is a single subproblem that determines crew schedules and routes. The second formulation considers the enumeration of the schedules, routes, and relief paths. In this case, columns can be generated resorting to two subproblems, one for generating schedules and routes and another for generating relief paths. The third formulation incorporates additional constraints and variables related to relief path decisions into the second formulation. These formulations have a stronger LP relaxation than existing ones, and significantly improve the lower bound obtained at the root node of the branch-and-bound tree.

(ii) We derive BP algorithms to solve the novel SCSRP formulations based on effective CG strategies. Also, three heuristics based on the BP algorithms are devised to solve the problem. In the heuristics, the pricing subproblems are not solved to optimality.

(iii) A labeling algorithm is devised to solve the pricing subproblems via DP. While labeling algorithms considering time-dependent costs are common in the literature (Ioachim et al. 1998, Desaulniers and Villeneuve 2000, Dabia et al. 2013), we propose a number of enhancement strategies that make our labeling algorithm capable of solving the pricing subproblem efficiently. To demonstrate the impact of the proposed strategies, a sensitivity analysis is further conducted.

The efficiency of the novel approaches is assessed using 648 benchmark instances. Remarkably, we managed to optimally solve 450 (69.44%) instances, 118 for the first time. Moreover, our BP algorithms provide new better feasible solutions for 67 instances, while improve the lower bounds, upper bounds, and optimality gaps by 13.3%, 3.8%, and 26.26%, respectively, on average.

The remainder of this paper is organized as follows. Section 2 reviews the relevant literature related to road restoration problems and BP algorithms. Section 3 describes the SCSRP and presents a compact formulation for this problem. Section 4 derives the new mathematical formulations, while Section 5 develops the BP algorithms. Finally, we report the computational results and analyses in Section 6 and draw the concluding remarks in Section 7.

2. Related Literature

This section reviews two literature streams pertinent to our paper. Namely, the SCSRP and other road restoration problems are discussed in Section 2.1, while the theme of BP algorithms applied to solving different scheduling and routing problems is presented in Section 2.2.

2.1. SCSRP and Related Problems

The SCSRP is an NP-hard problem (Moreno et al. 2020) that has been tackled recently in the literature by means of exact methods and heuristics. Maya-Duque, Dolinskaya, and Sörensen (2016) introduced the SCSRP and proposed a DP algorithm to solve it. However, the DP algorithm was able to solve to optimality only a few small instances of the problem. Therefore, because of the limitations regarding their DP algorithm, the authors developed a metaheuristic based on GRASP to solve medium and large instances. Later, Kim et al. (2018) and Shin, Kim, and Moon (2019) proposed ant colony algorithms to solve the problem. Different from Maya-Duque, Dolinskaya, and Sörensen (2016), Kim et al. (2018) defined a golden period for the repair operations and penalized the accessibility after the golden period at a higher rate. Shin, Kim, and Moon (2019) considered additional relief goods distribution decisions within the SCSRP, minimizing the relief distribution time. Moreno, Munari, and Alem (2019) developed a branch-and-Benders-cut (BBC) algorithm to solve the SCSRP. Additionally, they proposed a heuristic algorithm to find feasible solutions and warm-start the BBC. The authors provided, for the first time, a valid lower bound for all the tested instances of the problem. Moreno, Munari, and Alem (2020) enhanced this BBC approach and hybridized it with two metaheuristics, namely simulated annealing and genetic algorithm. Moreno et al. (2020) proposed new formulations and valid inequalities for the problem considering multiple heterogeneous crews, and presented managerial insights based on small instances. Finally, similar to Shin, Kim, and Moon (2019), Lakzaei et al. (2023) considered relief goods distribution decisions to minimize both, the total relief time of demand vertices and the total restoration time of the damaged vertices.

The literature on related variants of road restoration problems has been active in the last years as well. Exact approaches have involved compact formulations solved with general-purpose mixed-integer programming solvers (Yan and Shih 2009, Kasaei and Salman 2016, Sahin, Kara, and Karasan 2016, Akbari and Salman 2017a, Akbari and Sayarshad 2022, Farzaneh et al. 2023). The proposed heuristic methods have considered different strategies such as the division of the original damaged network into smaller subnetworks (Yan and Shih 2007, 2009, Tuzun Aksu and Ozdamar 2014); the relaxation of some characteristics of the problem (Çelik, Ergun, and Keskinocak 2015, Berktaş, Kara, and Karaşan 2016, Akbari and Salman 2017b,a); and construction algorithms followed by improvement steps (Yan and Shih 2009, Kasaei and Salman 2016, Sahin, Kara, and Karasan

2016, Akbari, Sadati, and Kian 2021). Moreover, matheuristics (Ajam, Akbari, and Salman 2022) and several metaheuristics (Ajam, Akbari, and Salman 2019, Li and Teo 2019, García-Alviz et al. 2021, Elifcan, Dilek, and Linet 2022, Souza Almeida and Goerlandt 2022) have been successfully developed in this context.

We are not aware of any BP algorithm developed thus far for the SCSRP or related problems. Furthermore, the few exact methods proposed for the SCSRP show relatively high optimality gaps even for small instances. For a comprehensive review of network repair and road restoration in humanitarian operations, the reader is referred to Çelik (2016) and Almeida, Goerlandt, and Pelot (2022).

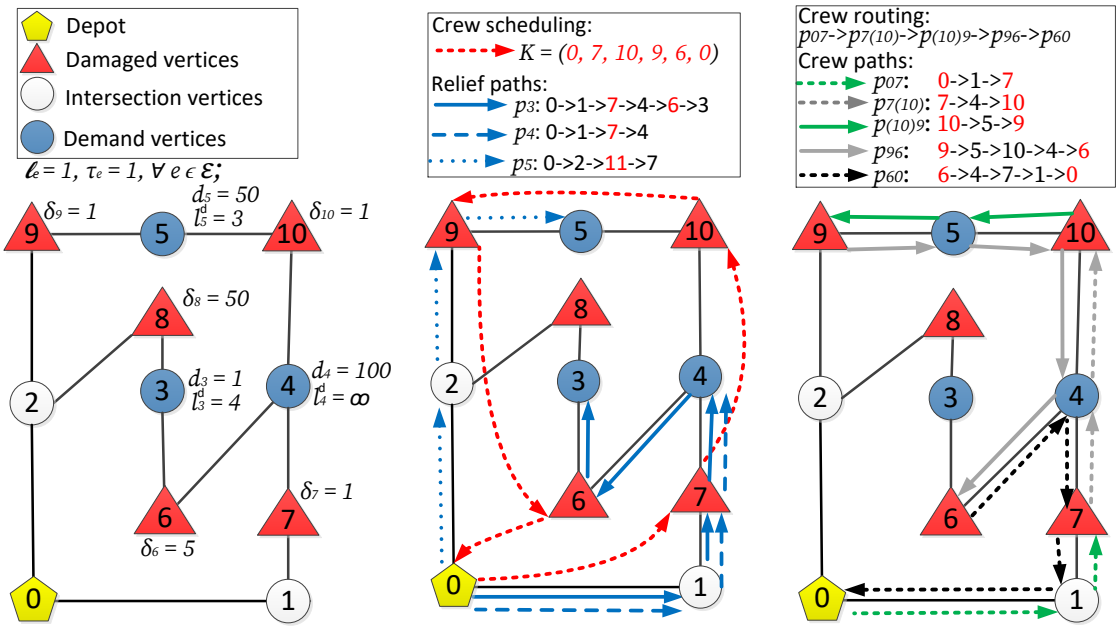
2.2. BP Algorithms for Scheduling and Routing problems

A variety of scheduling and routing problems have been solved using BP algorithms. For example, we find BP algorithms for the ship routing and scheduling problem (Stålhane et al. 2012), the home health care scheduling and routing problem (Yuan, Liu, and Jiang 2015), the technician routing and scheduling problem (Zamorano and Stolletz 2017), the aircraft scheduling and routing problem (Ruther et al. 2017), the vehicle and crew scheduling problem (Horváth and Kis 2019), and the bus driver scheduling and rostering problem (Lin, Juan, and Chang 2020). However, there are some differences between the SCSRP and these problems that prevent their BP algorithms from being straightforwardly applied to the SCSRP. First, unlike other scheduling and routing problems, the vertices that need to be visited (repaired) in the SCSRP are not given a priori. In fact, deciding which vertices need to be repaired is itself a combinatorial problem associated with the relief paths definition. Second, edges and vertices can be traversed multiple times in the SCSRP, while this is not allowed in other scheduling and routing problems; the shortest path between vertices dynamically change over time; and the service times at vertices also change over time. A damaged vertex has a service time greater than zero the first time it is visited (corresponding to the repair time), but the service time is zero on subsequent visits to this same node. Third, the considered network in the SCSRP is not a complete graph and, thus, the vertices that have to be repaired (i.e., those used in the relief path definition) may not be directly reachable from the depot or other vertices. Consequently, additional vertices that are not used in the relief paths may need to be repaired just to access those that are in the relief paths. In the traditional scheduling and routing problems, there is no need to visit additional vertices (customers in the context of vehicle routing) only to reach another one. Lastly, in the SCSRP, the objective function is not linearly dependent on the time at which vertices are visited, but actually defined by a more complex measure based on the accessibility time. In most of the scheduling and routing problems, the objective function is not time-dependent, or it depends linearly on travel and/or service times. All the mentioned differences

prevent a trivial adaptation of existing approaches to the SCSRP and require the development of new tailored formulations and procedures in a BP environment, especially regarding the effective solution of pricing subproblems.

3. Problem Definition and Compact Formulation

Let $G = (\mathcal{V}, \mathcal{E})$ be an undirected graph representing a damaged network affected by extreme events, in which $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of edges. Edges represent highways or road segments in the transportation network. Let $\mathcal{V}^{\mathrm{r}} \subset \mathcal{V}$ be the set of damaged vertices, representing the damaged locations in the transportation network, $\mathcal{V}^{\mathrm{d}} \subset \mathcal{V}$ be the set of demand vertices, representing the affected communities in need of humanitarian assistance, and $\mathcal{V}^{\mathrm{u}} \subset \mathcal{V}$ be the set of intersection vertices. Intersection vertices represent the connection of multiple edges, i.e., the points in the transportation network where multiple road segments or highways intersect, but there is no demand or damage associated to them. There is one depot (vertex 0) that is a supply vertex to be connected to the demand vertices. An illustrative instance of the problem is presented in Figure 1(a) with five damaged vertices (red triangles), three demand vertices (blue circles), two intersection vertices (white circles), and twelve edges. It is important to note that G is not a complete graph. This means that some vertices may not be directly accessible from the depot or other vertices, as this is often the case in real-world transportation networks.



(a) Ilustrative example. (b) Optimal relief paths and crew scheduling. (c) Optimal crew routing.

Figure 1 **Illustrative example and its corresponding optimal relief paths, crew scheduling and crew routing decisions.**

There is a demand d_i for each vertex $i \in \mathcal{V}^{\mathtt{d}}$ and a predefined maximum distance $l_i^{\mathtt{d}}$ to reach vertex $i \in \mathcal{V}^{\mathtt{d}}$ from the depot. Tight $l_i^{\mathtt{d}}$ values help avoid selecting long relief paths unnecessarily. It is also possible to set $l_i^{\mathtt{d}}$ to a sufficiently large number ($l_i^{\mathtt{d}} = \infty$) to allow paths of any length to reach the demand vertices. A travel time τ_e and a length (distance) ℓ_e are defined for each edge $e \in \mathcal{E}$. There is a repair time δ_j spent by the crew to repair damaged vertex $j \in \mathcal{V}^{\mathtt{r}}$. Damaged vertices cannot be traversed unless they have been repaired.

The SCSRP helps supporting the following decisions: (i) how to connect the depot to the demand vertices (relief path decisions), (ii) to determine the schedule of the crew that will repair the damaged vertices (scheduling decisions), and (iii) to define the route that the crew will travel in order to repair the damaged vertices and return to the depot (routing decisions). Figure 1(b) shows the optimal crew scheduling and optimal relief paths for the illustrative example. For the sake of simplicity, this example assumes $\tau_e = \ell_e = 1, \forall e \in \mathcal{E}$. The problem relies on minimizing the *total inaccessibility time*, which is evaluated as the time that the demand vertices remain inaccessible from the depot. The demands give the weight or importance of each demand vertex. A demand vertex i becomes accessible when the damaged vertices used in the relief path from the depot to i are completely repaired. For instance, demand vertex 3 in Figure 1(b) becomes accessible after damaged vertices 6 and 7 are completely repaired. Multiple relief paths $0-i$ may be available to reach a given demand vertex i. In Figure 1(b), for example, we have one relief path $0-4$ represented by $p_4 : 0 \to 1 \to 7 \to 4$. An alternative relief path $0-4$ would be $p_4' : 0 \to 2 \to 9 \to 5 \to 10 \to 4$.

The damaged vertices must be repaired by a single crew that departs from the depot. The scheduling decisions define the order of the vertices repaired by the crew. Figure 1(b) illustrates a schedule K defined by the ordered set of vertices (0, 7, 10, 9, 6, 0). Since the crew must depart and return to the depot, we include vertex 0 at the beginning and at the end of the schedule. Damaged vertices used in the relief paths (e.g., 6, 7, 9) must be repaired by the crew, and therefore, they have to be in the schedule. The crew can also repair damaged vertices that are not used in the relief paths (e.g., vertex 10). In this case, it is better for the crew to repair vertex 10 instead of taking a longer path to arrive at vertex 9. Some damaged vertices may not be necessary in order to restore the accessibility of the demand vertices (e.g., vertex 8).

The routing decisions determine the route that the crew must travel to repair the vertices within the schedule K and return to the depot at the end, as illustrated in Figure 1(c). The crew route comprises crew paths defined as the sequence of vertices and edges used by the crew to travel between two consecutive damaged vertices in its schedule. A path p_{ij} used by the crew to travel from vertex i to vertex j is called crew path $i-j$. Vertices and edges can be traversed multiple times by the crew. Damaged vertices are repaired at the first time they are visited by the crew, thus repair times are only incurred once. In Figure 1(c), the route of the crew is composed by

crew paths $0-7, 7-10, 10-9, 9-6, 6-0$, where crew path $9-6$ is $p_{96}: 9 \to 5 \to 10 \to 4 \to 6$, for example. More than one path can be available for a crew to travel from one damaged vertex to the next in its schedule. However, feasible paths between damaged vertices must include only vertices that were not damaged (e.g., vertices 5, 4 in p_{96}) and/or that have been repaired (e.g., vertex 10 in p_{96}). Any feasible path composed of repaired damaged vertices and/or undamaged vertices can be used by the crews to return to the depot without affecting the value of the objective function. Finally, note that the crew paths that are feasible at a specific moment depends on which vertices remain damaged (have not been repaired) at that moment, which in turn depends on the scheduling decisions. Therefore, the available crew paths between damaged vertices change dynamically during the restoration according to the schedule.

Mathematical formulations for the SCSRP have been previously presented by Maya-Duque, Dolinskaya, and Sörensen (2016) and Moreno, Munari, and Alem (2019), Moreno et al. (2020). Table 1 shows the mathematical notation of the following base model used in previous studies, which is hereafter referred to as the SCSRP-CF.

$$(\textbf{SCSRP-CF}) \quad \min \sum_{i \in \mathcal{V}^{\mathrm{d}}} d_i \cdot Z_i^{\mathrm{d}}. \tag{1}$$

$$\text{s.t. } Z_i^{\mathrm{d}} \geq Z_j^{\mathrm{r}} + (V_{ji} - 1) \cdot M, \forall\ i \in \mathcal{V}^{\mathrm{d}},\ j \in \mathcal{V}^{\mathrm{r}}, \tag{2}$$

$$Z_j^{\mathrm{r}} \geq Z_i^{\mathrm{r}} + \sum_{e \in \mathcal{E}} P_{eij} \cdot \tau_e + \delta_j - (1 - X_{ij}) \cdot M, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}^{\mathrm{r}}, \tag{3}$$

$$Z_j^{\mathrm{r}} \geq Z_l^{\mathrm{r}} + (N_{lij} - 1) \cdot M, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}^{\mathrm{r}},\ l \in \mathcal{V}^{\mathrm{r}}, \tag{4}$$

$$\sum_{\substack{i \in \mathcal{V}_0^{\mathrm{r}}: \\ i \neq j}} X_{ij} = W_j, \forall\, j \in \mathcal{V}^{\mathrm{r}}, \tag{5}$$

$$\sum_{\substack{i \in \mathcal{V}_0^{\mathrm{r}}: \\ i \neq l}} X_{il} - \sum_{\substack{j \in \mathcal{V}_0^{\mathrm{r}}: \\ j \neq l}} X_{lj} = 0, \forall\ l \in \mathcal{V}_0^{\mathrm{r}}, \tag{6}$$

$$\sum_{j \in \mathcal{V}^{\mathrm{r}}} X_{0j} \leq 1, \tag{7}$$

$$\sum_{e \in \mathcal{E}_i} P_{eij} = X_{ij}, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}^{\mathrm{r}}, \tag{8}$$

$$\sum_{e \in \mathcal{E}_j} P_{eij} = X_{ij}, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}^{\mathrm{r}}, \tag{9}$$

$$\sum_{e \in \mathcal{E}_k} P_{eij} = 2N_{lij}, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}^{\mathrm{r}},\ l \in \mathcal{V} \setminus \{i, j\}, \tag{10}$$

$$W_l \geq \sum_{i \in \mathcal{V}_0^{\mathrm{r}}} N_{lij}, \forall\ l \in \mathcal{V}^{\mathrm{r}}, j \in \mathcal{V}^{\mathrm{r}}, \tag{11}$$

$$W_l \geq V_{li}, \forall\ l \in \mathcal{V}^{\mathrm{r}}, i \in \mathcal{V}^{\mathrm{d}}, \tag{12}$$

$$\sum_{e \in \mathcal{E}_0} Y_{ej} = 1, \forall\ j \in \mathcal{V}^{\mathrm{d}}, \tag{13}$$

$$\sum_{e\in\mathcal{E}_j} Y_{ej} = 1, \forall\ j \in \mathcal{V}^{\mathrm{d}}, \tag{14}$$

$$\sum_{e\in\mathcal{E}_k} Y_{ej} = 2V_{lj}, \forall\ j \in \mathcal{V}^{\mathrm{d}},\ l \in \mathcal{V}\setminus\{0,j\}, \tag{15}$$

$$\sum_{e\in\mathcal{E}} Y_{ej}\cdot \ell_e \le l_j^{\mathrm{d}}, \forall\ j \in \mathcal{V}^{\mathrm{d}}, \tag{16}$$

$$W_j, X_{ij} \in \{0,1\}, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}_0^{\mathrm{r}}, \tag{17}$$

$$N_{lij} \in \{0,1\}, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}_0^{\mathrm{r}},\ l \in \mathcal{V}, \tag{18}$$

$$P_{eij} \ge 0, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}_0^{\mathrm{r}},\ e \in \mathcal{E}, \tag{19}$$

$$V_{li} \in \{0,1\}, \forall\ i \in \mathcal{V}^{\mathrm{d}},\ l \in \mathcal{V}, \tag{20}$$

$$Y_{ei} \ge 0, \forall\ i \in \mathcal{V}^{\mathrm{d}},\ e \in \mathcal{E}, \tag{21}$$

$$Z_i^{\mathrm{d}} \ge 0, \forall\ i \in \mathcal{V}^{\mathrm{d}}, \tag{22}$$

$$Z_i^{\mathrm{r}} \ge 0, \forall\ i \in \mathcal{V}_0^{\mathrm{r}}. \tag{23}$$

The objective function (1) minimizes the inaccessibility time of the demand vertices. Constraints (2) define the accessibility time of the demand vertices. Constraints (3) define the restoration time of the damaged vertices and avoid subtours. Constraints (4) ensure that a vertex l in the path from vertex i to vertex j must be repaired before vertex j. Constraints (5)–(7) define the schedule of the crew for the damaged vertices that must be repaired. Constraints (8) state that damaged vertices

Table 1 Mathematical notation of the SCSRP formulations.

Sets	
$\mathcal{V}$	Set of vertices.
$\mathcal{V}^{\mathrm{d}} \subset \mathcal{V}$	Set of demand vertices.
$\mathcal{V}^{\mathrm{r}} \subset \mathcal{V}$	Set of damaged vertices.
$\mathcal{V}_0^{\mathrm{r}}$	Set of damaged vertices including the source vertex 0 ($\mathcal{V}_0^{\mathrm{r}} = \mathcal{V}^{\mathrm{r}} \cup \{0\}$).
$\mathcal{E}$	Set of edges.
$\mathcal{E}_i \subseteq \mathcal{E}$	Set of edges incident to vertex $i \in \mathcal{V}$.
Parameters	
d_i	Demand of vertex $i \in \mathcal{V}^{\mathrm{d}}$.
δ_i	Repair time of vertex $i \in \mathcal{V}^{\mathrm{r}}$.
τ_e	Travel time on edge $e \in \mathcal{E}$.
ℓ_e	Length (distance) of edge $e \in \mathcal{E}$.
l_i^{d}	Maximum distance allowed between vertex 0 and demand vertex $i \in \mathcal{V}^{\mathrm{d}}$.
M	sufficiently large number.
Decision variables	
$W_i \in \{0,1\}$	1 if, and only if (iff), vertex $i \in \mathcal{V}^{\mathrm{r}}$ is repaired.
$X_{ij} \in \{0,1\}$	1 iff vertex $j \in \mathcal{V}_0^{\mathrm{r}}$ is repaired immediately after vertex $i \in \mathcal{V}_0^{\mathrm{r}}$.
$P_{eij} \in \{0,1\}$	1 iff edge $e \in \mathcal{E}$ is used in the crew path from vertex $i \in \mathcal{V}_0^{\mathrm{r}}$ to vertex $j \in \mathcal{V}_0^{\mathrm{r}}$.
$N_{lij} \in \{0,1\}$	1 iff vertex $l \in \mathcal{V}$ is used in the crew path from vertex $i \in \mathcal{V}_0^{\mathrm{r}}$ to vertex $j \in \mathcal{V}_0^{\mathrm{r}}$.
$Y_{ej} \in \{0,1\}$	1 iff edge $e \in E$ is used in the relief path from vertex 0 to vertex $j \in \mathcal{V}^{\mathrm{d}}$.
$V_{lj} \in \{0,1\}$	1 iff vertex $l \in \mathcal{V}$ is used in the relief path from vertex 0 to vertex $j \in \mathcal{V}^{\mathrm{d}}$.
$Z_i^{\mathrm{r}} \ge 0$	Restoration time of damaged vertex $i \in \mathcal{V}_0^{\mathrm{r}}$.
$Z_i^{\mathrm{d}} \ge 0$	Accessibility time of demand vertex $i \in \mathcal{V}^{\mathrm{d}}$.

used in relief paths must be repaired, while constraints (9) state that damaged vertices used in crew paths must be repaired. Constraints (10)–(12) ensure the flow conservation in the crew path $i-j$. Constraints (13)–(15) ensure the flow conservation in the relief path $0-j$. Constraints (16) prohibit the use of relief paths with a distance greater than the maximum distance allowed between the depot and the demand vertices. Finally, constraints (17)–(23) impose the domain of the decision variables. Variables $Y_{ej}(P_{eij})$ do not need to be defined as binary variables because they naturally assume binary values if variables $V_{kj}(N_{kij})$ are binaries (Moreno, Munari, and Alem 2019).

4. New Formulations for the SCSRP

In this section, we derive three new formulations for the SCSRP. The first formulation is based on the explicit enumeration of the crew scheduling and crew routing decisions, while the relief path decisions are considered as in model SCSRP-CF. The second formulation is based on the explicit enumeration of the relief path, crew scheduling, and crew routing decisions. Finally, the third formulation is derived from the second model by inserting additional relief path decision variables and constraints defined as in model SCSRP-CF.

4.1. Route-based Formulation

This formulation assumes that variables and constraints related to the design of schedules and routes are not explicitly considered in the model. Instead, all feasible schedules and routes are enumerated. The additional parameters and variables used in the route-based formulation are defined as follows.

Additional sets

$\mathcal{P}$ Set of all feasible crew schedules.
$\mathcal{R}$ Set of all feasible routes.
$\mathcal{R}_p$ Set of all feasible routes given a schedule $p \in \mathcal{P}$.

Additional parameters

z_{jr} Restoration time of damaged vertex $j \in \mathcal{V}^{\mathrm{r}}$ repaired using route $r \in \mathcal{R}$.
q_{jp} Binary parameter that is equal to 1 iff damaged vertex $j \in \mathcal{V}^{\mathrm{r}}$ is repaired in schedule $p \in \mathcal{P}$.

Additional decision variable

$\lambda_{pr} \in \{0,1\}$ 1 iff route $r \in \mathcal{R}_p$ associated to schedule $p \in \mathcal{P}$ is selected in the optimal solution.

The route-based formulation is cast as follows:

$$(\mathbf{MP1}) \quad \min \sum_{i \in \mathcal{V}^{\mathrm{d}}} d_i Z_i^{\mathrm{d}}, \tag{24}$$

$$\text{s.t. Constraints (2), (13)-(16), (20)-(23),}$$

$$\sum_{p \in \mathcal{P}} \sum_{r \in \mathcal{R}_p} q_{jp} \lambda_{pr} \geq V_{ji}, \forall\, i \in \mathcal{V}^{\mathrm{d}},\, j \in \mathcal{V}^{\mathrm{r}}, \tag{25}$$

$$Z_j^{\mathrm{r}} \geq \sum_{p \in \mathcal{P}} \sum_{r \in \mathcal{R}_p} z_{jr} \lambda_{pr}, \forall\ j \in \mathcal{V}^{\mathrm{r}}, \tag{26}$$

$$\sum_{p \in \mathcal{P}} \sum_{r \in \mathcal{R}_p} \lambda_{pr} \leq 1, \tag{27}$$

$$\lambda_{pr} \in \{0,1\}, \forall\ p \in \mathcal{P},\ r \in \mathcal{R}_p. \tag{28}$$

Constraints (25) ensure the selection of a schedule p that repairs damaged vertex j if this vertex is used in some relief path (i.e., $V_{ji} = 1$). Constraints (26) define the restoration time of the damaged vertices. Constraint (27) guarantees the selection of at most one pair route-schedule. Finally, constraints (28) define the domain of the decision variables related to the selection of crew schedules and routes.

4.2. Route and Path-based Formulation

In this formulation, all the possible schedules, routes and relief paths are explicitly enumerated. The additional parameters and variables to define this formulation are as follows.

Additional sets
$\mathcal{L}$ Set of all feasible relief paths.
$\mathcal{L}_i \subseteq \mathcal{L}$ Set of all relief paths to reach demand vertex $i \in \mathcal{V}^{\mathrm{d}}$.

Additional parameter
g_{jf} Binary parameter that is equal to 1 iff damaged vertex $j \in \mathcal{V}^{\mathrm{r}}$ is used in relief path $f \in \mathcal{L}$.

Additional decision variable
$\theta_f \in \{0,1\}$ 1 iff the relief path $f \in \mathcal{L}$ is selected in the optimal solution.

The route and path-based formulation is stated as follows:

$$\begin{aligned}
(\mathbf{MP2}) \quad & \min \sum_{i \in \mathcal{V}^{\mathrm{d}}} d_i Z_i^{\mathrm{d}}, && (29)\\
& \text{s.t. Constraints (22), (23), (26)-(28),} \\
& Z_i^{\mathrm{d}} \geq Z_j^{\mathrm{r}} - M(1 - \sum_{f \in \mathcal{L}_i} g_{jf}\theta_f),\ \forall\ i \in \mathcal{V}^{\mathrm{d}},\ j \in \mathcal{V}^{\mathrm{r}}, && (30)\\
& \sum_{p \in \mathcal{P}} \sum_{r \in \mathcal{R}_p} q_{jp}\lambda_{pr} \geq \sum_{f \in \mathcal{L}_i} g_{jf}\theta_f,\ \forall\ j \in \mathcal{V}^{\mathrm{r}},\ i \in \mathcal{V}^{\mathrm{d}}, && (31)\\
& \sum_{f \in \mathcal{L}_i} \theta_f = 1,\ \forall\ i \in \mathcal{V}^{\mathrm{d}}, && (32)\\
& \theta_f \in \{0,1\},\ \forall\ f \in \mathcal{L}. && (33)
\end{aligned}$$

The accessibility time is defined by constraints (30). Constraints (31) force the selection of a pair route-schedule that repairs damaged vertex j if this vertex is used in some relief path. Constraints (32) guarantee the selection of exactly one relief path for each demand vertex i. Finally, constraints (33) define the domain of the decision variables related to relief path selection.

4.3. Route and Path-based Formulation with Explicit Relief Path Variables

This third formulation builds upon the second formulation by explicitly including relief path variables and constraints as in the original compact formulation SCSRP-CF. The so-called route and

path-based formulation with explicit relief path variables is stated as follows:

$$\textbf{(MP3)} \quad \min \sum_{i \in \mathcal{V}^{\mathrm{d}}} d_i Z_i^{\mathrm{d}}, \tag{34}$$

$$\text{s.t. Constraints (2), (13)-(16), (20)-(23), (26)-(28), (30)-(33),}$$

$$V_{ji} \geq \sum_{f \in \mathcal{L}_i} p_{jf} \theta_f, \forall\ j \in \mathcal{V}^{\mathrm{r}}, i \in \mathcal{V}^{\mathrm{d}}. \tag{35}$$

Constraints (35) link the relief path variables V_{ji} of the compact formulation with the relief path variables θ_f of the route and path-based formulation, as constraints (2), (13)-(16) are included and they are related to V_{ji}.

5. Branch-and-Price Algorithms

In practical settings, the complete enumeration of the sets of feasible schedules, routes and relief paths is impractical for large-scale instances. Hence, we develop BP algorithms to solve the novel formulations presented in Section 4. BP is a branch-and-bound scheme in which we resort to the CG technique at each branching node of the branching tree. CG is an iterative procedure that solves the a sequence of restricted master problems (RMPs), defined as the linear programming (LP) relaxation of the node and considering a subset of schedules, routes, and relief paths. At each branching node, we initialize the first RMP with a subset of variables related to given columns (schedules, routes, and relief paths), and the remaining RMPs have additional variables/columns that are generated based on their reduced costs, by means of pricing subproblems. If no column with negative reduced costs can be generated by these subproblems, the optimal solution of the last solved RMP is also optimal for the node under evaluation.

In what follows, we introduce the restricted master problems and the pricing subproblems for the BP approaches related to each novel formulation (Sections 5.1 and 5.2). We also introduce the labeling algorithms tailored to solve the pricing subproblems (Sections 5.4 and 5.5). Then, we define the branching rules used in the BP to enforce integer solutions (Section 5.6). Finally, we present some enhancement strategies to speed up the proposed BP methods (Section 5.7).

5.1. Route-based Restricted Master Problem (RMP1)

We define the restricted master problem RMP1 as the LP relaxation of the problem MP1 over subsets of schedules $\tilde{\mathcal{R}} \subseteq \mathcal{R}$ and routes $\tilde{\mathcal{P}} \subseteq \mathcal{P}$. Let $\mu_{ij}^1 \geq 0, \upsilon_j \geq 0$, and $\nu \leq 0$ be the dual solution associated with constraints (25), (26), and (27), respectively. Let $\mu_j = \sum_{i \in \mathcal{V}^{\mathrm{d}}} \mu_{ij}^1$. The reduced cost $\bar{c}_{pr}$ of the variable λ_{pr} is given by

$$\bar{c}_{pr} = \sum_{j \in \mathcal{V}^{\mathrm{r}}} \upsilon_j z_{jr} - \sum_{j \in \mathcal{V}^{\mathrm{r}}} \mu_j q_{jp} - \nu.$$

Hence, the pricing subproblem to generate the schedules and routes in $\tilde{\mathcal{P}}$ and $\tilde{\mathcal{R}}$ for RMP1 is defined as follows:

$$(\textbf{SP1}) \quad \min \sum_{j \in \mathcal{V}^{\mathrm{r}}} \upsilon_j Z_j^{\mathrm{r}} - \sum_{j \in \mathcal{V}^{\mathrm{r}}} \mu_j W_j - \nu, \tag{36}$$
$$\text{s.t. Constraints (3)-(11), (17)-(19), (23).}$$

This subproblem is a variant of the shortest path problem (SPP) with negative costs and cycles that additionally involves crew routing and scheduling decisions. It is closely related to the SPP with resource constraints (SPPRC) that commonly appears in BP approaches for VRP variants (Feillet 2010, Irnich and Desaulniers 2005, Ioachim et al. 1998). However, different from the traditional SPPRC, vertices and edges may be traversed multiple times; service times change dynamically over time; and the shortest path between two vertices also change over time. Additionally, its objective function does not depend linearly on the elapsed time, but on the time each vertex is repaired.

In addition to SP1, we define a relaxed SP1 subproblem (rSP1), in which the constraints and variables defining the crew paths (N_{lij}, P_{eij}) are eliminated and a lower bound for the travel time of the crew between two consecutive damaged vertices $i-j$ in its schedule is defined instead. This relaxed subproblem is defined as follows:

$$(\textbf{rSP1}) \quad \min \sum_{j \in \mathcal{V}^{\mathrm{r}}} \upsilon_j Z_j^{\mathrm{r}} - \sum_{j \in \mathcal{V}^{\mathrm{r}}} \mu_j W_j - \nu, \tag{37}$$
$$\text{s.t. Constraints (5)-(7), (17), (23),}$$
$$Z_j^{\mathrm{r}} \geq Z_i^{\mathrm{r}} + \tilde{t}_{ij} + \delta_j - (1 - X_{ij}) \cdot M, \forall\ i \in \mathcal{V}_0^{\mathrm{r}},\ j \in \mathcal{V}^{\mathrm{r}}, \tag{38}$$

where $\tilde{t}_{ij}$ is defined as the shortest travel time of the crew between vertices i and j over the original graph G, and constraints (38) define the restoration time of the damaged vertices. Hence, the main difference to SP1 is that the shortest path between any two damaged vertices is fixed and thus does not change over time (but its cost is underestimated). The purpose of this relaxation is to reduce computation times. Although rSP1 does not provide a solution for the original pricing problem SP1, it is possible to efficiently derive a solution for SP1 given a solution for rSP1, using the algorithm presented in Appendix A.

5.2. Route and Path-based Restricted Master Problem (RMP2)

We define the restricted master problem RMP2 as the LP relaxation of the problem MP2 over subsets of schedules $\tilde{\mathcal{R}} \subseteq \mathcal{R}$, routes $\tilde{\mathcal{P}} \subseteq \mathcal{P}$, and relief paths $\tilde{\mathcal{L}} \subseteq \mathcal{L}$. Let $\omega_{ij} \geq 0, \mu_{ij}^2 \geq 0$, and η_i (free) be the dual solution vectors associated with constraints (30), (31), and (32), respectively. The reduced cost $\bar{s}_{if}$ of a column f associated with demand vertex i is given by

$$\bar{s}_{if} = \sum_{j \in \mathcal{V}^{\mathrm{r}}} M\omega_{ij} g_{jf} + \sum_{j \in \mathcal{V}^{\mathrm{r}}} \mu_{ij}^2 g_{jf} - \eta_i.$$

The pricing subproblem to generate the relief paths associated with demand vertex i ($\tilde{\mathcal{L}}_i$) in RMP2 is then defined as follows:

$$(\textbf{SP2}) \quad \min \sum_{j \in \mathcal{V}^{\mathrm{r}}} M\omega_{ij} V_{ji} + \sum_{j \in \mathcal{V}^{\mathrm{r}}} \mu^2_{ij} V_{ji} - \eta_i, \tag{39}$$
$$\text{s.t. Constraints (13)-(16), (20), (21).}$$

This subproblem is an SPPRC in which the only resource is related to the distance between vertex 0 and demand vertex $i \in \mathcal{V}^{\mathrm{d}}$ being limited by l^{d}_i. Notably, we also need a pricing subproblem to generate the schedules and routes in RMP2. This subproblem is defined similarly to SP1, using $\mu_j = \sum_{i \in \mathcal{V}^{\mathrm{d}}} \mu^2_{ij}$ instead of $\mu_j = \sum_{i \in \mathcal{V}^{\mathrm{d}}} \mu^1_{ij}$.

5.3. Route and Path-based Restricted Master Problem with Explicit Relief Path Variables (RMP3)

The restricted master problem RMP3 is the LP relaxation of the problem MP3 over the subsets of schedules $\tilde{\mathcal{R}} \subseteq \mathcal{R}$, routes $\tilde{\mathcal{P}} \subseteq \mathcal{P}$, and relief paths $\tilde{\mathcal{L}} \subseteq \mathcal{L}$. Let $\rho_{ij} \geq 0$ be a dual solution associated with constraints (35). The reduced cost $\bar{s}_{if}$ of a column f associated to demand vertex i is given by

$$\bar{s}_{if} = \sum_{j \in \mathcal{V}^{\mathrm{r}}} M\omega_{ij} g_{jf} + \sum_{j \in \mathcal{V}^{\mathrm{r}}} \mu^2_{ij} g_{jf} + \sum_{j \in \mathcal{V}^{\mathrm{r}}} \rho_{ij} g_{jf} - \eta_i.$$

Then, the pricing subproblem to generate the relief paths associated with demand vertex i ($\tilde{\mathcal{L}}_i$) in RMP3 is defined as follows:

$$(\textbf{SP3}) \quad \min \sum_{j \in \mathcal{V}^{\mathrm{r}}} M\omega_{ij} V_{ji} + \sum_{j \in \mathcal{V}^{\mathrm{r}}} \mu^2_{ij} V_{ji} + \sum_{j \in \mathcal{V}^{\mathrm{r}}} \rho_{ij} V_{ji} - \eta_i, \tag{40}$$
$$\text{s.t. Constraints (13)-(16), (20), (21).}$$

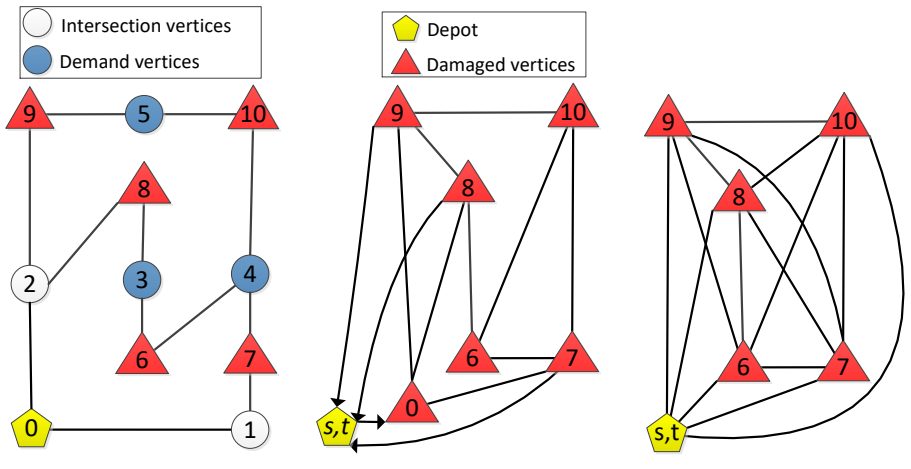
Note that this subproblem is the same as defined in (39), except for one additional term in the objective function. Finally, the pricing subproblem to generate crew schedules and routes in the RMP3 is the same subproblem defined for the RMP2.

5.4. Labeling Algorithms to Generate the Crew Schedules and Routes

Labeling algorithms are popular techniques for solving variants of the SPPRC that appear as pricing subproblems in BP algorithms for the VRP and its extensions (Costa, Contardo, and Desaulniers 2019, Alvarez and Munari 2017). Even though our subproblem SP1 can be cast as a variant of the SPPRC, adapting the existing labeling algorithms to efficiently solve this subproblem is not straightforward. Indeed, the objective function of SP1 depends on the restoration time of the damaged vertices, which in turn depends on the total travel time spent in the partial route traveled between damaged vertices as well as on the restoration time of the previously visited damaged vertices. Additionally, vertices can be visited multiple times in the SP1, while the dual solution and

the service time should only be accounted for during the first visit to the vertex. None of these characteristics are fully considered in the labeling algorithms available in the existing literature. Therefore, in the following subsections, we present labeling algorithms that are the first suited to solving subproblems SP1 and rSP1.

5.4.1. Graph construction The first step to solve SP1 using our labeling algorithm is to transform the original graph G into a graph G' that considers only the damaged vertices plus two additional vertices. The two additional vertices represent the depot as a source and sink vertex (vertices s and t), which are defined as damaged vertices with repair time $\delta_0 = 0$ and can be visited multiple times. Figure 2(b) shows an example of a graph G' generated from graph G in Figure 2(a). The travel time t_{ij} from vertex i to vertex j in graph G' is set as the shortest travel time of the crew between vertices i and j in the original graph G, but without using damaged vertices. Therefore, parameter t_{ij} is computed by solving the shortest path problem over a graph in which damaged vertices $l \in \mathcal{V}_0^r : l \neq i, l \neq j$ have been removed from the original graph G. The removal of the damaged vertices may result in multiple unconnected graph components in the graph. Consequently, there may be no paths between some pairs of vertices $i - j$ without using at least one damaged vertex. In this case, we do not consider an edge connecting vertices i and j. For instance, there is no edge connecting vertices 0 and 8 in Figure 2(b). In addition to graph G', we build a graph G'' that is used to solve the relaxed subproblem rSP1. An example of a graph G'' is presented in Figure 2(c). In graph G'', we allow the connection between all the damaged vertices, assuming a travel cost $\tilde{t}_{ij}$ that is equal to the shortest travel time of the crew between vertices i and j over the original graph G, considering that all vertices are undamaged/repaired.



(a) Graph G. (b) Graph G'. (c) Graph G''.

Figure 2 **Example of graphs G, G', and G''.**

5.4.2. Label extension The proposed labeling algorithms aim to determine a schedule and its corresponding route on graph G' that leads to a column with a negative reduced cost. The route is required to start and end at the depot (vertices s and t) and may visit other vertices in the network multiple times. Each partial route from depot s to a given vertex $i \in \mathcal{V}_0^{\mathrm{r}} \cup \{t\}$ is represented by a label $L_i = (C_i, V_i, R_i, w_i)$, where C_i is the total reduced cost of the route, V_i is the set containing all the vertices visited in the route, R_i is the total time consumption of the route, and w_i indicates the last damaged vertex repaired by the crew. To start the algorithm, a label is initially defined at the depot s with values $C_s = -\nu$, $V_s = \{s\}$, $R_s = 0$, and $w_s = \varnothing$. Subsequently, the algorithm iteratively extends each label L_i in a given vertex i to each vertex j for which an edge (i, j) exists, generating a new label L_j with the following values:

$$C_j = \begin{cases} C_i, & \text{if } j \in V_i, \\ C_i + \upsilon_j R_j - \mu_j, & \text{otherwise;} \end{cases} \qquad V_j = \begin{cases} V_i, & \text{if } j \in V_i, \\ V_j \cup \{j\}, & \text{otherwise;} \end{cases}$$
$$R_j = \begin{cases} R_i + t_{ij}, & \text{if } j \in V_i, \\ R_i + t_{ij} + \delta_j, & \text{otherwise;} \end{cases} \qquad w_j = \begin{cases} w_i, & \text{if } j \in V_i, \\ j, & \text{otherwise.} \end{cases}$$

Similarly, to find a pair schedule-route over graph G'' that leads to a column with a negative reduced cost, each partial route from depot s to a given vertex $i \in \mathcal{V}^{\mathrm{r}} \cup \{t\}$ is represented by a label $L_i = (C_i, V_i, R_i, w_i)$. The algorithm iteratively extends each label L_i in a given vertex i to each *non-visited* vertex j, which generates a new label L_j with the following values:

$$C_j = C_i + \upsilon_j R_j - \mu_j;\ V_j = V_i \cup \{j\};\ R_j = R_i + \tilde{t}_{ij} + \delta_j;\ w_j = j.$$

Note that the labeling algorithm for the SP1 allows vertices to be visited more than once, while in the algorithm for the rSP1, vertices can be visited at most once. As a consequence, the number of possible states generated in the labeling algorithm for the rSP1 is significantly smaller than in the algorithm for the SP1. Additionally, stronger dominance rules can be established for the labeling algorithm over graph G'', as shown in Section 5.4.3.

5.4.3. Dominance rule In labeling algorithms, dominance rules help to eliminate partial routes that are redundant or do not lead to optimal solutions, thus significantly reducing the number of required extensions. In what follows, we state dominance rules for the labeling algorithms designed to solve SP1 and rSP1.

Proposition 1 *(Dominance rule for the labeling algorithms). Let* $L_i^1 = (C_i^1, V_i^1, R_i^1, w_i^1)$ *and* $L_i^2 = (C_i^2, V_i^2, R_i^2, w_i^2)$ *be two labels associated with partial routes ending at vertex* i*. Let* $\phi_j = R_i^2 + \tilde{t}_{ij} + \delta_j$*, which represents the earliest possible time at which damaged vertex* $j \in \mathcal{V}^{\mathrm{r}} \setminus V_i^2$ *can be repaired in any extension from label* L_i^2*.*
L_i^1 *dominates* L_i^2 *in the labeling algorithm to solve SP1 if the following conditions hold:*

(i) $C_i^1 \leq C_i^2$;

(ii) $(V_i^1 = V_i^2)$
or $(V_i^1 \subset V_i^2$ **and** $\sum_{j \in V_i^2 \setminus V_i^1} \delta_j \leq R_i^2 - R_i^1$ **and** $\upsilon_j = 0, \forall j \in V_i^2 \setminus V_i^1)$
or $(V_i^2 \subset V_i^1$ **and** $\mu_j - \phi_j \upsilon_j \leq 0, \forall j \in V_i^1 \setminus V_i^2)$;

(iii) $R_i^1 \leq R_i^2$;

L_i^1 *dominates* L_i^2 *in the labeling algorithm to solve rSP1 if the following conditions hold:*

(i) $C_i^1 \leq C_i^2$;

(ii) $V_i^1 \subseteq V_i^2$ **or** $(V_i^2 \subset V_i^1$ **and** $\mu_j - \phi_j \upsilon_j < 0, \ \forall j \in V_i^1 \setminus V_i^2)$;

(iii) $R_i^1 \leq R_i^2$.

Proposition 1 is proved in Appendix B.1. Condition (ii) could be set only as $V_i^1 = V_i^2$. However, this would be a weaker dominance rule.

5.4.4. Lower bound for the reduced cost The number of extensions required to find an optimal solution in the labeling algorithms can be further reduced by eliminating partial paths that do not lead to routes with negative reduced cost. Given a partial path represented by label L_i, we use a lower bound to verify whether the reduced costs of the routes that can be generated from L_i are greater than or equal 0.

Proposition 2 *(Lower bound for the reduced cost in the labeling algorithms). Assume a label* $L_i = (C_i, V_i, R_i, w_i)$*, with* $C_i \geq 0$ *and* $w_i = i$*. Let* $S \subseteq \mathcal{V}^{\mathrm{r}} \setminus V_i$ *be the set of all the non-visited damaged vertices* j *with* $\mu_j > 0$*. Let* $\phi_j = R_i + \tilde{t}_{ij} + \delta_j$*, representing the earliest possible time at which damaged vertex* $j \in S$ *can be repaired in any extension from label* L_i*. Therefore, we define* $\texttt{prize}_j$ *as the estimated potential reduction in cost* C_i *if damaged vertex* j *is repaired by the crew after damaged vertex* i*, calculated as follows:*

$$\texttt{prize}_j = \max\left\{\mu_j - \phi_j \upsilon_j, 0\right\}.$$

Let N *be a lower bound for the number of damaged vertices that should be repaired after vertex* i *to find a route with a reduced cost smaller than 0. The value of* N *can be computed based on the prizes. Let* $S_p \subseteq S$ *be a set containing the* p *vertices with the highest prizes* $\texttt{prize}_j$*. Then,* N *is equal to the smallest* p *such that* $\sum_{j \in S_p} \texttt{prize}_j > C_i$*. Therefore, we define* $\texttt{penalty}_n$*, with* $n \leq N$*, as the estimated penalty associated with the* n*th damaged vertex repaired after damaged vertex* i*, calculated as follows:*

$$\texttt{penalty}_n = \rho_n^{\texttt{rank}} \upsilon_n^{\texttt{rank}},$$

where, $\rho_j = \min_{l \in S}\{\tilde{t}_{lj}\} + \delta_j, \forall j \in S : l \neq j$*;* $\rho_n^{\texttt{rank}}$ *is the sum of the* $n-1$ *smallest* ρ_j *values such that* $j \in \mathcal{V}^{\mathrm{r}} \setminus V_i$ *for* $n > 1$ *and* $\rho_n^{\texttt{rank}} = 0$ *for* $n = 1$*; and* $\upsilon_n^{\texttt{rank}}$ *is the* $(|S| - N + n - 1)$*th highest* υ_j *value such that* $j \in S$ *for* $n > 1$ *and* $\upsilon_n^{\texttt{rank}} = 0$ *for* $n = 1$.

A lower bound C_i^{LB} *for the reduced cost of any route derived from* L_i *in the labeling algorithm is given by:*

$$C_i^{\text{LB}} = \begin{cases} C_i, & \text{if } \sum_{j\in S} \texttt{prize}_j \le C_i \text{ and } \sum_{j\in S} \texttt{prize}_j = 0, \\ 0, & \text{if } \sum_{j\in S} \texttt{prize}_j \le C_i \text{ and } \sum_{j\in S} \texttt{prize}_j > 0, \\ C_i, & \text{if } \sum_{j\in S} \texttt{prize}_j > C_i \text{ and } \sum_{j\in S} \texttt{prize}_j \le \sum_{n=1}^{N} \texttt{penalty}_n, \\ C_i - \sum_{j\in S} \texttt{prize}_j + \sum_{n=1}^{N} \texttt{penalty}_n, & \text{if } \sum_{j\in S} \texttt{prize}_j > C_i \text{ and } \sum_{j\in S} \texttt{prize}_j > \sum_{n=1}^{N} \texttt{penalty}_n. \end{cases}$$

Proposition 2 is proved in Appendix B.2.

5.4.5. Unreachable vertices We define a set of unreachable vertices to strengthen the dominance rule, following the approach proposed in related studies (Feillet et al. 2004). These unreachable vertices are determined based on the reduced cost of the extensions that can be generated from a given partial route L_i. We define $\phi_j = R_i + \tilde{t}_{ij} + \delta_j$, $\bar{V}_i = \mathcal{V}^{\text{r}} \setminus V_i$, and $\mathcal{L}_i$ as the set of routes that can be generated from L_i. For the labeling algorithm to solve SP1, if $\mu_j - \phi_j \upsilon_j \le 0,\ \forall j \in \bar{V}_i$, then there exists no route $L_j \in \mathcal{L}_i$ such that $C_j < C_i$, and consequently, all vertices in set $\bar{V}_i$ are defined as unreachable. Regarding the labeling algorithm to solve rSP1, we introduce $\mathcal{L}_i'$ as the set of routes that visit vertex j and can be generated from L_i, and $\mathcal{L}_i$ as the set of routes that do not visit vertex j and can be generated from L_i. Additionally, C^l represents the reduced cost of the partial route l. If $\mu_j - \phi_j \upsilon_j \le 0$, then it follows that $\min_{l\in\mathcal{L}_i} C^l \le \min_{l\in\mathcal{L}_i'} C^l$, and vertex j is considered unreachable.

5.5. Labeling Algorithm to Generate the Relief Paths

We resort to a labeling algorithm to solve subproblem SP2 as well. In this case, the algorithm is similar to that used in the literature to solve the SPPRC on traditional VRP variants. We execute this algorithm for each demand vertex $i \in \mathcal{V}^{\text{d}}$ independently, over an auxiliary graph G^i. This graph considers the damaged vertices and two additional vertices only, representing the depot and demand vertex i. An example of graph G^5 is presented in Figure 3. The travel time t_{jk} between two different vertices j and k in graph G^i is set as the shortest distance between vertices j and k in the original graph G, but without using damaged vertices, i.e., removing the damaged vertices from the original graph G. Therefore, an edge connecting vertex j to vertex k can exist in graph G^i if there is a path connecting vertex j to vertex k without using damaged vertices in the original graph G.

A time window $[0, l_i^{\text{d}}]$ is set for demand vertex i in graph G^i, thus forcing the selection of paths with distance less than the maximum allowed distance between vertex 0 and demand vertex $i \in \mathcal{V}^{\text{d}}$, as defined in constraints (16) of model SCSRP-CF. Given graph G^i, the label extension, dominance rules, lower bounds, and unreachable vertex criteria are similar to the used to solve the SPPRC derived from the classic VRP with time windows (Munari and Morabito 2018, Costa, Contardo, and Desaulniers 2019).

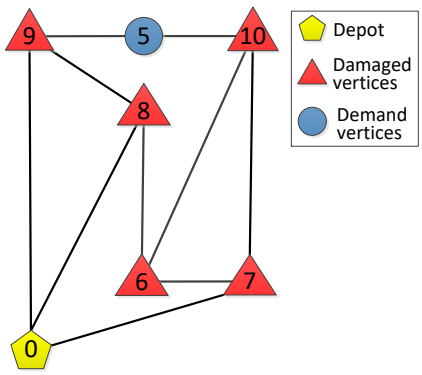


Figure 3 **An example of graph G^5.**

5.6. Branching

At each node of the BP tree, we achieve an optimal solution of the RMP using the CG procedure. If this optimal solution is integral, we have two cases: either its objective value is less than the current upper bound, thus we update the upper bound and define this solution as the incumbent; or its objective value is not less than the current upper bound, thus the associated node is pruned. If the optimal solution is not integral, branching must take place. After much experimentation, we implement a scheme with three different branching strategies. To explain our branching scheme, we introduce the following notation. Let $\bar{\lambda}_{pr}$ and $\bar{\theta}_f$ represent the optimal solution vectors obtained from the last solved RMP in the node. Additionally, we have the optimal solution vector $\bar{V}_{jl}$ for formulation RMP1, and define the vector $\bar{V}_{jl} = \sum_{f\in\mathcal{L}_l} g_{jf}\bar{\theta}_f$ for RMP2 and RMP3. Let $\bar{X}_{ij} = \sum_{p\in\tilde{\mathcal{P}}}\sum_{r\in\tilde{\mathcal{R}}_p} b_{ijr}\bar{\lambda}_{pr}$, where $b_{ijr} = 1$ if vertex j is repaired immediately after vertex i in route r, and $b_{ijr} = 0$ otherwise; $z_j^{\text{avg}} = \sum_{p\in\tilde{\mathcal{P}}}\sum_{r\in\tilde{\mathcal{R}}_p} z_{jr}\bar{\lambda}_{pr}$; and $z_j^{\max} = \max_{p\in\tilde{\mathcal{P}},r\in\tilde{\mathcal{R}}_p:\bar{\lambda}_{pr}>0}\{z_{jr}\}$.

Based on the defined notation, three branching strategies are defined: on the scheduling decisions, on the restoration time of damaged vertices, and on the vertices used in relief paths. We define a metric h to prioritize the selection of branching decisions that may have a higher impact in the child nodes. When branching takes place, we use the branching decision that has the higher metric h. After branching, the selection of the next node to explore in the tree is based on the the best-first strategy. The branching strategies as well as the definition of the metric h is presented in the following subsections. The proposed metric prioritizes the branching on the vertices used in the relief paths.

5.6.1. Branching on the scheduling decisions Assume $\bar{X}_{ij}$ is not integer for a given pair of nodes $i, j \in \mathcal{V}_0^{\text{r}}$. In child node 1, we set $X_{ij} = 0$ by forbidding the restoration of damaged vertex j after the restoration of damaged vertex i in the labeling algorithm to solve SP1; removing edge (i, j) from graph G'' in the labeling algorithm to solve rSP1; and dropping all columns corresponding to schedules that do not satisfy $\bar{X}_{ij} = 0$ in the RMPs. In child node 2, we set $X_{ij} = 1$ by adding

the term $(2 - W_i - W_j)\max\{0, 2(\sum_{j\in\mathcal{V}^{\mathrm{r}}} \mu_j - \nu)\}$ to the objective functions of SP1 and rSP1 to force the restoration of damaged vertices i and j; forbidding the restoration of vertex j if vertex i was not the last repaired vertex in the labeling algorithms to solve SP1 and rSP1; forbidding the restoration of any vertex $l \neq j$ if the last repaired vertex is vertex i in the labeling algorithms to solve SP1 and rSP1; and dropping all columns corresponding to schedules that do not satisfy $\bar{X}_{ij} = 1$ in RMPs. For branching on the scheduling decisions, the metric h is defined as $h = \frac{1}{2}h_1 + \frac{1}{2}h_2$, where $h_1 = 2(0.5 - |1 - \bar{X}_{ij} - 0.5|)$, $h_2 = \frac{t_{ij}}{2t^*} + \frac{\delta_j}{2\delta^*}$, $t^* = \max_{i'\in\mathcal{V}_0^{\mathrm{r}}, j'\in\mathcal{V}^{\mathrm{r}}}\{t_{i'j'}\}$, $\delta^* = \max_{j'\in\mathcal{V}^{\mathrm{r}}}\{\delta_{j'}\}$.

5.6.2. Branching on the restoration time of damaged vertices Given $z_j^{\mathrm{avg}} \neq z_j^{\max}$ for some $j \in \mathcal{V}^{\mathrm{r}}$, we proceed as follows. In child node 1, we set a time window for the restoration time of vertex j such that $Z_j^{\mathrm{r}} \leq z_j^{\mathrm{avg}}$, when applying the labeling algorithms to solve SP1 and rSP1. Additionally, we remove from the RMPs all columns corresponding to routes that do not satisfy $z_{jr} \leq z_j^{\mathrm{avg}}$. In child node 2, we set a time window for the restoration time of vertex j such that $Z_j^{\mathrm{r}} > z_j^{\mathrm{avg}}$ in the labeling algorithms that solve SP1 and rSP1. In the RMPs, we drop all columns corresponding to routes that do not satisfy $z_{jr} > z_j^{\mathrm{avg}}$. For branching on the restoration time of damaged vertices, the metric h is defined as $h = z_j^{\mathrm{avg}}/z_j^{\max}$.

5.6.3. Branching on the vertices used in relief paths Assume that $\bar{V}_{jl}$ is not integer for some $j \in \mathcal{V}$ in RMP1 or $j \in \mathcal{V}^{\mathrm{r}}$ in RMP2 or RMP3, and $l \in \mathcal{V}^{\mathrm{d}}$. In child node 1, we set $V_{jl} = 0$ in RMP1; drop from RMP2 and RMP3 all columns corresponding to relief paths that do not satisfy $\bar{V}_{jl} = 0$; and remove vertex j and all edges adjacent to it from graph G^i in the labeling algorithm that solves SP2. In child node 2, we set $V_{jl} = 1$ in the RMP1; drop from RMP2 and RMP3 all columns corresponding to relief paths that do not satisfy $\bar{V}_{jl} = 1$; and add the term $(1 - V_{jl})\max\{0, 2\eta_l\}$ to the objective function of SP2 to force the restoration of damaged vertex j. For branching on the vertices used in relief paths, the metric h is defined as $h = 10(\frac{1}{2}h_1 + \frac{1}{2}h_2)$, where $h_1 = 2(0.5 - |1 - \bar{V}_{jl} - 0.5|)$, $h_2 = \frac{d_l}{2d^*} + \frac{\delta_j}{2\delta^*}$ if $j \in \mathcal{V}^{\mathrm{r}}$, $h_2 = \frac{d_l}{2d^*}$ if $j \in \mathcal{V} \setminus \mathcal{V}^{\mathrm{r}}$, $d^* = \max_{l'\in\mathcal{V}^{\mathrm{d}}}\{d_{l'}\}$, $\delta^* = \max_{j'\in\mathcal{V}^{\mathrm{r}}}\{\delta_{j'}\}$.

5.7. Improvement Strategies

Different strategies were implemented to improve the performance of the BP algorithms. Some of the enhancements are for the labeling algorithms proposed for the pricing subproblems SP1 and rSP1 (Subsections 5.7.1, 5.7.2), while others affect the RMPs (Subsections 5.7.3, 5.7.4, 5.7.5). The proposed enhancements are based on particular properties of the problem and on speed-up strategies developed for related BP approaches.

5.7.1. Priority vertices and maximum number of vertices in crew paths Moreno, Munari, and Alem (2019) developed a procedure based on solving shortest path problems to identify the so-called priority vertices, which are damaged vertices that are needed in the relief paths

connecting the depot to demand vertices, and thus must be repaired by the crew. However, in preliminary computational experiments, we observed that the pricing subproblems SP1 and rSP1 generated routes with negative reduced cost that did not include the restoration of some priority vertices. The corresponding columns are infeasible in the original SCSRP and thus should be avoided. Hence, in the labeling algorithms, we force the consideration of the priority vertices by adding the term $|\mathcal{Q}|M - \sum_{j\in\mathcal{Q}} MW_j$ in the objective function of SP1 and rSP1, in which $M = \max\{0, 2(\sum_{j\in\mathcal{V}^{\mathrm{r}}} \mu_j - \nu)\}$ and $\mathcal{Q}$ is the set of priority vertices. It is important to highlight that, although priority vertices are necessary to restore the connectivity of the network, they may not be sufficient.

Moreover, we resort to a procedure to calculate the maximum number of damaged vertices ($R_{ij}^{\max}$) that should be considered in a crew path p_{ij}, as introduced by Moreno et al. (2020). Such calculation is performed using an algorithm that finds the elementary longest path (the one with more damaged vertices) from vertex i to vertex j. Then, in the labeling algorithm for SP1, we limit to $R_{ij}^{\max}$ the number of vertices that can be visited in the crew path between vertices i and j.

5.7.2. Heuristic approaches to accelerate the labeling algorithms We adapted the savings heuristic (Clarke and Wright 1964) to generate initial routes in our labeling algorithms. In addition, local search heuristics based on vertex insertion, deletion and exchange operations were implemented to improve the quality of the initial solutions. In some cases, the proposed heuristics were able to generate routes with negative reduced cost. Therefore, we add directly such solutions to the RMPs without proving their optimality. It is well-known that suboptimal solutions with negative reduced cost can be used to generate columns in the RMPs.

We also generate suboptimal solutions with negative reduced cost using the relaxed pricing subproblem rSP1. Basically, we solve this subproblem using the labeling algorithm, verify if the relaxed solutions are feasible, and then derive solutions for SP1 using the algorithm presented in Appendix A. If a route with a negative reduced cost is found for SP1, we add the corresponding columns to the RMPs without solving SP1. Otherwise, we solve SP1.

5.7.3. Multiple feasible columns added to the RMPs by iteration The labeling algorithm for SP1 may generate many routes with negative reduced cost. In this case, we add multiple columns to the RMPs. Although such columns can be feasible for the RMPs in a given branching node of the BP tree, some of them may be infeasible for the original SCSRP when integrality constraints are considered. We observe that, in most cases, a column with negative reduced cost is infeasible for the original SCSRP when the repaired vertices are not enough to restore the connectivity of the network, i.e., define the relief paths from the depot to demand vertices.

Hence, for each solution obtained by the labeling algorithm for SP1, we check if the corresponding pair schedule-route is feasible for the original SCSRP, before adding the column to the RMPs. To do so, we create an auxiliary graph G^{check} by removing from graph G the damaged vertices that were not repaired in the solution. Then, for each demand vertex i, we find the shortest relief path (p_i) from the depot to vertex i using Dijkstra's algorithm. If the distance of path p_i is greater than $l_i^{\mathtt{d}}$ or path p_i does not exist, then the pair schedule-route is infeasible for the original SCSRP and the corresponding column is discarded. If all the solutions generated by the labeling algorithm are infeasible for the original RMPs, we still add the column with the lowest reduced cost.

5.7.4. Best route selection Given a scheduling decision p, the best route r^* for the crew can be determined using the solution procedure presented in Appendix A. Thus, instead of considering all possible routes $r \in \mathcal{R}_p$ in the master problems, we modify them to consider only the optimal routes r^* for a schedule $p \in \mathcal{P}$. For example, MP1 is redefined as follows:

$$
\begin{aligned}
(\textbf{MP1*}) \quad & \min \sum_{i \in \mathcal{V}^{\mathtt{d}}} d_i Z_i^{\mathtt{d}}, \\
& \text{s.t. Constraints (2), (13)-(16), (20)-(23),} \\
& \sum_{p \in \mathcal{P}} q_{jp} \lambda_{pr^*} \geq V_{ji}, \; \forall\, i \in \mathcal{V}^{\mathtt{d}}, \, j \in \mathcal{V}^{\mathtt{r}}, && (41) \\
& Z_j^{\mathtt{r}} \geq \sum_{p \in \mathcal{P}} z_{jr^*} \lambda_{pr^*}, \; \forall \; j \in \mathcal{V}^{\mathtt{r}}, && (42) \\
& \sum_{p \in \mathcal{P}} \lambda_{pr^*} \leq 1, && (43) \\
& \lambda_{pr^*} \in \{0,1\}, \; \forall \; p \in \mathcal{P}. && (44)
\end{aligned}
$$

MP2 and MP3 are modified similarly, using only one optimal route r^* for each schedule.

The pricing subproblems are the same as the ones presented in Section 5. However, every time a pair schedule-route is obtained from the pricing subproblems, we determine the best route r^* associated to that schedule using the solution procedure presented in Appendix A.

5.7.5. Other strategies In addition to the already described strategies, we use a metaheuristic approach based on simulated annealing (SA) (Moreno, Munari, and Alem 2020) to warm-start the BP with good quality initial columns. For formulations RMP1 and RMP2, we add valid inequalities (VIs) related to relief path decisions (Moreno et al. 2020). The VIs are based on the idea that multiple paths can connect depot 0 to a demand vertex i, and that given two paths p and p', one path may dominate the other. This strategy is detailed in Appendix C. Another enhancement, based on previous works, is the graph reduction strategy proposed by Moreno, Munari, and Alem (2019). Basically, the idea is to set lower bounds for the accessibility time of the affected areas, and thus

for the total cost, based on solving smaller graphs with a reduced number of demand and damaged vertices. Finally, based on the framework proposed by Munari and Gondzio (2013), Gondzio, Gonzalez-Brevis, and Munari (2013), we do not run the CG algorithm until proven optimality at all nodes of the BP tree. Instead, we terminate the CG procedure prematurely, by setting a loose optimality tolerance, and apply early branching if the node is not eligible for pruning.

6. Computational Experiments

In this section, we present the results of computational experiments to verify the performance of the proposed algorithms. All the algorithms were coded in C++ and resorting to the libraries of the IBM CPLEX Optimizer 22.11. The experiments were conducted on a Linux PC with an Intel Core i7 CPU at 3.7 GHz and 16 GB of RAM using a single thread and a CPU time limit of one hour. The algorithms were tested using 648 benchmark instances from the literature (Maya-Duque, Dolinskaya, and Sörensen 2016, Moreno, Munari, and Alem 2019, Moreno et al. 2020). Originally, these instances were derived from undamaged original networks by varying two parameters, namely, α and β. Parameter α defines the proportion of damaged edges in the network. Parameter β specifies the factor by which the distance between the depot and the demand vertices can increase in relation to the shortest distance. The instances are described in Appendix D and publicly available at Mendelay data (Moreno et al. 2022).

6.1. Experiment Description

In this section, we summarize the different variants of the CG and BP algorithms proposed to solve the SCSRP. Initially, we test three CG algorithms, namely CG1, CG2 and CG3, to solve the linear relaxation of the proposed formulations MP1, MP2 and MP3, respectively. These CG approaches use the enhancement strategies presented in Section 5.7. For example, we depict in Figure 4 a basic scheme of CG3. The algorithm begins by adding VIs and initial columns to the RMP3. After solving this restricted master problem, it defines rSP1 using the obtained dual solution and then solves this subproblem. If a solution with a negative reduced cost regarding SP1 is derived from rSP1, the corresponding column is added to RMP3. Otherwise, it resorts to SP1. If a solution with a negative reduced cost is found with SP1, the corresponding column is added to RMP3. If not, it solves SP2 and the resulting solution with negative reduced cost, if any, is used to define a new column for RMP3. This process terminates when no column with negative reduced costs can be found. Note that every time a new column is added to RMP3, the described process is repeated.

In addition, six BP methods were considered. The first three methods (BP1, BP2, BP3) use the three CG approaches (CG1, CG2, CG3) to solve the linear relaxation at the branching nodes of the tree. These methods are exact and provide valid lower and upper bounds for the problem. The second group of BP methods (BP1H, BP2H, BP3H) are similar to BP1, BP2, and BP3, but the

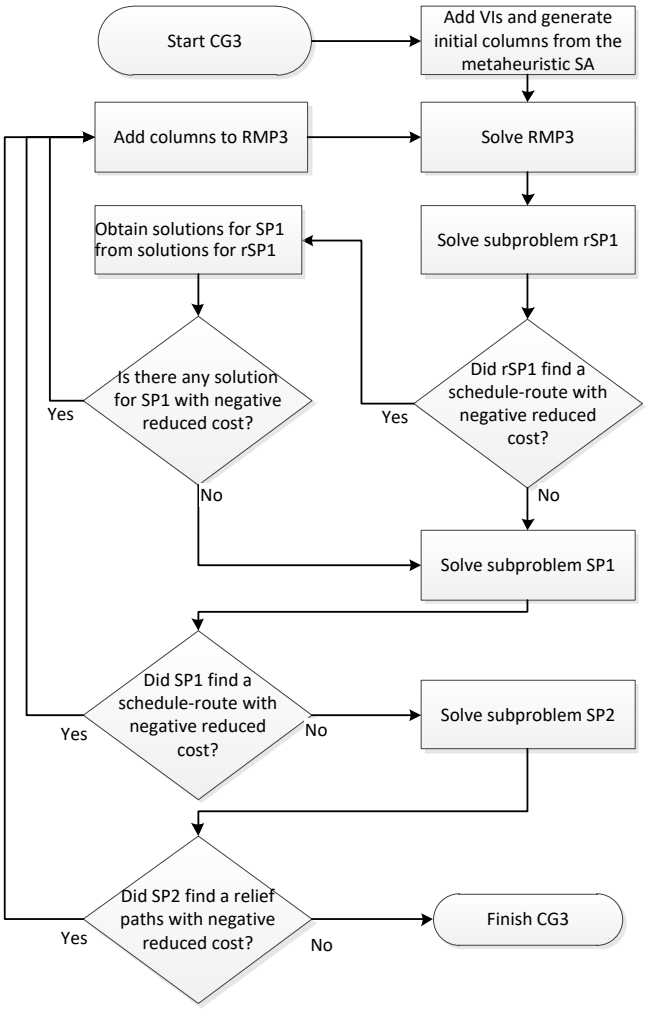


Figure 4 Basic scheme of the CG3 algorithm.

pricing subproblems are not solved to optimality. Instead, the labeling algorithms are limited to the generation of a given reduced number of states, to speed up the convergence to an integer solution (not necessarily optimal). Thus, BP1H, BP2H, and BP3H are heuristics that provide valid upper bounds for the problem. The proposed solution strategies are summarized in Table 2.

Table 2 Description of the proposed solution strategies.

Method	Description
LPR	Linear relaxation of SCSRP-CF, including VIs from the literature
CG1	CG to solve the root node of MP1 + VIs related to relief paths
CG2	CG to solve the root node of MP2
CG3	CG to solve the root node of MP3 + VIs related to relief paths
BP1	BP to solve MP1 (using CG1)
BP2	BP to solve MP2 (using CG2)
BP3	BP to solve MP3 (using CG3)
BP1H	Heuristic BP1 (pricing subproblems are not solved to optimality)
BP2H	Heuristic BP2 (pricing subproblems are not solved to optimality)
BP3H	Heuristic BP3 (pricing subproblems are not solved to optimality)

6.2. Computational Performance of the CG Approaches

We first compare the performance of the CG approaches to solve the LP relaxation of the proposed models (root node of the BP approaches). They are compared with the LP relaxation of the compact formulation SCSRP-CF, referred to as the LPR. Table 3 summarizes the average results of the LPR and CG approaches (first column). The second and third columns (#opt, %opt) indicate the number and percentage of instances for which the LP relaxation at root node was solved to optimality within one-hour time limit. The fourth and fifth columns indicate the average objective value and computation time of the approaches, respectively. Finally, from column six to eight, Table 3 shows the relative improvement in the objective value with respect to the value of the LPR ($\%\frac{(\text{CG}-\text{LPR})}{LPR}$), and the number and percentage of instances (#CG > LPR, %CG > LPR) for which the CG approach strictly improved the value of the LPR.

Evidently, the new proposed formulations (MP1, MP2, MP3) have stronger LP relaxations than the compact formulation (SCSRP-CF). In Table 3, we can observe that the average LP bound was increased by more than 40%. Improvements were observed in up to 505 (77.93%) of the tested instances. In the other instances, the CG approaches obtained the same linear relaxation cost than LPR. Note that all CG methods failed at solving the LP relaxation for some instances within the 1-hour time limit. In fact, they solved to optimality less instances than LPR, and took a larger average computation time. This is not a problem in the BP approaches since we do not need to solve to optimality all nodes in the tree (only those eligible for pruning). Instead, in the BP tree, we terminate the CG algorithms prematurely by setting a loose optimality tolerance on the first explored nodes.

Among the different CG strategies, CG1 and CG2 were the ones that solved to optimality more instances within the time limit of one hour. However, CG3 was the one that provided the best objective value, on average. Note that the valid inequalities added (in advance) to MP1 and MP3 seem to significantly improve the quality of the linear relaxation cost. CG1 and CG3 improved the average linear relaxation cost by 42.04% and 51.47%, respectively, while in CG2 the improvement was 35.60% when compared to LPR. With the explicit consideration of the relief path decision variables, CG3 seems to outperform CG2, mainly because of the valid inequalities added along with the relief path variables.

Table 3 Average results of the CG strategies on the 648 tested instances.

Approach	#opt	%opt	Avg. obj. value*	Avg. time (seconds)	CG improvements		
					$\%\frac{(\text{CG}-\text{LPR})}{LPR}$	#CG > LPR	%CG > LPR
LPR	588	90.74	27,492	387			
CG1	510	78.70	39,049	803	42.04	478	73.77
CG2	510	78.70	37,279	806	35.60	425	65.59
CG3	504	77.78	41,642	835	51.47	505	77.93
Avg. CG1-CG3	508	78.40	39,323	815	43.03	469	72.43

* If not solved to optimality, a lower bound is obtained from the graph reduction strategy.

Table 4 shows the improvements of the CG approaches in relation to LPR for different classes of instances grouped according to the network size. The first column indicates the instance sets, while the second column indicates the average number of demand nodes, total nodes, and total arcs of the original networks from which the instances were generated. The third column shows the average LP bound obtained with LPR (LPR obj. value). The following columns indicate, for the different CG strategies, the average LP bound (obj. value) and the improvement in relation to the bounds obtained with LPR. Better improvements in the objective values were observed for larger network classes. For example, CG3 improved by 37.33%, on average, the objective value for instances L1-L15, while the improvement for instances L16-L39 was 47.87%. In fact, the improvement was remarkably up to 80.97% for some instance sets. Detailed results of the CG strategies for the different instances classes are presented in Appendix E.

Table 4 Lower bound improvements of the CG strategies.

Instance	Original network	LPR	CG1		CG2		CG3	
sets	$\lvert\mathcal{V}^{\mathrm{d}}\rvert$-$\lvert\mathcal{V}\rvert$-$\lvert\mathcal{E}\rvert$	Obj. value	Obj. value	%imp.*	Obj. value	%imp.*	Obj. value	%imp.*
L1, L2, L3	19-25-39	17,291	26,389	30.48	26,157	28.67	28,155	34.71
L4, L5, L6	24-30-85	9,347	15,751	40.34	15,309	38.54	16,180	42.19
L7, L8, L9	28-35-115	13,833	23,565	40.15	22,565	37.55	24,679	43.30
L10, L11, L12	15-20-38	24,323	33,294	27.60	30,396	20.31	34,673	30.54
L13, L14, L15	35-40-144	22,686	35,264	35.56	33,979	32.69	35,196	35.89
L16, L17, L18	50-60-195	29,739	34,552	17.09	35,667	19.31	35,080	18.46
L19, L20, L21	70-80-247	13,954	20,196	30.95	19,236	27.25	20,123	30.60
L22, L23, L24	90-100-273	30,892	37,761	20.95	35,603	16.59	36,882	19.85
L25, L26, L27	125-140-323	12,963	22,449	41.35	22,053	40.58	21,207	37.28
L28, L29, L30	140-170-398	13,772	23,596	42.05	25,030	45.68	27,036	49.16
L31, L32, L33	200-200-448	5,075	15,336	66.98	15,235	66.77	15,352	67.00
L34, L35, L36	300-300-525	4,325	21,167	79.64	21,045	79.51	21,167	79.64
L37, L38, L39	400-400-625	4,195	20,610	79.34	21,740	80.69	22,014	80.97
CS1, CS2, CS3	13-60-89	66,283	82,866	11.42	80,907	15.13	96,600	31.09
CS4, CS5, CS6	20-60-89	78,820	127,832	38.77	115,098	31.33	138,311	43.29
L1-L15	24-30-84	17,496	26,853	34.83	25,681	31.55	27,777	37.33
L16-L39	172-181-379	14,365	24,458	47.29	24,451	47.05	24,858	47.87
CS1-CS6	17-60-89	72,551	105,349	25.10	98,002	23.23	117,456	37.19

* $\%\mathrm{imp} = \frac{\text{Obj. value in CG} - \text{Obj. value in LPR}}{\text{Obj. value in LPR}}(\%)$

6.3. Computational Performance of the BP Approaches

Next, we analyze the computational performance of the proposed BP approaches. Table 5 summarizes the average results of the BP methods and shows a comparison with the best solutions found in the literature for the tested instances (LM) (Moreno et al. 2020). The first column indicates the solution approaches, while the second and third columns show the number and percentage of instances with solutions that have been proved optimal. Columns 4 to 8 present the average upper bound, lower bound, gap, computation time, and number of branching nodes, respectively. The following columns show the improvement of the BP strategies with respect to the best upper bounds and gaps from the literature.

Table 5 Average results of the BP methods for the 648 tested instances.

Solution approach	#opt	%opt	Avg. UB	Avg. LB[2]	Avg. gap	Avg. time (seconds)	# branching nodes	BP improvements (UB)			BP improvements (gap)			
								Ratio[3]	#imp	%imp	Diff[3]	Ratio[3]	#imp	%imp
LM[1]	332	51.23	96,683	42,206	23.07	1,899								
BP1	387	59.72	93,461	46,199	18.55	1,533	3,709	3.33	41	6.33	4.52	19.60	227	35.03
BP2	440	67.90	92,409	47,376	17.58	1,355	90	4.42	58	8.95	5.49	23.82	262	40.43
BP3	450	69.44	93,018	47,817	17.01	1,228	59	3.79	55	8.49	6.06	26.26	271	41.82
BP1H	0	0.00	91,838	46,199	NA	1,206	2,852	5.01	69	10.65	NA	NA	NA	NA
BP2H	0	0.00	92,220	47,304	NA	656	109	4.62	64	9.88	NA	NA	NA	NA
BP3H	0	0.00	91,916	47,817	NA	612	110	4.93	62	9.57	NA	NA	NA	NA

NA: Not available.
[1] LM: Best solutions found in the literature for the tested instances (Maya-Duque, Dolinskaya, and Sörensen 2016, Moreno et al. 2020).
[2] If not solved to optimality by the BP, a lower bound is obtained from the graph reduction strategy.
[3] Diff = Obj. value in LM − Obj. value in BP; Ratio = $\frac{\text{Obj. value in LM} - \text{Obj. value in BP}}{\text{Obj. value in LM}}$(%).

In general, the BP approaches improved the average upper bound, lower bound and gap of the solutions from the literature. Regarding the gap, BP3 showed the best overall performance, obtaining smaller gaps for 271 (41.82%) of the instances when compared with LM. The relative (absolute) reduction of the average gap was 26.26% (6.06), from 23.07 with LM to 17.01 with BP3. BP3 was also the method that obtained the higher number of proven optimal solutions, for 450 (69.44%) instances, while BP2 and BP1 proved optimality for 440 (67.90%) and 387 (59.72%) instances, respectively. The approaches proposed so far in literature have managed to prove optimality for 332 (51.2%) instances only. Finally, notice that BP3 is the method that provided the greater average lower bound. On average, the lower bound increased 13.29% from 42,206 with LM to 47,817 with BP3. BP1 was the solution approach that presented the smallest improvement in the lower bound.

Figure 5 presents the performance profiles (Dolan and Moré 2002) for the BP approaches based on upper bounds for the considered instances. Given a set $\mathcal{P}$ of instances and a set $\mathcal{F}$ of solution methods, let UB_{fp} be the upper bound of the solution of instance p solved by method f. The value $P(f,q)$ (y-axis) when $q>0$ (x-axis) indicates the fraction of instances for which strategy f provides solutions with an upper bound within a factor of 2^q of the best obtained upper bound, i.e., the fraction of instances for which $UB_{fp}+\epsilon \leq 2^q \cdot \min_{f' \in \mathcal{F}}\{UB_{f'p}+\epsilon\}$, where $\epsilon = 0.01$ is a near-zero value. The value of $P(f,q)$ when $q=0$ is the fraction of instances for which the strategy f reached the best upper bound. For example, the red asterisk (*) in Figure 5 indicates that for 98% of the instances, strategy BP1H provides solutions with upper bound within a factor of $2^{0.03}$ (1.02) of the best upper bound among all methods. Indeed, $UB_{fp}+\epsilon \leq 1.02 \cdot \min_{f' \in \mathcal{F}}\{UB_{f'p}+\epsilon\}$ for 98% of the instances, with $f =$ BP1H and $F = \{$BP1, BP2, BP3, BP1H, BP2H, BP3H$\}$.

Figure 5 evidences that the BP methods presented a similar performance for approximately 84% of the instances regarding the obtained upper bounds. Interestingly, the heuristic BP1H is the method showing the best performance for most of the instances, whereas the worst performance was due to BP1, the exact version of BP1H. In general, the heuristic methods provided good upper bounds.

In summary, BP3 and BP1H are approaches with the best performance related to optimality gap and upper bound, respectively. BP3 proved optimality for 118 (18.21%) new instances for the first

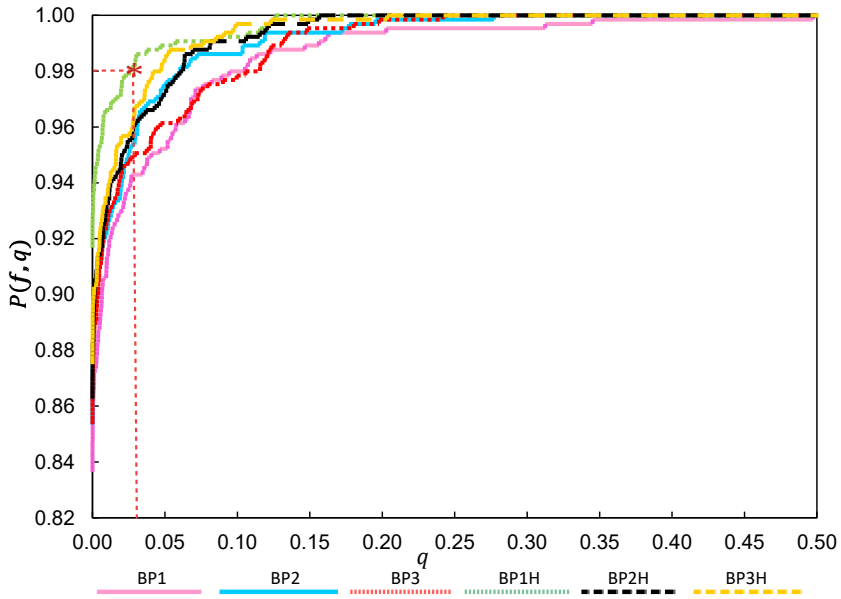


Figure 5 **Performance profiles based on upper bounds for the BP approaches.**

time and achieved better gaps for other 153 (23.61%) instances. BP1H reduced the upper bound by 5.01%, on average, when compared with LM. The reduction on the upper bound was for 69 (10.65%) instances, most of which are large instances. Overall and considering all approaches, the gap was improved by more than 50% in 25% of the instances. In terms of upper bound (total accessibility time), the improvement was up to 10% for most instances. In some cases, the total accessibility time is dramatically reduced (by more than 40%), which is of utmost importance to mitigate human suffering in a disaster aftermath. Detailed results of the BP strategies for the different instances classes are presented in Appendix E

Tables 6 and 7 show, for instances L1-L39 and CS1-CS6, respectively, the average gap and computation time of BP3 according to different values of α and β. The parameter α defines the proportion of damaged edges in the network, while β specifies the factor by which the distance between the depot and the demand vertices can increase in relation to the shortest path (see Appendix D for details). As in previous studies (Maya-Duque, Dolinskaya, and Sörensen 2016, Moreno, Munari, and Alem 2019), the instances become more challenging when the percentage of damage (α) increases. For example, the average computation time was 176 seconds for instances L1-L39 when $\alpha = 5\%$ and 2,796 seconds when $\alpha = 50\%$. A more damaged network leads to an increase in crew schedules and routes, consequently raising the number of columns to be added to the MP and slowing down the convergence of the BPs. Additionally, solving SP1 becomes more challenging when there are more damaged vertices in the network.

While previous studies have shown that the difficulty of solving an instance decreases as the maximum tolerable percentage (β) increases, this behavior is not particularly pronounced for BP3.

Table 6 **Average gap and computation time of BP3 according to different α and β values for instances L1-L39.**

Gap							Time (seconds)						
	α							α					
β	5	10	25	30	50	Avg.	β	5	10	25	30	50	Avg.
5	0.20	9.62	23.34	29.01	36.99	19.84	5	178	943	1,636	1,747	2,869	1,475
10	0.00	7.24	25.93	29.24	40.59	20.60	10	166	947	1,654	1,654	3,045	1,493
25	0.00	4.61	23.90	30.16	43.90	20.51	25	181	569	1,491	1,777	2,623	1,328
50	0.00	8.37	24.68	25.78	43.92	20.55	50	177	458	1,495	1,611	2,645	1,277
Avg.	0.05	7.46	24.46	28.55	41.35	20.37	Avg.	176	729	1,569	1,697	2,796	1,393

Table 7 **Average gap and computation time of BP3 according to different α and β values for instances CS1-CS6.**

Gap					Time (seconds)				
	α					α			
β	5	10	14	Avg.	β	5	10	14	Avg.
5	0.00	0.00	0.00	0.00	5	13	26	45	28
10	0.00	0.00	0.00	0.00	10	13	36	269	106
25	0.00	0.15	0.00	0.05	25	13	631	331	325
50	0.00	0.00	0.35	0.12	50	15	238	477	243
100	0.00	0.00	1.74	0.58	100	3	929	1,895	942
0	0.00	0.00	0.00	0.00	0	1	3	3	3
Avg.	0.00	0.02	0.35	0.12	Avg.	10	311	503	274

For example, L1-L39 exhibited similar average gaps and computation times across different β values. Conversely, instances CS1-CS6 appear to become more challenging with increasing β. Nonetheless, instances with $\beta = \infty$ were the easiest in this case. Overall, BP3 showed longer computation times for higher β values since more states have to be generated in SP2 and more columns are needed in the MP. However, if β increases greatly, it is more likely to find non-dominated relief paths, generating stronger VIs in the MP and thus speeding up the method.

Finally, we conducted further experiments to test the relevance of different improvement strategies to CG3, the column generation algorithm of BP3. In Appendix F, we describe the results of six alternative configurations of CG3 in which we turn off some improvement strategies. One of the most effective strategies seem to be using the relaxed subproblem rSP. When subproblem rSP is not adopted, the CG algorithm becomes slower since SP1 is more challenging to be solved than rSP1, and it must be solved in all iterations to find columns with negative reduced cost. In this case, the computational time increased by 144%. To achieve good results using CG3, strategies for efficiently eliminating states in the labeling algorithm are also essential. We performed experiments in which we turned off the use of lower bounds for the reduced cost and the use of strong dominance rules by considering $V_i^1 = V_i^2$ in condition (ii) of Proposition 1. By tuning off both strategies, the computation time increased by 164%, while the linear relaxation was optimality proven only in 5.4% of the instances, compared to 77.8% in CG3.

7. Conclusions and Future Research

We proposed three formulations based on the enumeration of feasible routes, schedules and paths for the single crew scheduling and routing problem (SCSRP) in road restoration, and designed column generation (CG) and branch-and-price (BP) algorithms to solve each of them. For the first

formulation, we generate columns representing the crew schedules and routes while the relief paths are defined explicitly. In the second formulation, we generate columns representing the schedules, routes and relief paths. Finally, the third formulation builds upon the second by the incorporation of additional constraints and variables related to relief path decisions. These are the first formulations and solution approaches for the SCSRP that are suitable to be solved by BP strategies. To improve the performance of these approaches, we proposed labeling algorithms tailored for the subproblems, a branching scheme based on different types of rules, and several other algorithmic enhancements. We also derived effective heuristic algorithms from the proposed exact approaches.

The results of extensive computational experiments with 648 benchmark instances showed that the proposed methods outperform existing state-of-the-art solution approaches. First, we have compared the computational performance of the CG approaches to solve the LP relaxation of the proposed models. The results evidenced that the proposed formulations have a considerably stronger LP relaxation than the compact formulation used in previous studies, providing bounds that are 40% large, on average. In general, the BPs improved the average upper bound, lower bound, and optimality gaps of the known solutions from the literature. The BP based on the third formulation (BP3) showed the best overall performance, as it proved optimality for 108 (16.67%) new instances for the first time and achieved better gaps for other 159 (24.53%) instances. In relation to the upper bound, all the BP methods presented a similar performance for approximately 82% of the instances. The heuristic derived from the BP based on the first formulation (BP1H) was the method that presented the best overall performance in terms of upper bounds. On average, BP1H reduced the upper bounds by 4.68% when compared with the solutions from the literature. This reduction was in 65 (10.03%) instances, mainly, on larger instances. Finally, one key observation is that the improvement in the performance of the BP approaches with respect to existing methods was significantly more pronounced for larger instances (7.61%, on average). In some large instance classes, BP1H improved the upper bound for up to 60% of the instances. The instances based on real data had their optimality gaps reduced from 4.39 to 0.20, on average.

There are several avenues that can be further explored in future research. Regarding the solution methods, one can devise other types of valid inequalities tailored for the proposed formulations and the CG algorithm to improve the lower bounds and reduce the gap in even larger instances. Other promising research direction is to extend the solution approaches to consider other variants of the problem with distinct characteristics such as relief distribution, damaged edges, and alternative objective functions such as latency. Also, the proposed BPs can be extended to consider multiple crews, used as a subroutine for multi-crew problems, or adapted to solve scheduling and routing problems in other contexts.

Acknowledgments

This work was supported by the São Paulo Research Foundation (FAPESP) [grant numbers 13/07375-0, 15/26453-7, 16/15966-6 and 22/05803-3]; the National Council for Scientific and Technological Development (CNPq) [grant number 313220/2020-4]; and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) [Finance Code 001].

References

Ajam M, Akbari V, Salman FS, 2019 *Minimizing latency in post-disaster road clearance operations. European Journal of Operational Research* 277(3):1098–1112.

Ajam M, Akbari V, Salman FS, 2022 *Routing multiple work teams to minimize latency in post-disaster road network restoration. European Journal of Operational Research* 300(1):237–254.

Akbari V, Sadati MEH, Kian R, 2021 *A decomposition-based heuristic for a multicrew coordinated road restoration problem. Transportation Research Part D: Transport and Environment* 95(April):102854.

Akbari V, Salman FS, 2017a *Multi-vehicle prize collecting arc routing for connectivity problem. Computers and Operations Research* 82:52–68.

Akbari V, Salman FS, 2017b *Multi-vehicle synchronized arc routing problem to restore post-disaster network connectivity. European Journal of Operational Research* 257(2):625–640.

Akbari V, Sayarshad HR, 2022 *Integrated and coordinated relief logistics and road recovery planning problem. Transportation Research Part D: Transport and Environment* 111(September):103433.

Almeida LS, Goerlandt F, Pelot R, 2022 *Trends and gaps in the literature of road network repair and restoration in the context of disaster response operations. Socio-Economic Planning Sciences* 36:101398.

Alvarez A, Munari P, 2017 *An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen. Computers & Operations Research* 83:1–12.

Berktaş N, Kara BY, Karaşan OE, 2016 *Solution methodologies for debris removal in disaster response. EURO Journal on Computational Optimization* 4(3-4):403–445.

Bono F, Gutiérrez E, 2011 *A network-based analysis of the impact of structural damage on urban accessibility following a disaster: The case of the seismically damaged port au prince and carrefour urban road networks. Journal of Transport Geography* 19:1443–1455.

Caunhye AM, Aydin NY, Duzgun HS, 2020 *Robust post-disaster route restoration. OR Spectrum* 42:1055–1087.

Çelik M, 2016 *Network restoration and recovery in humanitarian operations: Framework, literature review, and research directions. Surveys in Operations Research and Management Science* 21(2):47–61.

Çelik M, Ergun Ö, Johnson B, Keskinocak P, Lorca Á, Pekgün P, Swann J, 2012 *Humanitarian logistics. New directions in informatics, optimization, logistics, and production*, 18–49 (INFORMS).

Çelik M, Ergun Ö, Keskinocak P, 2015 *The post-disaster debris clearance problem under incomplete information. Operations Research* 63(1):65–85.

Clarke G, Wright JW, 1964 *Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. Operations Research* 12(4):568–581.

Costa L, Contardo C, Desaulniers G, 2019 *Exact Branch-Price-and-Cut Algorithms for Vehicle Routing. Transportation Science* 53(4):946–985.

Dabia S, Ropke S, Van Woensel T, De Kok T, 2013 *Branch and price for the time-dependent vehicle routing problem with time windows. Transportation Science* 47(3):380–396.

Desaulniers G, Villeneuve D, 2000 *Shortest path problem with time windows and linear waiting costs. Transportation Science* 34(3):312–319.

Dolan ED, Moré JJ, 2002 *Benchmarking optimization software with performance profiles. Mathematical Programming* 91(2):201–213.

Elifcan Y, Dilek TA, Linet Ö, 2022 *Metaheuristics for the Stochastic Post-Disaster Debris Clearance Problem. IISE Transactions* 0(0):1–33.

Farzaneh MA, Rezapour S, Baghaian A, Amini MH, 2023 *An integrative framework for coordination of damage assessment, road restoration, and relief distribution in disasters. Omega* 115:102748.

Feillet D, 2010 *A tutorial on column generation and branch-and-price for vehicle routing problems. 4OR: A Quarterly Journal of Operations Research* 8:407–424.

Feillet D, Dejax P, Gendreau M, Gueguen C, 2004 *An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. Networks* 44(3):216–229.

García-Alviz J, Galindo G, Arellana J, Yie-Pinedo R, 2021 *Planning road network restoration and relief distribution under heterogeneous road disruptions. OR Spectrum* 43(4):941–981.

Gondzio J, Gonzalez-Brevis P, Munari P, 2013 *New developments in the primal-dual column generation technique. European Journal of Operational Research* 224(1):41–51.

Horváth M, Kis T, 2019 *Computing strong lower and upper bounds for the integrated multiple-depot vehicle and crew scheduling problem with branch-and-price. Central European Journal of Operations Research* 27(1):39–67.

Ioachim I, Gélinas S, Soumis F, Desrosiers J, 1998 *A Dynamic Programming Algorithm for the Shortest Path Problem with Time Windows and Linear Node Costs. Networks* 31(3):193–204.

Irnich S, Desaulniers G, 2005 *Shortest path problems with resource constraints.* Desaulniers G, Desrosiers J, Solomon MM, eds., *Column Generation*, 33–65 (Springer US).

Kasaei M, Salman FS, 2016 *Arc routing problems to restore connectivity of a road network. Transportation Research Part E: Logistics and Transportation Review* 95:177–206.

Kim S, Shin Y, Lee GM, Moon I, 2018 *Network repair crew scheduling for short-term disasters. Applied Mathematical Modelling* 64:510–523.

Lakzaei S, Rahmani D, Tosarkani BM, Nasiri S, 2023 *Integrated optimal scheduling and routing of repair crew and relief vehicles after disaster: a novel hybrid solution approach. Annals of Operations Research* 328:1495–1522.

Legrain A, Omer J, Rosat S, 2020 *A rotation-based branch-and-price approach for the nurse scheduling problem. Mathematical Programming Computation* 12(3):417–450.

Li S, Ma Z, Teo KL, 2020 *A new model for road network repair after natural disasters: Integrating logistics support scheduling with repair crew scheduling and routing activities. Computers & Industrial Engineering* 145:106506.

Li S, Teo KL, 2019 *Post-disaster multi-period road network repair: work scheduling and relief logistics optimization. Annals of Operations Research* 283(1-2):1345–1385.

Lin DY, Juan CJ, Chang CC, 2020 *A Branch-and-Price-and-Cut Algorithm for the Integrated Scheduling and Rostering Problem of Bus Drivers. Journal of Advanced Transportation* 2020:1–19.

Masson A, 2014 *The extratropical transition of Hurricane Igor and the impacts on Newfoundland. Natural Hazards* 72(2):617–632.

Maya-Duque PA, Dolinskaya IS, Sörensen K, 2016 *Network repair crew scheduling and routing for emergency relief distribution problem. European Journal of Operational Research* 248(1):272–285.

Moreno A, Alem D, Ferreira D, Clark A, 2018 *An effective two-stage stochastic multi-trip location-transportation model with social concerns in relief supply chains. European Journal of Operational Research* 269(3):1050–1071.

Moreno A, Alem D, Gendreau M, Munari P, 2020 *The heterogeneous multicrew scheduling and routing problem in road restoration. Transportation Research Part B: Methodological* 141:24–58.

Moreno A, Maya-Duque A, Alem D, Munari P, 2022 *Data set for the crew scheduling and routing problem in road restoration. Mendeley Data* v1.

Moreno A, Munari P, Alem D, 2019 *A branch-and-Benders-cut algorithm for the Crew Scheduling and Routing Problem in road restoration. European Journal of Operational Research* 275(1):16–34.

Moreno A, Munari P, Alem D, 2020 *Decomposition-based algorithms for the crew scheduling and routing problem in road restoration. Computers & Operations Research* 119:104935.

Munari P, Gondzio J, 2013 *Using the primal-dual interior point algorithm within the branch-price-and-cut method. Computers & Operations Research* 40(8):2026 – 2036.

Munari P, Morabito R, 2018 *A branch-price-and-cut algorithm for the vehicle routing problem with time windows and multiple deliverymen. TOP* 26(3):437–464.

Munari P, Moreno A, De La Vega J, Alem D, Gondzio J, Morabito R, 2019 *The Robust Vehicle Routing Problem with Time Windows: Compact Formulation and Branch-Price-and-Cut Method. Transportation Science* 53(4):1043–1066.

Ruther S, Boland N, Engineer FG, Evans I, 2017 *Integrated Aircraft Routing, Crew Pairing, and Tail Assignment: Branch-and-Price with Many Pricing Problems. Transportation Science* 51(1):177–195.

Sahin H, Kara BY, Karasan OE, 2016 *Debris removal during disaster response: A case for Turkey. Socio-Economic Planning Sciences* 53:49–59.

Shin Y, Kim S, Moon I, 2019 *Integrated optimal scheduling of repair crew and relief vehicle after disaster. Computers and Operations Research* 105:237–247.

Souza Almeida L, Goerlandt F, 2022 *An ant colony optimization approach to the multi-vehicle prize-collecting arc routing for connectivity problem. Multimodal Transportation* 1(3):100033.

Stålhane M, Andersson H, Christiansen M, Cordeau JF, Desaulniers G, 2012 *A branch-price-and-cut method for a ship routing and scheduling problem with split loads. Computers & Operations Research* 39(12):3361–3375.

Tuzun Aksu D, Ozdamar L, 2014 *A mathematical model for post-disaster road restoration: Enabling accessibility and evacuation. Transportation Research Part E: Logistics and Transportation Review* 61:56–67.

Yan S, Shih YL, 2007 *A time-space network model for work team scheduling after a major disaster. Journal of the Chinese Institute of Engineers* 30(1):63–75.

Yan S, Shih YL, 2009 *Optimal scheduling of emergency roadway repair and subsequent relief distribution. Computers and Operations Research* 36(6):2049–2065.

Yuan B, Liu R, Jiang Z, 2015 *A branch-and-price algorithm for the home health care scheduling and routing problem with stochastic service times and skill requirements. International Journal of Production Research* 53(24):7450–7464.

Zamorano E, Stolletz R, 2017 *Branch-and-price approaches for the Multiperiod Technician Routing and Scheduling Problem. European Journal of Operational Research* 257(1):55–68.

Appendix A: Feasibility Check Algorithm

This appendix presents the feasibility check algorithm used to determine the best crew route r^* associated with a schedule K, adapted from (Moreno, Munari, and Alem 2019, 2020). Let $K = (v_0, v_1, ..., v_{(h-1)}, v_h, ..., v_p, ..., v_{|\mathcal{V}^r|})$ represent a schedule for the crew, where v_i denotes the ith damaged vertex to be repaired, and $v_0 = 0$. Algorithm 1 finds the optimal paths between damaged vertices for a given schedule K.

Algorithm 1 Feasibility check algorithm (adapted from Moreno, Munari, and Alem (2019, 2020)).

`Input:`

Graph $G = (\mathcal{V}, \mathcal{E})$; Schedule $K = (v_0, v_1, ..., v_j, ..., v_{|\mathcal{V}^r|})$; Parameters δ_j, $\forall j \in \mathcal{V}^r$, and τ_e, $\forall e \in \mathcal{E}$;

`Output:`

If K is feasible in the CSRP, return "Feasible Schedule" and save optimal values of Z^r_j, $\forall j \in \mathcal{V}^r$;
If schedule K is infeasible in the original CSRP, return "Infeasible Schedule";

$C_e := \tau_e$, $\forall e \in \mathcal{E}$; $C_e := \infty$, $\forall e \in \mathcal{E}_j, j \in \mathcal{V}^r$; $Z^r_j := 0$, $\forall j \in \mathcal{V}^r$;
for $j = 1$ **to** $|\mathcal{V}^r|$ **do**
  $C_e := \tau_e + \delta_{v_j}$, $\forall e \in \mathcal{E}_{v_j}$;
  Find the cost $\mathcal{C}$ of the shortest path from vertex $v_{j-1} \in K$ to vertex $v_j \in K$ by Dijkstra's algorithm;
  if $\mathcal{C} < \infty$ **then**
    $Z^r_{v_j} := Z^r_{v_{j-1}} + \mathcal{C}$;
    $C_e := \tau_e$, $\forall e \in \mathcal{E}_{v_j} : e \notin \bigcup_{i=j+1}^{|\mathcal{V}^r|} \mathcal{E}_{v_i}$ and $C_e := \infty$, $\forall e \in \mathcal{E}_{v_j} : e \in \bigcup_{i=j+1}^{|\mathcal{V}^r|} \mathcal{E}_{v_i}$;
  else
    return "Infeasible Schedule";
  end if
end for
return "Feasible Schedule";

Appendix B: Proof of Propositions

In this section, we present the proof of propositions related to dominance rule (subsection B.1) and lower bound for the reduced cost in the labeling algorithm (subsection B.2).

B.1. Proof of Dominance Rule

We will prove the proposition for the labeling algorithm to solve SP1. The proof for the labeling algorithm to solve rSP1 is straightforward. We need to show that, under Hypotheses (i)–(iii), any feasible extension of L^2_i is also feasible for L^1_i and leads to a column with a smaller or equal reduced cost. Let $e \in E(L^2_i)$ be a feasible extension of label L^2_i that visits $k > 0$ vertices and ends at vertex j, resulting in label L^2_j. Let $S_e = \{e_1, \ldots, e_k\}$ be the ordered set of vertices in extension e, where $e_k = j$. Let R^2_l be the total time consumption after visiting a vertex $l \in S_e$, and let $P^2 = \{S_e\} \cap \{\mathcal{V}^r \setminus V^2_i\}$ be the set of damaged vertices that were not visited in label

L_i^2, but visited in extension S_e. Since we do not have resource constraints, extension e is also feasible for label L_i^1, resulting in label L_j^1. In this context, the reduced cost for labels L_j^1 and L_j^2 are

$$C_j^1 = C_i^1 - \sum_{e_r \in P^1} (\mu_{e_r} - v_{e_r} R_{e_r}^1),$$

$$C_j^2 = C_i^2 - \sum_{e_r \in P^2} (\mu_{e_r} - v_{e_r} R_{e_r}^2).$$

Then,

$$C_i^1 = C_j^1 + \sum_{e_r \in P^1} (\mu_{e_r} - v_{e_r} R_{e_r}^1),$$

$$C_i^2 = C_j^2 + \sum_{e_r \in P^2} (\mu_{e_r} - v_{e_r} R_{e_r}^2).$$

The time consumption values for vertex $e_r \in S_e$ in the extension e from labels L_i^1 and L_i^2 are

$$R_{e_r}^1 = t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j + R_i^1,$$

$$R_{e_r}^2 = t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j + R_i^2.$$

Then,

$$R_i^1 = R_{e_r}^1 - (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j),$$

$$R_i^2 = R_{e_r}^2 - (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j).$$

Now, we analyse the three cases considered in hypothesis (ii).

CASE I: $V_i^1 = V_i^2$.

Consider $R_i^1 \leq R_i^2$ (hypothesis (iii)), then

$$R_{e_r}^1 - (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j) \leq R_{e_r}^2 - (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j),$$

and

$$R_{e_r}^1 - \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j \leq R_{e_r}^2 - \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j.$$

Since $V_i^1 = V_i^2$, we have that $P^1 = P^2$. Then, $R_{e_r}^1 \leq R_{e_r}^2, \forall e_r \in S_e$. From $C_i^1 \leq C_i^2$ (hypothesis (i)), we have that

$$C_j^1 + \sum_{e_r \in P^1} (\mu_{e_r} - v_{e_r} R_{e_r}^1) \leq C_j^2 + \sum_{e_r \in P^2} (\mu_{e_r} - v_{e_r} R_{e_r}^2),$$

and

$$\sum_{e_r \in P^1} \mu_{e_r} - \sum_{e_r \in P^2} \mu_{e_r} - \sum_{e_r \in P^1} v_{e_r} R_{e_r}^1 + \sum_{e_r \in P^2} v_{e_r} R_{e_r}^2 \leq C_j^2 - C_j^1,$$

thus

$$\sum_{e_r \in P^1} v_{e_r}(R^2_{e_r} - R^1_{e_r}) \leq C^2_j - C^1_j.$$

Since $R^1_{e_r} \leq R^2_{e_r}$, then $\sum_{e_r \in P^1} v_{e_r}(R^2_{e_r} - R^1_{e_r}) \geq 0$ and $C^2_j - C^1_j \geq 0$. Consequently, $C^1_j \leq C^2_j$.

CASE II: $V^1_i \subset V^2_i$ and $\sum_{j \in V^2_i \setminus V^1_i} \delta_j \leq R^2_i - R^1_i$ and $v_j = 0$, $\forall j \in V^2_i \setminus V^1_i$.

Consider $\sum_{j \in V^2_i \setminus V^1_i} \delta_j \leq R^2_i - R^1_i$, then

$$\sum_{j \in V^2_i \setminus V^1_i} \delta_j \leq R^2_{e_r} - (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j) - R^1_{e_r} + (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j),$$

and

$$\sum_{j \in V^2_i \setminus V^1_i} \delta_j \leq R^2_{e_r} - R^1_{e_r} + \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j - \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j.$$

According to hypothesis (ii), $V^1_i \subset V^2_i$, and consequently, $P^2 \subset P^1$. Then

$$\sum_{j \in V^2_i \setminus V^1_i} \delta_j - \sum_{\substack{e_l \in P^1 \setminus P^2: \\ l \leq r}} \delta_j \leq R^2_{e_r} - R^1_{e_r}.$$

Since $\sum_{j \in V^2_i \setminus V^1_i} \delta_j - \sum_{\substack{e_l \in P^1 \setminus P^2: \\ l \leq r}} \delta_j \geq 0$, then $R^2_{e_r} - R^1_{e_r} \geq 0$, and $R^1_{e_r} \leq R^2_{e_r}$, $\forall e_r \in S_e$. Now, consider $C^1_i \leq C^2_i$ (hypothesis (i)). Then,

$$\sum_{e_r \in P^1} \mu_{e_r} - \sum_{e_r \in P^2} \mu_{e_r} - \sum_{e_r \in P^1} v_{e_r} R^1_{e_r} + \sum_{e_r \in P^2} v_{e_r} R^2_{e_r} \leq C^2_j - C^1_j,$$

and

$$\sum_{e_r \in P^1 \setminus P^2} \mu_{e_r} - \sum_{e_r \in P^1 \setminus P^2} v_{e_r} R^1_{e_r} + \sum_{e_r \in P^1 \cap P^2} v_{e_r}(R^2_{e_r} - R^1_{e_r}) \leq C^2_j - C^1_j.$$

Since $R^1_{e_r} \leq R^2_{e_r}$, then $\sum_{e_r \in P^1 \cap P^2} v_{e_r}(R^2_{e_r} - R^1_{e_r}) \geq 0$. From $v_j = 0$, $\forall j \in V^2_i \setminus V^1_i$, we have that $\sum_{e_r \in P^1 \setminus P^2} \mu_{e_r} - \sum_{e_r \in P^1 \setminus P^2} v_{e_r} R^1_{e_r} \geq 0$. Consequently, $C^2_j - C^1_j \geq 0$, i.e., $C^1_j \leq C^2_j$.

CASE III: $V^2_i \subset V^1_i$ and $\mu_j - \phi_j v_j \leq 0$, $\forall j \in V^1_i \setminus V^2_i$.

Consider $R^1_i \leq R^2_i$ (hypothesis (iii)), then

$$R^1_{e_r} - (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j) \leq R^2_{e_r} - (t_{ie_1} + t_{e_1e_2} + \ldots + t_{e_{(r-1)}e_r} + \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j),$$

and

$$R^1_{e_r} - \sum_{\substack{e_l \in P^1: \\ l \leq r}} \delta_j \leq R^2_{e_r} - \sum_{\substack{e_l \in P^2: \\ l \leq r}} \delta_j.$$

According to hypothesis (ii), $V_i^2 \subset V_i^1$ and, consequently, $P^1 \subset P^2$. Then

$$\sum_{\substack{e_l \in P^2 \setminus P^1: \\ l \leq r}} \delta_j \leq R_{e_r}^2 - R_{e_r}^1.$$

Since $\sum_{\substack{e_l \in P^2 \setminus P^1: \\ l \leq r}} \delta_j \geq 0$, then $R_{e_r}^2 - R_{e_r}^1 \geq 0$, and $R_{e_r}^1 \leq R_{e_r}^2, \forall e_r \in S_e$. For $C_i^1 \leq C_i^2$ (hypothesis (i)), we have that

$$\sum_{e_r \in P^1} \mu_{e_r} - \sum_{e_r \in P^2} \mu_{e_r} - \sum_{e_r \in P^1} \upsilon_{e_r} R_{e_r}^1 + \sum_{e_r \in P^2} \upsilon_{e_r} R_{e_r}^2 \leq C_j^2 - C_j^1,$$

and

$$\sum_{e_r \in P^2 \setminus P^1} \upsilon_{e_r} R_{e_r}^2 - \sum_{e_r \in P^2 \setminus P^1} \mu_{e_r} + \sum_{e_r \in P^1 \cap P^2} \upsilon_{e_r} (R_{e_r}^2 - R_{e_r}^1) \leq C_j^2 - C_j^1.$$

Since $R_{e_r}^1 \leq R_{e_r}^2$, then $\sum_{e_r \in P^1 \cap P^2} \upsilon_{e_r} (R_{e_r}^2 - R_{e_r}^1) \geq 0$. Given that $\mu_j - \phi_j \upsilon_j \leq 0$, $\forall j \in V_i^1 \setminus V_i^2$, we obtain $\sum_{e_r \in P^2 \setminus P^1} \upsilon_{e_r} R_{e_r}^2 - \sum_{e_r \in P^2 \setminus P^1} \mu_{e_r} \geq 0$. Consequently, $C_j^1 \leq C_j^2$.

□

B.2. Proof of Lower Bound for the Reduced Cost (Proposition 2)

Assume a label $L_i = (C_i, V_i, R_i, w_i)$, with $C_i \geq 0$ and $w_i = i$. Let $S \subseteq \mathcal{V}^{\mathrm{r}} \setminus V_i$ be the set of all the non-visited damaged vertices j with $\mu_j > 0$. Since $\mu_j > 0$, a potential reduction in the reduced cost C_i can be obtained by visiting a damaged vertex j. We call this potential reduction "prize". Let $\phi_j = R_i + \tilde{t}_{ij} + \delta_j$ be the earliest possible time at which damaged vertex $j \in S$ can be repaired in any extension from label L_i. Therefore, we define $\texttt{prize}_j$ as an estimated potential reduction in cost C_i if damaged vertex j is repaired by the crew after damaged vertex i, which is calculated as follows:

$$\texttt{prize}_j = \max \Big\{ \mu_j - \phi_j \upsilon_j, 0 \Big\}.$$

Since vertex j may not be repaired immediately after vertex i and/or it may not be possible to arrive at vertex j using the shortest path (some vertices are still damaged), the actual time at which vertex j is repaired may be higher, i.e., $R_j \geq \phi_j$. Consequently, the actual prize to visit vertex j may be smaller than the estimated prize, i.e., $\mu_j - R_j \upsilon_j \leq \mu_j - \phi_j \upsilon_j$. Furthermore, if $\mu_j - \phi_j \upsilon_j \leq 0$, then $\mu_j - R_j \upsilon_j \leq 0$. Thus, $\texttt{prize}_j$ is an upper bound for the actual prize of visiting vertex j and $C_i - \sum_{j \in S} \texttt{prize}_j$ is a lower bound for the reduced cost of extensions derived from label L_i.

Consider the case in which $\sum_{j \in S} \texttt{prize}_j \leq C_i$. If additionally we have $\sum_{j \in S} \texttt{prize}_j = 0$, then it would not be possible to obtain a reduced cost $C_j < C_i$ for label extensions derived from L_i and thus we set $C_i^{\texttt{LB}} = C_i$ as a lower bound for the reduced cost. If $\sum_{j \in S} \texttt{prize}_j > 0$, it would not be possible to have a reduced cost $C_j < 0$ for label extensions derived from L_i and we set $C_i^{\texttt{LB}} = 0$ as a lower bound for the reduced cost.

Regarding the case with $\sum_{j \in S} \texttt{prize}_j > C_i$, let N be a lower bound on the number of damaged vertices that should be repaired after vertex i to find a route with a negative reduced cost. The value of N can be calculated based on the prizes. Let $S_p \subseteq S$ be a set containing the p vertices with the highest prizes $\texttt{prize}_j$. Then, N is equal to the smallest p such that $\sum_{j \in S_p} \texttt{prize}_j > C_i$. In this case, we can say that at least N

damaged vertices have to be repaired after vertex i to potentially find a reduced cost $C_j < 0$ in extensions derived from label L_i. Let l_n be the nth damaged vertex repaired after damaged vertex i. Then, a lower bound estimation for the restoration time of damaged vertex l_1 is $\tilde{R}_{l_1} = R_i + \tilde{t}_{i(l_1)} + \delta_{l_1}$, a lower bound estimation for the restoration time of damaged vertex l_2 is $\tilde{R}_{l_2} = \tilde{R}_{l_1} + \tilde{t}_{(l_1)(l_2)} + \delta_{l_2} = R_i + \tilde{t}_{i(l_1)} + \tilde{t}_{(l_1)(l_2)} + \delta_{l_1} + \delta_{l_2}$, and a estimation for the restoration time of damaged vertex l_n is

$$\tilde{R}_{l_n} = R_i + \tilde{t}_{i(l_1)} + \tilde{t}_{(l_1)(l_2)} + \ldots + \tilde{t}_{(l_{(n-1)})(l_n)} + \delta_{l_1} + \delta_{l_2} + \ldots + \delta_{l_n}.$$

If we consider $\tilde{\rho}_n = \tilde{t}_{(l_{(n-1)})(l_n)} + \delta_{l_n}$ and $\tilde{\rho}_n^{\texttt{rank}} = \sum_{i=2}^{i=n} \rho_i$, then we obtain

$$\tilde{R}_{l_n} = R_i + \tilde{t}_{i(l_1)} + \delta_{l_1} + \tilde{\rho}_{l_2} + \ldots + \tilde{\rho}_{l_n} = R_i + \tilde{t}_{i(l_1)} + \delta_{l_1} + \tilde{\rho}_n^{\texttt{rank}}.$$

Note that the first three terms $(R_i + \tilde{t}_{i(l_1)} + \delta_{l_1})$ are considered in the definition of prize ($\texttt{prize}_{l_1}$). An additional cost $\tilde{\rho}_n^{\texttt{rank}} \upsilon_{l_n}$ can be computed based on the last term $\tilde{\rho}_n^{\texttt{rank}}$. We call this cost "penalty". Although extensions derived from label L_i can consider different restoration orders, a lower bound for the penalties can be established using the $N-1$ lowest values of $\tilde{\rho}_n^{\texttt{rank}}$ and υ_n. In this context, the nth highest value of the $N-1$ lowest values of υ_n is multiplied by the nth lowest value of the $N-1$ lowest values of $\tilde{\rho}_n^{\texttt{rank}}$. Therefore, we define $\texttt{penalty}_n$ as an estimated penalty associated with the nth damaged vertex repaired after damaged vertex i, which is calculated as follows:

$$\texttt{penalty}_n = \rho_n^{\texttt{rank}} \upsilon_n^{\texttt{rank}},$$

where, given $\rho_j = \min_{l \in S}\{\tilde{t}_{lj}\} + \delta_j, \forall j \in S : l \neq j$, $\rho_n^{\texttt{rank}}$ is the sum of the $n-1$ smallest ρ_j values such that $j \in \mathcal{V}^{\texttt{r}} \setminus V_i$ for $n > 1$ and $\rho_n^{\texttt{rank}} = 0$ for $n = 1$. Additionally, $\upsilon_n^{\texttt{rank}}$ is the $(|S| - N + n - 1)$th highest υ_j value such that $j \in S$ for $n > 1$ and $\upsilon_n^{\texttt{rank}} = 0$ for $n = 1$. If $\sum_{j \in S} \texttt{prize}_j - \sum_{n=1}^{N} \texttt{penalty}_n \leq 0$, it would not be possible to have a reduced cost $C_j < C_i$ for label extensions derived from L_i, and consequently, $C_i^{\texttt{LB}} = C_i$ is used as a lower bound for the reduced cost. Otherwise, we use the lower bound $C_i^{\texttt{LB}} = C_i - \sum_{j \in S} \texttt{prize}_j + \sum_{n=1}^{N} \texttt{penalty}_n$.

□

Appendix C: Valid Inequalities

In this section, we present the VIs related to the relief path decisions proposed by Moreno et al. (2020). Multiple relief paths $0-i$ may be available to reach a demand vertex i from the depot 0. Let $\mathcal{P}_i^{\texttt{d}}$ be the set of possible $0-i$ relief paths. We call E_p and $\mathcal{V}_p$ as the set of edges and vertices used in path $p \in \mathcal{P}_i^{\texttt{d}}$. Similarly, $\mathcal{V}_p^{\texttt{r}}$ and $\mathcal{V}_p^{\texttt{u}}$ are the set of damaged and undamaged vertices used in path $p \in \mathcal{P}_i^{\texttt{d}}$. We define w_p as the sum of the length of the edges used in path p, i.e., $w_p = \sum_{e \in E_p} \ell_e$. We also define $\theta_{pi}^{\texttt{d}}$ as the accessibility time of the demand vertex i if path p is selected to connect the depot with the demand vertex i and $\theta_j^{\texttt{r}}$ as the restoration time of the damaged vertex j. Given two paths p, $p' \in \mathcal{P}_i^{\texttt{d}}$ such that $p \neq p'$, we say that p dominates p' if $\mathcal{V}_p^{\texttt{r}} \subseteq \mathcal{V}_{p'}^{\texttt{r}}$ and $w_p \leq l_i^{\texttt{d}}$. In this case, p' is a dominated path. We define $\mathcal{S}_i^{\texttt{d}} \subseteq \mathcal{P}_i^{\texttt{d}}$ as the set of nondominated paths from the depot to demand vertex i. Finally, let $\mathcal{P}_i^{\texttt{d}*} = \{p \in \mathcal{S}_i^{\texttt{d}} \,|\, \mathcal{V}_p^{\texttt{r}} = \emptyset\}$ be the set of nondominated

paths that do not visit any damaged vertex and p_i^* be an element of set $\mathcal{P}_i^{\mathrm{d}*}$. Using the notation above, we state the valid inequalities as follows:

$$|E_{p_i^*}| + |\mathcal{V}^{\mathrm{u}}_{p_i^*}| = \sum_{e \in E_{p_i^*}} Y_{ei} + \sum_{j \in \mathcal{V}^{\mathrm{u}}_{p_i^*}} V_{ji}, \forall\ i \in \mathcal{V}^{\mathrm{d}},\ p_i^* \in \mathcal{P}_i^{\mathrm{d}*} : \mathcal{P}_i^{\mathrm{d}*} \neq \emptyset, \tag{45}$$

$$(|E| + |\mathcal{V}|) \cdot (|\mathcal{V}_p^{\mathrm{r}}| - \sum_{j \in \mathcal{V}_p^{\mathrm{r}}} V_{ji}) \geq \sum_{e \in E \setminus E_p} Y_{ei} + \sum_{j \in \mathcal{V} \setminus \mathcal{V}_p} V_{ji}, \forall\ i \in \mathcal{V}^{\mathrm{d}}, p \in \mathcal{S}_i^{\mathrm{d}} : \mathcal{P}_i^{\mathrm{d}*} = \emptyset, \tag{46}$$

$$\sum_{j \in \mathcal{U}_i} V_{ji} \geq 1, \forall\ i \in \mathcal{V}^{\mathrm{d}} : \mathcal{P}_i^{\mathrm{d}*} = \emptyset, \tag{47}$$

$$Z_i^{\mathrm{d}} \geq \min_{j \in \mathcal{U}_i} (\rho^*_{0j} + \delta_j), \forall\ i \in \mathcal{V}^{\mathrm{d}} : \mathcal{P}_i^{\mathrm{d}*} = \emptyset, \tag{48}$$

$$\sum_{j \in n_i} V_{ji} = |n_i|, \forall\ i \in \mathcal{V}^{\mathrm{d}} : \mathcal{P}_i^{\mathrm{d}*} = \emptyset, \tag{49}$$

$$Z_i^{\mathrm{d}} \geq Z_j^{\mathrm{r}}, \forall\ j \in n_i, i \in \mathcal{V}^{\mathrm{d}} : \mathcal{P}_i^{\mathrm{d}*} = \emptyset, \tag{50}$$

$$Z_i^{\mathrm{d}} \geq \sum_{j \in \mathcal{V}^{\mathrm{r}}} \min_{\substack{l \in \mathcal{V}_0^{\mathrm{r}}: \\ l \neq j}} \left\{ \rho^*_{lj} + \delta_j \right\} \cdot V_{ji}, \forall\ i \in \mathcal{V}^{\mathrm{d}} : \mathcal{P}_i^{\mathrm{d}*} = \emptyset, \tag{51}$$

where ρ^*_{ij} is the shortest time for the crew to travel from vertex i to vertex j; $\mathcal{U}_i = \bigcup_{p \in \mathcal{S}_i^{\mathrm{d}}} \mathcal{V}_p^{\mathrm{r}}$ contains all the damaged vertices of the nondominated paths; and $n_i = \bigcap_{p \in \mathcal{S}_i^{\mathrm{d}}} \mathcal{V}_p^{\mathrm{r}}$ contains the damaged vertices that are used in all the nondominated paths.

Appendix D: Instance Description

The algorithms were tested using 648 benchmark instances from the literature (Maya-Duque, Dolinskaya, and Sörensen 2016, Moreno, Munari, and Alem 2019, Moreno et al. 2020). The instances can be found in Mendelay data (Moreno et al. 2022). Originally, these instances were derived from undamaged original networks by varying two parameters, namely, α and β. Parameter α defines the proportion of damaged edges in the network. Parameter β specifies the factor by which the distance between the depot and the demand vertices can increase in relation to the shortest distance, i.e., $l_i = (1+\beta) \cdot dist_{0i}$, where $dist_{0i}$ is the shortest distance between the depot and the demand vertex i. Table 8 shows the characteristics of the set of instances.

The instance set and the number of demand nodes, total nodes and arcs in the original networks are given in columns 1, 2, 3 and 4 of Table 8, respectively. Parameter α and β are shown in column 5 and 6, while the total numbers of nodes and arcs in the damaged networks can be seen in columns 7 and 8 of Table 8, respectively. The total number of instances (column 9 of Table 8) is generated by combining the values of α and β. By combining the values of α and β for the original network L1, for example, 20 instances were generated. For the original network L16, the values of $\alpha = 5\%$, 25%, and 50% are combined with $\beta = 5\%$ and 10% to form 6 instances, while the values of $\alpha = 10\%$ and 30% are combined with $\beta = 25\%$ and 50% to form 4 instances, totalling 10 instances. Instances L1-L39 were randomly generated by Maya-Duque, Dolinskaya, and Sörensen (2016) while instances CS1-CS6 where generated based on data from a real case disaster by Moreno et al. (2020). It is worth mentioning that some of the large instances are actually much larger than the practical instances we typically find in real-world situations. For example, Akbari and Salman (2017a,b) considered one of the largest practical cases in the literature, involving networks with 240 damaged points, 349 vertices and 689 edges. Note that we are considering instances with up to 312 damaged vertices, 712 total vertices and 937 total edges.

Table 8 Characteristics of the proposed set of instances.

Instance		Original network				Damaged network		Total
set	$\lvert\mathcal{V}^{\mathrm{d}}\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	Values for α (%)	Values for β (%)	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	instances
L1	19	25	40	5, 10, 25, 30, 50	5, 10, 25, 50	27 to 45	42 to 60	20
L2	19	25	37	5, 10, 25, 30, 50	5, 10, 25, 50	26 to 43	38 to 55	20
L3	19	25	39	5, 10, 25, 30, 50	5, 10, 25, 50	26 to 44	40 to 58	20
L4	24	30	83	5, 10, 25, 30, 50	5, 10, 25, 50	34 to 71	87 to 124	20
L5	24	30	89	5, 10, 25, 30, 50	5, 10, 25, 50	34 to 74	93 to 133	20
L6	24	30	84	5, 10, 25, 30, 50	5, 10, 25, 50	34 to 72	88 to 126	20
L7	28	35	118	5, 10, 25, 30, 50	5, 10, 25, 50	40 to 94	123 to 177	20
L8	28	35	115	5, 10, 25, 30, 50	5, 10, 25, 50	40 to 92	120 to 172	20
L9	28	35	113	5, 10, 25, 30, 50	5, 10, 25, 50	40 to 91	118 to 169	20
L10	15	20	39	5, 10, 25, 30, 50	5, 10, 25, 50	21 to 39	40 to 58	20
L11	15	20	37	5, 10, 25, 30, 50	5, 10, 25, 50	21 to 38	38 to 55	20
L12	15	20	37	5, 10, 25, 30, 50	5, 10, 25, 50	21 to 38	38 to 55	20
L13	35	40	146	5, 10, 25, 30, 50	5, 10, 25, 50	47 to 113	153 to 219	20
L14	35	40	143	5, 10, 25, 30, 50	5, 10, 25, 50	47 to 111	150 to 214	20
L15	35	40	143	5, 10, 25, 30, 50	5, 10, 25, 50	47 to 111	150 to 214	20
L16	50	60	191	5, 25, 50 | 10, 30	05, 10 | 25, 50	69 to 155	200 to 286	6 | 4
L17	50	60	197	5, 25, 50 | 10, 30	25, 50 | 05, 10	69 to 158	206 to 295	6 | 4
L18	50	60	196	5, 25, 50 | 10, 30	05, 10 | 25, 50	69 to 158	205 to 294	6 | 4
L19	70	80	247	5, 25, 50 | 10, 30	25, 50 | 05, 10	92 to 203	259 to 370	6 | 4
L20	70	80	245	5, 25, 50 | 10, 30	05, 10 | 25, 50	92 to 202	257 to 367	6 | 4
L21	70	80	248	5, 25, 50 | 10, 30	25, 50 | 05, 10	92 to 204	260 to 372	6 | 4
L22	90	100	274	5, 25, 50 | 10, 30	05, 10 | 25, 50	113 to 237	287 to 411	6 | 4
L23	90	100	271	5, 25, 50 | 10, 30	25, 50 | 05, 10	113 to 235	284 to 406	6 | 4
L24	90	100	273	5, 25, 50 | 10, 30	05, 10 | 25, 50	113 to 236	286 to 409	6 | 4
L25	125	140	324	5, 25, 50 | 10, 30	25, 50 | 05, 10	156 to 302	340 to 486	6 | 4
L26	125	140	323	5, 25, 50 | 10, 30	05, 10 | 25, 50	156 to 301	339 to 484	6 | 4
L27	125	140	322	5, 25, 50 | 10, 30	25, 50 | 05, 10	156 to 301	338 to 483	6 | 4
L28	140	170	398	5, 25, 50 | 10, 30	05, 10 | 25, 50	189 to 369	417 to 597	6 | 4
L29	140	170	399	5, 25, 50 | 10, 30	25, 50 | 05, 10	189 to 369	418 to 598	6 | 4
L30	140	170	396	5, 25, 50 | 10, 30	05, 10 | 25, 50	189 to 368	415 to 594	6 | 4
L31	200	200	447	5, 25, 50 | 10, 30	25, 50 | 05, 10	222 to 423	469 to 670	6 | 4
L32	200	200	449	5, 25, 50 | 10, 30	05, 10 | 25, 50	222 to 424	471 to 673	6 | 4
L33	200	200	449	5, 25, 50 | 10, 30	25, 50 | 05, 10	222 to 424	471 to 673	6 | 4
L34	300	300	524	5, 25, 50 | 10, 30	05, 10 | 25, 50	326 to 562	550 to 786	6 | 4
L35	300	300	525	5, 25, 50 | 10, 30	25, 50 | 05, 10	326 to 562	551 to 787	6 | 4
L36	300	300	525	5, 25, 50 | 10, 30	05, 10 | 25, 50	326 to 562	551 to 787	6 | 4
L37	400	400	625	5, 25, 50 | 10, 30	25, 50 | 05, 10	431 to 712	656 to 937	6 | 4
L38	400	400	625	5, 25, 50 | 10, 30	05, 10 | 25, 50	431 to 712	656 to 937	6 | 4
L39	400	400	625	5, 25, 50 | 10, 30	25, 50 | 05, 10	431 to 712	656 to 937	6 | 4
CS1	13	60	89	5, 10, 15	5, 10, 25, 50, 100, ∞	66 to 74	95 to 103	18
CS2	13	60	89	5, 10, 15	5, 10, 25, 50, 100, ∞	66 to 74	95 to 103	18
CS3	13	60	89	5, 10, 15	5, 10, 25, 50, 100, ∞	66 to 74	95 to 103	18
CS4	20	60	89	5, 10, 15	5, 10, 25, 50, 100, ∞	66 to 74	95 to 103	18
CS5	20	60	89	5, 10, 15	5, 10, 25, 50, 100, ∞	66 to 74	95 to 103	18
CS6	20	60	89	5, 10, 15	5, 10, 25, 50, 100, ∞	66 to 74	95 to 103	18
Total								648

Appendix E: Additional Results

In this section, additional computational results are presented for the proposed CG and BP strategies. Table 9 shows the improvements of the CG approaches in relation to LPR. The first column indicates the instance sets, while the second column shows the LP bound obtained with LPR (LPR obj. value). The following columns indicate, for the different CG strategies, the average LP bound (obj. value) and the improvement in relation to the values obtained by LPR. Note that higher improvements are observed in larger instances.

Table 9 Lower bound improvements of the CG strategies for the different instances sets.

Instance	LPR	CG1		CG2		CG3	
sets	Obj. value	Obj. value	%imp.*	Obj. value	%imp.*	Obj. value	%imp.*
L1	6,670	8,858	24.70	8,309	19.73	8,862	24.74
L2	17,952	24,205	25.83	24,057	25.38	28,737	37.53
L3	27,250	46,105	40.90	46,105	40.90	46,867	41.86
L4	9,835	16,382	39.97	15,976	38.44	16,441	40.18
L5	7,862	16,146	51.30	15,791	50.21	16,217	51.52
L6	10,343	14,726	29.76	14,160	26.95	15,883	34.88
L7	16,694	30,743	45.70	28,285	40.98	31,463	46.94
L8	12,376	18,091	31.59	16,958	27.02	19,851	37.66
L9	12,428	21,861	43.15	22,452	44.65	22,723	45.31
L10	34,558	43,604	20.75	40,729	15.15	45,215	23.57
L11	20,004	31,646	36.79	28,652	30.19	33,002	39.39
L12	18,407	24,630	25.27	21,806	15.59	25,801	28.66
L13	14,432	20,495	29.58	19,073	24.33	21,094	31.58
L14	35,317	51,953	32.02	49,682	28.91	51,012	30.77
L15	18,310	33,344	45.09	33,183	44.82	33,483	45.31
L16	20,481	22,663	9.63	22,029	7.03	22,876	10.47
L17	13,484	19,825	31.98	21,945	38.55	20,508	34.25
L18	55,252	61,167	9.67	63,028	12.34	61,856	10.68
L19	16,104	20,969	23.20	18,904	14.81	20,295	20.65
L20	11,535	20,208	42.92	19,758	41.62	20,401	43.46
L21	14,222	19,412	26.73	19,047	25.33	19,672	27.70
L22	55,957	63,159	11.40	59,049	5.24	60,789	7.95
L23	18,984	22,526	15.72	21,752	12.72	23,153	18.01
L24	17,735	27,599	35.74	26,007	31.81	26,705	33.59
L25	11,170	17,950	37.77	18,922	40.97	17,980	37.87
L26	12,537	26,593	52.85	25,656	51.13	26,240	52.22
L27	15,182	22,804	33.43	21,580	29.65	19,402	21.75
L28	19,988	25,854	22.69	31,903	37.35	37,578	46.81
L29	12,335	20,824	40.77	20,114	38.68	20,424	39.61
L30	8,993	24,111	62.70	23,074	61.02	23,104	61.07
L31	7,159	15,801	54.69	15,742	54.52	15,802	54.69
L32	2,995	15,450	80.62	15,350	80.49	15,497	80.68
L33	5,072	14,758	65.63	14,613	65.29	14,758	65.63
L34	5,046	23,724	78.73	23,720	78.73	23,724	78.73
L35	3,674	19,358	81.02	19,117	80.78	19,359	81.02
L36	4,256	20,418	79.15	20,299	79.03	20,418	79.15
L37	4,237	17,232	75.41	20,637	79.47	20,657	79.49
L38	5,083	26,988	81.17	26,973	81.16	26,988	81.17
L39	3,267	17,609	81.45	17,609	81.45	18,397	82.24
CS1	53,245	43,310	-22.94	59,639	10.72	77,807	31.57
CS2	82,862	120,604	31.29	116,445	28.84	124,606	33.50
CS3	62,742	84,684	25.91	66,635	5.84	87,387	28.20
CS4	60,607	110,766	45.28	103,820	41.62	131,303	53.84
CS5	93,107	144,551	35.59	138,679	32.86	149,897	37.89
CS6	82,745	128,178	35.45	102,794	19.50	133,734	38.13
Avg. L1-L15	17,496	26,853	34.83	25,681	31.55	27,777	37.33
Avg. L16-L39	14,365	24,458	47.29	24,451	47.05	24,858	47.87
Avg. CS1-CS6	72,551	105,349	25.10	98,002	23.23	117,456	37.19

* $\%\text{imp} = \frac{(\text{Obj. value in CG} - \text{Obj. value in LPR})}{\text{Obj. value in LPR}}(\%)$

Tables 10 to 12 show the improvement of BP1 (BP1H), BP2 (BP2H), and BP3 (BP3H), concerning gap and upper bound for the different instance sets.

Table 10 Improvement of BP1 and BP1H concerning gap and upper bound for the different instance sets.

Instance sets	LM[1]		BP1								BP1H			
	Avg. UB	Avg. Gap	Avg. UB	Avg. Gap	Avg. time (seconds)	%UB imp[2]	UB ratio[2]	%Gap imp[2]	Gap diff[2]	Gap ratio[2]	Avg. UB	Avg. time (seconds)	%UB imp[2]	UB ratio[2]
L1	9,745	0.58	9,745	0.00	160	0.00	0.00	15.00	0.58	100.00	9,745	8	0.00	0.00
L2	34,089	2.10	34,089	1.53	869	5.00	0.00	10.00	0.58	27.41	34,089	779	5.00	0.00
L3	49,862	0.37	49,862	1.21	278	0.00	0.00	0.00	-0.84	-230.79	49,862	281	0.00	0.00
L4	18,031	2.60	18,027	1.90	467	5.00	0.02	10.00	0.69	26.70	18,027	206	5.00	0.02
L5	18,074	5.65	18,074	4.23	733	0.00	0.00	20.00	1.42	25.08	18,074	196	0.00	0.00
L6	20,917	7.58	20,917	6.77	727	0.00	0.00	30.00	0.81	10.74	20,917	409	0.00	0.00
L7	36,511	4.03	36,511	5.86	929	0.00	0.00	40.00	-1.83	-45.34	36,511	835	0.00	0.00
L8	26,021	9.18	26,636	6.62	857	0.00	-2.37	25.00	2.56	27.85	26,021	783	0.00	0.00
L9	33,903	15.25	34,220	7.84	908	0.00	-0.94	40.00	7.41	48.60	33,903	402	0.00	0.00
L10	48,460	2.12	48,460	3.88	1,478	0.00	0.00	10.00	-1.76	-82.81	48,460	1,177	0.00	0.00
L11	38,538	3.52	38,538	0.52	501	0.00	0.00	20.00	3.00	85.23	38,538	414	0.00	0.00
L12	28,037	0.66	28,037	0.40	248	15.00	0.00	10.00	0.26	39.29	28,037	81	15.00	0.00
L13	23,528	4.29	23,546	4.10	738	0.00	-0.08	15.00	0.19	4.48	23,528	581	0.00	0.00
L14	80,975	18.02	81,007	15.58	2,162	0.00	-0.04	35.00	2.44	13.56	80,631	1,438	10.00	0.43
L15	51,385	15.85	51,397	5.13	1,225	0.00	-0.02	55.00	10.72	67.66	51,385	227	0.00	0.00
L16	38,737	11.68	38,737	14.72	2,163	0.00	0.00	20.00	-3.04	-26.04	38,737	1,835	0.00	0.00
L17	30,448	14.06	30,475	6.01	826	0.00	-0.09	50.00	8.04	57.22	30,448	884	0.00	0.00
L18	94,580	18.90	94,754	16.99	2,522	0.00	-0.18	50.00	1.91	10.11	94,580	2,543	0.00	0.00
L19	45,802	29.14	45,811	16.22	2,167	0.00	-0.02	40.00	12.91	44.33	45,802	1,524	0.00	0.00
L20	41,776	29.95	41,780	21.07	1,820	10.00	-0.01	40.00	8.87	29.63	41,781	1,128	10.00	-0.01
L21	58,692	36.72	59,119	41.53	2,167	0.00	-0.73	20.00	-4.81	-13.09	58,727	2,187	0.00	-0.06
L22	145,789	33.24	149,042	32.54	2,883	0.00	-2.23	30.00	0.71	2.12	145,923	2,899	0.00	-0.09
L23	57,415	29.46	57,636	30.41	2,834	0.00	-0.39	30.00	-0.95	-3.22	57,415	2,190	0.00	0.00
L24	93,233	38.43	93,541	37.89	2,173	0.00	-0.33	40.00	0.54	1.41	93,435	1,034	0.00	-0.22
L25	72,577	40.81	73,082	37.34	2,190	0.00	-0.70	50.00	3.48	8.52	72,726	1,696	0.00	-0.21
L26	111,583	46.08	111,585	43.00	3,135	0.00	0.00	50.00	3.08	6.69	111,583	2,650	0.00	0.00
L27	69,320	41.72	69,320	35.86	2,198	0.00	0.00	50.00	5.85	14.03	69,322	2,217	0.00	0.00
L28	175,526	41.63	175,573	36.43	2,709	0.00	-0.03	40.00	5.20	12.50	175,534	2,854	0.00	0.00
L29	87,407	57.33	87,752	46.40	2,893	0.00	-0.40	50.00	10.94	19.08	87,317	1,992	10.00	0.10
L30	122,538	46.89	123,305	43.65	2,209	0.00	-0.63	50.00	3.24	6.90	122,607	2,208	0.00	-0.06
L31	130,166	76.33	123,383	52.01	2,898	40.00	5.21	100.00	24.33	31.87	123,196	1,028	50.00	5.35
L32	128,151	72.48	119,672	52.04	2,238	50.00	6.62	90.00	20.44	28.20	116,607	1,243	50.00	9.01
L33	96,869	68.23	93,037	51.09	2,731	30.00	3.96	80.00	17.15	25.13	89,218	1,258	50.00	7.90
L34	279,824	87.75	268,036	61.48	3,126	50.00	4.21	100.00	26.27	29.94	267,066	2,288	60.00	4.56
L35	254,391	86.08	252,855	69.24	3,369	10.00	0.60	100.00	16.84	19.56	247,678	2,360	60.00	2.64
L36	283,101	86.97	275,452	69.81	2,926	30.00	2.70	100.00	17.17	19.74	264,519	1,430	60.00	6.56
L37	393,965	90.85	364,153	71.18	3,600	40.00	7.57	100.00	19.67	21.65	347,717	2,183	60.00	11.74
L38	546,699	90.68	514,809	77.25	3,600	40.00	5.83	100.00	13.43	14.81	457,720	2,250	80.00	16.28
L39	441,764	88.61	326,663	70.94	3,001	60.00	26.05	100.00	17.67	19.94	330,628	2,107	60.00	25.16
CS1	84,656	3.56	84,656	3.01	2,543	0.00	0.00	5.56	0.55	15.42	84,596	2,874	11.11	0.07
CS2	134,255	1.49	134,255	0.82	5	0.00	0.00	11.11	0.66	44.64	134,255	1,024	0.00	0.00
CS3	99,514	2.46	99,514	1.29	617	0.00	0.00	11.11	1.17	47.55	99,512	1,494	5.56	0.00
CS4	145,455	8.48	145,455	7.46	2,918	0.00	0.00	11.11	1.02	12.02	143,739	2,871	22.22	1.18
CS5	166,114	3.78	166,114	2.16	87	0.00	0.00	11.11	1.62	42.76	166,114	90	0.00	0.00
CS6	163,644	6.55	163,644	5.15	739	0.00	0.00	16.67	1.40	21.34	163,644	840	0.00	0.00
Avg. L1-L15	34,538	6.12	34,604	4.37	819	1.67	-0.23	22.33	1.75	7.85	34,515	521	2.33	0.03
Avg. L16-L39	158,348	52.67	149,566	43.13	2,599	15.00	2.38	61.67	9.54	15.88	145,429	1,916	22.92	3.69
Avg. CS1-CS6	132,273	4.39	132,273	3.32	1,151	0.00	0.00	11.11	1.07	30.62	131,977	1,532	6.48	0.21

[1] LM: literature method. Best solutions found in the literature for the tested instances (Moreno et al. 2020).
[2] %UB imp: percentage of instances for which UB in LM > UB in BP1; %Gap imp: percentage of instances for which Gap in LM > Gap in BP1; Gap diff = Gap in LM − Gap in BP1; Gap (UB) ratio = $\frac{\text{Gap (UB) in LM} - \text{Gap (UB) in BP1}}{\text{Gap (UB) in LM}}$ (%).

Table 11 Improvement of BP2 and BP2H concerning gap and upper bound for the different instance sets.

Instance sets	LM[1]		BP2								BP2H			
	Avg. UB	Avg. Gap	Avg. UB	Avg. Gap	Avg. time (seconds)	%UB imp[2]	UB ratio[2]	%Gap imp[2]	Gap diff[2]	Gap ratio[2]	Avg. UB	Avg. time (seconds)	%UB imp[2]	UB ratio[2]
L1	9,745	0.58	9,745	0.15	371	0.00	0.00	10.00	0.43	74.28	9,745	12	0.00	0.00
L2	34,089	2.10	34,089	0.83	537	5.00	0.00	10.00	1.27	60.38	34,089	143	5.00	0.00
L3	49,862	0.37	49,862	0.00	35	0.00	0.00	5.00	0.37	100.00	49,862	29	0.00	0.00
L4	18,031	2.60	18,027	1.64	545	5.00	0.02	15.00	0.96	36.97	18,027	10	5.00	0.02
L5	18,074	5.65	18,074	0.86	429	0.00	0.00	30.00	4.79	84.72	18,074	13	0.00	0.00
L6	20,917	7.58	20,917	3.29	386	0.00	0.00	40.00	4.29	56.60	20,917	30	0.00	0.00
L7	36,511	4.03	36,511	4.62	733	0.00	0.00	45.00	-0.59	-14.52	36,511	425	0.00	0.00
L8	26,021	9.18	26,022	5.70	767	0.00	-0.01	25.00	3.47	37.84	26,021	611	0.00	0.00
L9	33,903	15.25	33,903	7.59	730	0.00	0.00	40.00	7.66	50.21	34,024	404	0.00	-0.36
L10	48,460	2.12	48,460	1.51	632	0.00	0.00	10.00	0.61	28.85	48,460	264	0.00	0.00
L11	38,538	3.52	38,538	1.45	562	0.00	0.00	15.00	2.07	58.87	38,538	256	0.00	0.00
L12	28,037	0.66	28,037	0.40	286	15.00	0.00	10.00	0.26	39.29	28,037	277	15.00	0.00
L13	23,528	4.29	23,528	4.05	858	0.00	0.00	15.00	0.24	5.63	23,529	249	0.00	0.00
L14	80,975	18.02	81,076	15.58	2,162	0.00	-0.12	35.00	2.44	13.51	81,050	900	0.00	-0.09
L15	51,385	15.85	51,402	5.20	1,192	0.00	-0.03	55.00	10.65	67.20	51,392	50	0.00	-0.01
L16	38,737	11.68	38,737	7.29	749	0.00	0.00	40.00	4.39	37.60	38,737	785	0.00	0.00
L17	30,448	14.06	30,448	5.97	777	0.00	0.00	60.00	8.09	57.55	30,448	831	0.00	0.00
L18	94,580	18.90	95,095	16.75	2,168	0.00	-0.54	50.00	2.15	11.39	95,526	1,634	0.00	-1.00
L19	45,802	29.14	45,840	16.23	2,168	0.00	-0.08	40.00	12.90	44.29	45,818	1,063	0.00	-0.04
L20	41,776	29.95	41,828	21.08	1,822	10.00	-0.12	40.00	8.86	29.60	41,728	278	20.00	0.12
L21	58,692	36.72	59,007	41.38	2,166	0.00	-0.54	20.00	-4.65	-12.67	58,831	740	0.00	-0.24
L22	145,789	33.24	147,844	30.34	2,189	0.00	-1.41	50.00	2.90	8.72	146,986	1,806	0.00	-0.82
L23	57,415	29.46	57,380	32.47	2,883	10.00	0.06	20.00	-3.01	-10.23	57,509	2,999	0.00	-0.16
L24	93,233	38.43	94,063	37.92	2,176	0.00	-0.89	40.00	0.51	1.31	94,186	437	0.00	-1.02
L25	72,577	40.81	72,939	36.77	2,187	0.00	-0.50	60.00	4.04	9.91	73,006	1,342	10.00	-0.59
L26	111,583	46.08	111,610	41.62	2,187	0.00	-0.02	60.00	4.45	9.67	111,614	661	0.00	-0.03
L27	69,320	41.72	69,317	35.86	2,188	20.00	0.00	50.00	5.85	14.03	69,336	1,940	0.00	-0.02
L28	175,526	41.63	175,627	37.63	2,603	10.00	-0.06	40.00	4.00	9.60	175,818	1,439	0.00	-0.17
L29	87,407	57.33	87,425	46.09	2,761	10.00	-0.02	50.00	11.24	19.61	87,504	1,313	10.00	-0.11
L30	122,538	46.89	122,720	43.47	2,207	0.00	-0.15	50.00	3.42	7.30	122,585	1,110	0.00	-0.04
L31	130,166	76.33	120,003	51.94	2,899	50.00	7.81	100.00	24.40	31.96	118,120	685	50.00	9.25
L32	128,151	72.48	117,090	51.97	2,230	60.00	8.63	90.00	20.51	28.29	116,564	696	50.00	9.04
L33	96,869	68.23	88,805	50.64	2,896	40.00	8.32	90.00	17.60	25.79	91,574	631	50.00	5.47
L34	279,824	87.75	264,924	61.53	2,994	30.00	5.32	100.00	26.22	29.88	268,042	1,128	50.00	4.21
L35	254,391	86.08	241,378	69.19	2,945	50.00	5.12	100.00	16.89	19.62	239,871	1,118	60.00	5.71
L36	283,101	86.97	284,765	69.85	2,925	30.00	-0.59	100.00	17.12	19.69	272,396	1,157	50.00	3.78
L37	393,965	90.85	345,522	71.20	3,600	60.00	12.30	100.00	19.64	21.62	352,782	1,515	60.00	10.45
L38	546,699	90.68	482,535	77.44	3,600	40.00	11.74	100.00	13.24	14.60	474,262	1,512	60.00	13.25
L39	441,764	88.61	331,426	70.92	2,993	60.00	24.98	100.00	17.69	19.96	330,638	1,400	60.00	25.15
CS1	84,656	3.56	84,614	0.52	458	5.56	0.05	22.22	3.03	85.26	84,596	480	11.11	0.07
CS2	134,255	1.49	134,255	0.00	222	0.00	0.00	22.22	1.49	100.00	134,255	45	0.00	0.00
CS3	99,514	2.46	99,512	0.00	1,168	5.56	0.00	44.44	2.46	100.00	99,512	1,108	5.56	0.00
CS4	145,455	8.48	143,761	2.55	942	16.67	1.16	38.89	5.93	69.94	143,761	859	16.67	1.16
CS5	166,114	3.78	166,114	0.00	320	0.00	0.00	33.33	3.78	100.00	166,114	135	0.00	0.00
CS6	163,644	6.55	163,644	1.16	1,924	0.00	0.00	33.33	5.39	82.24	163,644	1,891	0.00	0.00
Avg. L1-L15	34,538	6.12	34,546	3.52	682	1.67	-0.01	24.00	2.59	46.66	34,552	245	1.67	-0.03
Avg. L16-L39	158,348	52.67	146,930	42.73	2,430	20.00	3.31	64.58	9.94	18.71	146,412	1,176	22.08	3.42
Avg. CS1-CS6	132,273	4.39	131,983	0.71	839	4.63	0.20	32.41	3.68	89.57	131,980	753	5.56	0.21

[1] LM: literature method. Best solutions found in the literature for the tested instances (Moreno et al. 2020).
[2] %UB imp: percentage of instances for which UB in LM > UB in BP2; %Gap imp: percentage of instances for which Gap in LM > Gap in BP2; Gap diff = Gap in LM − Gap in BP2; Gap (UB) ratio = $\frac{\text{Gap (UB) in LM} - \text{Gap (UB) in BP2}}{\text{Gap (UB) in LM}}$(%).

Table 12 Improvement of BP3 and BP3H concerning gap and upper bound for the different instance sets.

Instance sets	LM[1]		BP3								BP3H			
	Avg. UB	Avg. Gap	Avg. UB	Avg. Gap	Avg. time (seconds)	%UB imp[2]	UB ratio[2]	%Gap imp[2]	Gap diff[2]	Gap ratio[2]	Avg. UB	Avg. time (seconds)	%UB imp[2]	UB ratio[2]
L1	9,745	0.58	9,745	0.15	289	0.00	0.00	15.00	0.43	74.28	9,745	21	0.00	0.00
L2	34,089	2.10	34,089	0.00	210	5.00	0.00	10.00	2.10	100.00	34,089	318	5.00	0.00
L3	49,862	0.37	49,862	0.00	22	0.00	0.00	5.00	0.37	100.00	49,862	23	0.00	0.00
L4	18,031	2.60	18,027	0.00	235	5.00	0.02	20.00	2.60	100.00	18,027	18	5.00	0.02
L5	18,074	5.65	18,074	0.43	354	0.00	0.00	30.00	5.22	92.41	18,074	16	0.00	0.00
L6	20,917	7.58	20,917	3.29	469	0.00	0.00	40.00	4.29	56.60	20,917	45	0.00	0.00
L7	36,511	4.03	36,511	4.62	727	0.00	0.00	45.00	-0.59	-14.52	36,511	460	0.00	0.00
L8	26,021	9.18	26,021	5.70	727	0.00	0.00	25.00	3.47	37.85	26,021	429	0.00	0.00
L9	33,903	15.25	33,903	7.59	905	0.00	0.00	40.00	7.66	50.21	33,903	387	0.00	0.00
L10	48,460	2.12	48,460	0.76	201	0.00	0.00	15.00	1.36	64.37	48,460	205	0.00	0.00
L11	38,538	3.52	38,538	0.93	198	0.00	0.00	15.00	2.59	73.63	38,538	238	0.00	0.00
L12	28,037	0.66	28,037	0.40	189	15.00	0.00	10.00	0.26	39.29	28,037	202	15.00	0.00
L13	23,528	4.29	23,528	4.05	760	0.00	0.00	15.00	0.24	5.63	23,528	233	0.00	0.00
L14	80,975	18.02	81,017	15.57	2,162	0.00	-0.05	35.00	2.45	13.59	81,002	676	0.00	-0.03
L15	51,385	15.85	51,397	5.19	1,194	0.00	-0.02	55.00	10.66	67.22	51,385	44	0.00	0.00
L16	38,737	11.68	38,737	7.29	1,168	0.00	0.00	40.00	4.39	37.60	38,737	780	0.00	0.00
L17	30,448	14.06	30,476	5.99	766	0.00	-0.09	60.00	8.07	57.37	30,448	834	0.00	0.00
L18	94,580	18.90	94,667	16.65	2,163	0.00	-0.09	50.00	2.25	11.91	94,589	1,611	0.00	-0.01
L19	45,802	29.14	45,802	16.21	2,168	0.00	0.00	40.00	12.92	44.36	46,209	784	0.00	-0.89
L20	41,776	29.95	41,781	21.07	1,822	10.00	-0.01	40.00	8.87	29.63	41,836	289	0.00	-0.14
L21	58,692	36.72	58,765	41.30	2,166	0.00	-0.12	20.00	-4.57	-12.45	58,934	1,113	0.00	-0.41
L22	145,789	33.24	145,923	30.15	2,606	10.00	-0.09	50.00	3.09	9.29	146,787	1,904	0.00	-0.68
L23	57,415	29.46	57,415	27.78	2,884	0.00	0.00	20.00	1.68	5.69	57,566	2,661	0.00	-0.26
L24	93,233	38.43	93,412	38.04	2,174	0.00	-0.19	40.00	0.39	1.02	93,846	391	0.00	-0.66
L25	72,577	40.81	73,715	36.84	2,192	0.00	-1.57	60.00	3.98	9.75	73,187	1,275	0.00	-0.84
L26	111,583	46.08	111,710	41.87	2,191	0.00	-0.11	60.00	4.21	9.13	111,589	590	0.00	-0.01
L27	69,320	41.72	69,342	35.89	1,290	0.00	-0.03	50.00	5.83	13.98	69,320	1,927	0.00	0.00
L28	175,526	41.63	176,425	35.62	2,219	0.00	-0.51	50.00	6.01	14.44	175,613	1,356	0.00	-0.05
L29	87,407	57.33	87,500	46.26	2,892	10.00	-0.11	50.00	11.07	19.32	87,437	480	10.00	-0.03
L30	122,538	46.89	122,615	43.45	2,203	0.00	-0.06	50.00	3.45	7.35	122,782	1,021	0.00	-0.20
L31	130,166	76.33	124,848	52.02	2,899	40.00	4.09	100.00	24.31	31.85	126,716	668	40.00	2.65
L32	128,151	72.48	119,797	52.04	2,230	50.00	6.52	90.00	20.44	28.20	120,039	578	50.00	6.33
L33	96,869	68.23	96,281	51.41	2,896	20.00	0.61	90.00	16.82	24.65	91,001	640	40.00	6.06
L34	279,824	87.75	266,167	60.04	2,966	50.00	4.88	100.00	27.71	31.58	261,542	1,291	50.00	6.53
L35	254,391	86.08	243,900	67.62	2,946	60.00	4.12	100.00	18.46	21.44	243,955	1,093	60.00	4.10
L36	283,101	86.97	273,684	69.80	2,932	40.00	3.33	100.00	17.17	19.74	266,028	1,116	60.00	6.03
L37	393,965	90.85	359,226	66.18	3,600	60.00	8.82	100.00	24.66	27.15	347,436	1,425	60.00	11.81
L38	546,699	90.68	499,140	68.34	3,600	30.00	8.70	100.00	22.34	24.63	469,915	1,548	70.00	14.05
L39	441,764	88.61	334,612	71.01	2,988	50.00	24.26	100.00	17.60	19.87	319,100	1,373	60.00	27.77
CS1	84,656	3.56	84,596	0.00	263	11.11	0.07	33.33	3.56	100.00	84,596	296	11.11	0.07
CS2	134,255	1.49	134,255	0.00	209	0.00	0.00	22.22	1.49	100.00	134,255	59	0.00	0.00
CS3	99,514	2.46	99,512	0.00	444	5.56	0.00	44.44	2.46	100.00	99,512	1,208	5.56	0.00
CS4	145,455	8.48	143,739	0.06	394	22.22	1.18	50.00	8.43	99.32	143,739	582	22.22	1.18
CS5	166,114	3.78	166,114	0.00	37	0.00	0.00	33.33	3.78	100.00	166,114	83	0.00	0.00
CS6	163,644	6.55	163,644	1.16	1,058	0.00	0.00	38.89	5.39	82.25	163,644	1,840	0.00	0.00
Avg. L1-L15	34,538	6.12	34,542	3.24	576	1.67	0.00	25.00	2.87	57.37	34,540	221	1.67	0.00
Avg. L16-L39	158,348	52.67	148,581	41.79	2,415	17.92	2.60	65.00	10.88	20.31	145,609	1,114	20.83	3.38
Avg. CS1-CS6	132,273	4.39	131,977	0.20	401	6.48	0.21	37.04	4.18	96.93	131,977	678	6.48	0.21

[1] LM: literature method. Best solutions found in the literature for the tested instances (Moreno et al. 2020).
[2] %UB imp: percentage of instances for which UB in LM > UB in BP3; %Gap imp: percentage of instances for which Gap in LM > Gap in BP3; Gap diff = Gap in LM − Gap in BP3; Gap (UB) ratio = $\frac{\text{Gap (UB) in LM} - \text{Gap (UB) in BP3}}{\text{Gap (UB) in LM}}$(%).

Appendix F: Sensitivity Analysis on the Improvement Strategies

We present the results of experiments that verify the impact of different improvement strategies to the performance of CG3, the column generation algorithm of BP3. Table 13 summarizes the results of six alternative configurations of CG3 in which we turn off one improvement strategy at a time. For each configuration, the table presents the average computation time (Avg. time) as well as the number and percentage (#Opt, %Opt) of instances solved to optimality within the 1-hour time limit. According to the results, all the improvement strategies considered in Table 13 significantly affect the performance of CG3.

Table 13 Average computational results of CG3 without some improvement strategies.

	CG3	CG3-S1	CG3-S2	CG3-S3	CG3-S4	CG3-S5	CG3-S6
Avg. time (seconds)	1,348	2,258	1,611	3,257	2,044	1,804	3,583
#Opt	504	244	453	260	326	413	35
#%Opt	77.78	37.65	69.91	40.12	50.31	63.73	5.40

Configuration CG3-S1 turns off the use of vertice priorities in the relief paths and crew routes (see Section 5.7.1). In this case, the computation time was 67% higher. The reason for this increase is that many routes generated from SP1 are infeasible for the original SCSRP problem. CG3-S2 turns off the SA metaheuristic to generate initial columns in the MP (see Section 5.7.5). This configuration resorts to a simple construction heuristic (Moreno, Munari, and Alem 2019) to generate initial columns. The computation time increases by 19% and the LP relaxation was solved to optimality for 51 less instances when compared to CG3. CG3-S3 do not use rSP1, only SP1. As a result, the CG algorithm becomes slower since SP1 is more difficult to solve than rSP1, and it is solved in all iterations to find columns with negative reduced cost.

CG3-S4, CG3-S5, and CG3-S6 are directly related with the performance of the labeling algorithm to solve SP1 and SP1r, regarding the use of improved lower bounds and dominance rules (see Sections 5.4.3 and 5.4.4). CG3-S4 does not use the proposed lower bounds for the reduced cost, whereas CG3-S5 turns off the strong dominance rules by considering $V_i^1 = V_i^2$ in condition (ii) of Proposition 1. Finally, CG3-S6 does not consider any of these two improvements. CG3-S4 proved optimality only for 50.31% of the instances, while CG3-S5 proved optimality for 63.73%. The average computation time in both cases is greater than the one in CG3. In CG3-S6, the computation time increases by 164%, while optimality was only proved for 5.4% of the instances, compared to 77.78% in CG3. Overall, these strategies show a slower labeling algorithm that is inefficient at eliminating dominated states.