Cadernos do LOGIS

Exact Methods for Two Vehicle Routing Problems with Lockers in Last-Mile Delivery

Bruno Davi Mattos de Oliveira, Diogo Lima, Artur Pessoa and Marcos Roboredo

Volume 2024, Number 5

October, 2024

Preprint submitted to OR Spectrum

LOGIS
Núcleo de Logística Integrada e Sistemas

Universidade Federal Fluminense

Exact Methods for Two Vehicle Routing Problems with Lockers in Last-Mile Delivery

Bruno Davi Mattos de Oliveira[a], Diogo Lima[b], Artur Pessoa[b], Marcos Roboredo[b,*]

[a] *Universidade Federal Fluminense - Pós-Graduação Engenharia de Produção*
Niterói - Brasil
[b] *Universidade Federal Fluminense - Departamento de Engenharia de Produção*
Niterói - Brasil

Abstract

Recent variants of Vehicle Routing Problems incorporate strategic delivery locations known as lockers, which can receive demands from various customers. In addition to the routing aspect, these problems consider the assignment of customers to lockers rather than requiring direct visits, leading to assignment costs associated with last-mile trips. The objective in these variants is to define routes and assignments so that every customer is either served directly at home or assigned to a locker, with the aim of minimizing the total traversal and assignment costs. This study examines two variants prevalent in the literature: the Vehicle Routing Problem with Lockers and Time Windows (VRPLTW) and the Vehicle Routing Problem with Private and Shared Delivery Locations (VRPPSDL). These problems differ in their consideration of factors such as time windows, vehicle capacity, and constant assignment costs. For VRPLTW, we propose and compare two Branch-and-Cut-and-Price (BCP) algorithms implemented within the VRPSolver framework. For VRPPSDL, we develop a specialized BCP algorithm based on the one proposed for VRPLTW. Computational experiments demonstrate the robustness of our procedures, which outperform the best exact methods available in the literature and achieve optimal solutions for several instances for the first time for both addressed problems.

Keywords: Vehicle Routing Problem, Lockers, Branch-and-cut-and-price

1. Introduction

Since Dantzig and Ramser (1959), Vehicle Routing Problems (VRPs), have stood out among practical applications of Operations Research, involving various challenges in the daily activities of organizations. Frequently, practical characteristics of routing problems are incorporated into models so that they address real aspects faced by managers and decision-makers. For example,

*Corresponding author.
Email addresses: bruno_mattos@id.uff.br (Bruno Davi Mattos de Oliveira), diogofls@id.uff.br (Diogo Lima), arturpessoa@id.uff.br (Artur Pessoa), mcroboredo@id.uff.br (Marcos Roboredo)

specific routing problems include heterogeneous fleets, time windows, hotel selection, and vehicle refueling stops.

A class of VRPs that has attracted the attention of researchers in recent years is the one involving so-called lockers, which are special structures, strategically located, where the demands of different customers are stored. Customers then travel to the locker where their demand has been stored for pickup, generating an assignment cost. Thus, in VRPs with lockers, the routes of each vehicle must be defined, as well as decisions on which customers are served directly and which are assigned to a locker so that the traversal cost combined with the assignment cost is minimized. The literature include studies proposed by Grabenschweiger et al. (2021), Vincent et al. (2022), Buzzega and Novellani (2023), DellÁmico et al. (2023), and Boschetti and Novellani (2024).

In this study, we focus on the Vehicle Routing Problem with Lockers and Time Windows (VRPLTW) and the Vehicle Routing Problem with Private and Shared Delivery Locations (VRPPSDL). The VRPLTW, proposed by Buzzega and Novellani (2023), is characterized by several features: the demand of each customer can be met either directly at their home within specified time windows or stored in a nearby locker; both vehicles and lockers have capacity constraints; and each locker can be visited by at most one vehicle. The authors also considered a variant of the problem without time window constraints, referred to as the Vehicle Routing Problem with Lockers (VRPL). The VRPPSDL, proposed by Mancini and Gansterer (2021), is quite similar to the VRPLTW but has two key differences: there are no capacity constraints for the vehicles, and the assignment costs are constant.

To solve the VRPLTW and VRPL, Buzzega and Novellani (2023) proposed exact Mixed Integer Linear Programming (MILP) formulations. These authors also adapted their formulations to solve VRPPSDL instances and compared them with the exact formulation proposed by Mancini and Gansterer (2021). The results indicated that the formulations proposed by Buzzega and Novellani (2023) outperform those of Mancini and Gansterer (2021).

As highlighted by Costa et al. (2019), among the exact methods for solving VRPs exactly, BCP algorithms stand out. Unfortunately, implementing a BCP algorithm with state-of-the-art components for a specific VRP requires a significant amount of time, even for a skilled team. To mitigate this issue, Pessoa et al. (2020) proposed the VRPSolver framework, capable of generating a BCP algorithm to solve a specific VRP cariant based on the modeling provided by the user. A VRPSolver model contains a MILP formulation that includes variables representing feasible routes. The set of feasible routes is modeled by a set of paths with resource constraints on one or more directed graphs. The VRPSolver solves the model using a generic BCP algorithm that dynamically generates path variables by solving pricing subproblems. These are modeled in the framework as resource-constrained shortest-path problems (RCSP). In addition to the MILP formulation and directed graphs, the user can also define the so-called packing sets, which allow the user to activate some state-of-the-art components for BCP algorithms, such as ng-paths,

generation of rounded capacity cuts and rank-1 cuts with limited memory, route enumeration, among others. VRPSolver models are competitive or even considered state-of-the-art for several VRPs (Roboredo et al., 2023; Soares and Roboredo, 2023; Praxedes et al., 2024; Roboredo et al., 2024; Póvoa et al., 2025).

This work proposes two Branch-and-Cut-and-Price (BCP) algorithms implemented within the VRPSolver framework for the VRPLTW/VRPL. One of these algorithms integrates the routing and assignment decisions within the same pricing subproblem, while the other algorithm addresses these decisions in separate pricing subproblems. Building on the VRPSolver models proposed for VRPLTW/VRPL, we develop a specialized VRPSolver model for the Vehicle Routing Problem with Private and Shared Delivery Locations (VRPPSDL).

We present several computational experiments to demonstrate the robustness of the proposed approaches. For VRPL and VRPLTW, we apply the proposed methods to benchmark instances with up to 60 customers and 6 lockers, as used by Buzzega and Novellani (2023). For VRPPSDL, we evaluate our approach on 30 instances with up to 75 customers and 5 lockers, as proposed by Mancini and Gansterer (2021). Regarding VRPL/VRPLTW, the results indicate that the method which incorporates the assignment decision in the master problem outperforms the best exact formulations available in the literature, achieving lower computational times for almost all instances and solving several instances to optimality for the first time. For VRPPSDL, our method presented a lower computational time for each one of the 30 instances, where the hardest instances was optimalluy solved in under three minutes of execution. Moreover, for both VRPL/VRPLTW and VRPPSDL, we generated several additional instances with up to 60 customers and 6 lockers, and 100 customers and 10 lockers, respectively.

The remainder of this paper is organized as follows. Section 2 provides a brief literature review of the VRPLTW and VRPPSDL. Section 3 formally describes the VRPLTW and VRPPSDL. Section 4 presents the proposed VRPSolver models for both the VRPLTW and VRPPSDL. Section 5 discusses the computational results obtained from the proposed approaches on benchmark VRPLTW and VRPPSDL instances from the literature, as well as on additional instances introduced in this paper for both problems. Finally, Section 6 summarizes our conclusions.

2. Literature review

Jiang et al. (2020) investigated the traveling salesman problem with time windows (TSPTW) in the context of last-mile delivery for e-commerce. This problem extends the classic TSPTW by allowing customers to either receive their deliveries at home or collect them from lockers within a specified radius. Lockers are subject to both a maximum service radius and capacity constraints. The tour is designed in order to minimize the sum of three types of costs: the total travel cost, the total cost incurred by customers traveling to lockers, and the total cost associated with opening lockers. To solve the problem, the authors developed a MILP formulation alongside a variable neighborhood descent (VND) algorithm.

Mancini and Gansterer (2021) introduced the VRPPSDL, which is one of the problems addressed in this paper. The VRPPSDL considers three types of customers: those who must receive deliveries at home within a specified time window, those who must collect their deliveries from a locker within a certain radius, and those who have the flexibility to either receive deliveries at home or pick them up from a locker. Deliveries are handled by a fleet of homogeneous vehicles, and the objective involves three types of costs: a fixed cost for each vehicle used, travel costs, and a compensation cost applied whenever a customer is assigned to a locker. While vehicle capacity is not a constraint, the routes are restricted by a maximum duration, and each locker has a limited capacity and can be visited at most once across all routes. To address the problem, the authors proposed a MILP formulation, a matheuristic based on large neighborhood search (LNS), and an iterated local search algorithm.

Grabenschweiger et al. (2021) introduced the Vehicle Routing Problem with Heterogeneous Locker Boxes (VRPHLB), which incorporates the concept of a locker box station. At such stations, multiple slots of varying sizes are available. If a customer receives more than one parcel, the parcels can either be consolidated into a larger slot or distributed across several smaller slots. Therefore, in addition to routing and assignment decisions, the VRPHLB also involves determining how to pack the customers' parcels within the locker box station. To solve the variants pf the problem without and with package decision, the authors developed MILP formulations and a metaheuristic algorithm.

Buzzega and Novellani (2023) introduced the VRPLTW, which is one of the problems addressed in this paper. The VRPL can be viewed as a generalization of the VRPPSDL as all VRPPSDL characteristics are encompassed by VRPLTW as well the following characteristics is considered: the vehicles are capacitated and the assignment costs are not necessarily identical as it happens for VRPPSDL. To solve the problem, the authors proposed MILP formulations that were applied to VRPLTW as well to more three variants: a variant without time windows constraints, a variant with a single route, and a variant with a single route and without time windows constraints. They also adapted their formulations to solve VRPPSDL instances and instances from the problem proposed by Jiang et al. (2020).

DellÁmico et al. (2023) presented a variation of the pickup and delivery routing problem that incorporates time windows and the use of lockers. In this problem, customers can receive their deliveries either at home, using one of the available capacitated trucks, or through self-service lockers, which offer a more flexible option. Couriers are tasked with both delivering parcels and collecting returns during their routes. Returned packages may either be retrieved directly from customers' homes or from lockers. Customers have the choice to opt for home delivery, self-service at nearby lockers with a discount, or leave the decision to the logistics company. All services must be completed within specified time windows. To solve the problem, the authors presented three formulations, two branch-and-cut algorithms, and some valid inequalities.

Boschetti and Novellani (2024) introduced a VRP with lockers where, in addition to being

directly served or assigned to a locker, customers can also be served by a drone carried on the truck. The authors proposed four different formulations, and for one of them, they developed a branch-and-cut algorithm.

3. The problems

The Vehicle Routing Problem with Lockers and Time Windows (VRPLTW) and the Vehicle Routing Problem with Lockers (VRPL) are defined on a complete undirected graph $G = (V, E)$. The set of vertices V is partitioned into $V = \{0\} \cup V_C \cup V_L$, where 0 represents the depot, $V_C = \{1, 2, \ldots, n\}$ denotes a set of n customers, and $V_L = \{n+1, n+2, \ldots, n+m\}$ signifies a set of m available lockers. Each vertex $i \in V$ is associated with a time window $[a_i, b_i]$ and a service time st_i. Each customer $i \in V_C$ is linked to a positive demand d_i, and for each edge $e \in E$, there exists an associated cost and traversal time, denoted by c_e and t_e, respectively.

To serve the customers, there is an unlimited fleet of homogeneous vehicles, each with a positive capacity Q. Each customer $i \in V_C$ can either be visited directly or assigned to a locker $l \in V_L$, which generates an assignment cost f'_{il}. Each locker $l \in V_L$ has a service capacity ν_l and a coverage radius R_l, such that a customer $i \in V_C$ can only be assigned to locker l if $c_{il} \leq R_l$. In this context, for a given locker $l \in V_L$, we define γ_l as the set of customers that can be assigned to locker l.

The objective is to define routes that start and end at the depot while assigning customers to lockers, in accordance with the following constraints:

- Every customer $i \in V_C$ must either be directly visited by one of the routes or assigned to a locker $l \in V_L$ that is visited by one of the routes such that $i \in \gamma_l$.
- No locker can be visited more than once, even by different routes.
- The number of customers assigned to a locker $l \in V_L$ must not exceed its service capacity ν_l.
- For each vehicle, the sum of the demands of the customers it visits directly, combined with the demands of customers assigned to lockers it visits, must not exceed the vehicle's capacity Q.
- A vertex $i \in V$ can only be visited within its time window $[a_i, b_i]$, including the depot. This constraint is not applicable to the VRPL.

The routing and assignment decisions aim to minimize the total traversal cost in addition to the total assignment cost. The Vehicle Routing Problem with Private and Shared Delivery Locations (VRPPSDL) is very similar to the VRPLTW, with three key differences:

- There is no capacity constraint for the vehicles ($Q = +\infty$).

- The assignment costs are constant ($f'_{il} = f'_{jk}$, for $i, j \in V_C$ and $l, k \in V_L$).
- The total number of routes is also minimized in the objective function.

To illustrate the problems, we propose a toy instance, Example 1, with $|V_C| = 4$, $|V_L| = 2$, $Q = 3$ (with $Q = +\infty$ for VRPPSDL), and $\nu_5 = \nu_6 = 2$. Assume that the coverage radius of the lockers is defined such that $\gamma_5 = \{1, 2\}$ and $\gamma_6 = \{2, 3\}$. Additionally, consider that all demands are unitary and that there are no time window constraints. Figure 1 illustrates three examples of feasible solutions for Example 1, where the depot and customers are represented by circles, and lockers are represented by triangles. In the first solution, no assignments are made, and two routes are constructed, each visiting two customers. In the second solution, one of the routes visits locker 5, to which customers 1 and 2 are assigned. In the third solution, one route visits only customer 4 and returns to the depot, while the other route visits both lockers 5 and 6, with customer 1 assigned to locker 5 and customers 2 and 3 assigned to locker 6. We highlight that it is infeasible for the VRPL and VRPLTW to serve all customers with a single vehicle, as this would exceed the capacity constraint.

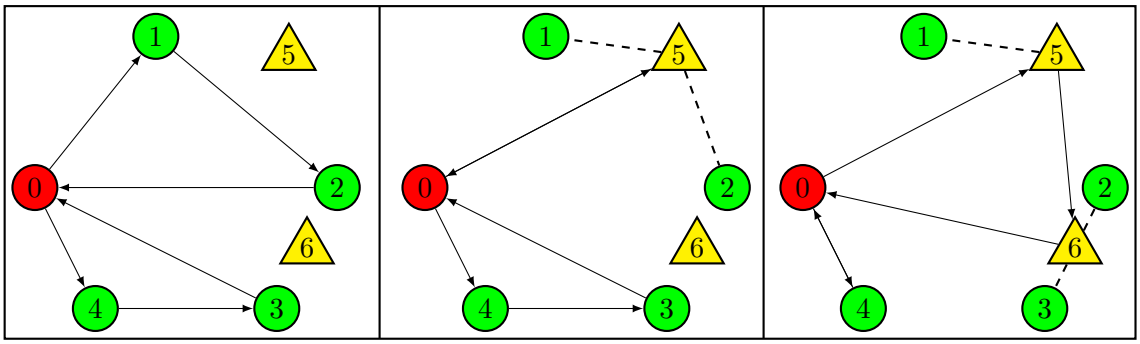

Figure 1: Example 1: illustration of feasible solutions.

4. VRPSolver models

A VRPSolver model consists of a Mixed Integer Linear Programming (MILP) formulation that incorporates path variables within one or more directed graphs defined by the user. The problem is solved using a generic BCP algorithm, where path variables are generated on demand by solving pricing subproblems, modeled as RCSP problem over the defined graphs.

In this study, we propose two VRPSolver models for the VRPLTW/VRPL. In the first model, each path variable encompasses both routing and assignment decisions. In the second model, each path variable represents either routing or assignment decisions, but not both.

VRPSolver Model 1 is intuitive in that all information related to a given route (such as visited customers/lockers and assignments made to visited lockers) is contained within a single path variable. However, this integration can significantly increase the complexity of solving the pricing

subproblems. To address this issue, VRPSolver Model 2 decomposes the visit and assignment decisions into separate paths, obtained through different pricing subproblems.

While VRPSolver Models 1 and 2 can also be easily adapted to solve VRPPSDL, we demonstrate how to derive a specialized VRPSolver model specifically for this problem based on these models.

4.1. VRPSolver Model 1 (single path-generator graph)

Sections 4.1.1 and 4.1.2 present the path-generator graph and the MILP formulation that constitute VRPSolver Model 1.

4.1.1. Path-generator graph for the VRPSolver Model 1

As mentioned, VRPSolver Model 1 includes path variables that encompass both routing and assignment decisions. To generate these paths, we define the graph $G' = (V', A)$, where $V' = \{v_{so}, v_{si}\} \cup \{v_i \mid i \in V \setminus \{0\}\} \cup \{\bar{v}_l \mid l \in V_L\} \cup \{\hat{v}_l^i \mid l \in V_L, i \in \gamma_l\}$. Before presenting the set of arcs A, we first define the constants ϕ_l^k for each $l \in V_L$ and $k = 1, \ldots, |\gamma_l|$, representing the k-th smallest customer index considering all customers in γ_l. The set of arcs A is defined as $A = A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_5 \cup A_6$, where the sets A_1, A_2, A_3, and A_4 are defined as follows:

- $A_1 = \{(v_{so}, v_i), (v_{so}, v_l), (v_i, v_{si}), (\bar{v}_l, v_{si}) : i \in V_C, l \in V_L\}$
- $A_2 = \{(v_i, v_j), (v_j, v_i) : i, j \in V_C, i \neq j\}$
- $A_3 = \{(v_i, v_l) : i \in V_C, l \in V_L\}$
- $A_4 = \{(\bar{v}_l, v_i) : l \in V_L, i \in V \setminus \{v_l\}\}$

For the set A_5, we define two parallel arcs a_{lk}^+ and a_{lk}^-, for each $l \in V_L$ and $k = 1, \ldots, |\gamma_l|$, that are defined as follows:

$$a_{lk}^+, a_{lk}^- = \begin{cases} (v_l, \hat{v}_l^{\phi_l^1}), & \text{if } k = 1 \\ (\hat{v}_l^{\phi_l^{k-1}}, \hat{v}_l^{\phi_l^k}), & \text{otherwise.} \end{cases}$$

Finally, the last set of arcs, A_6, is defined as $A_6 = \{(\hat{v}_l^{\phi_l^{|\gamma_l|}}, \bar{v}_l) : l \in V_L\}$.

The underlying idea of the graph is that each route, including assignments to visited lockers, corresponds to a path in G' that begins at v_{so} and ends at v_{si}. Thus, these vertices represent the depot at the start and end of the route, respectively. Each vertex v_i, with $i \in V_C$, represents customer i being visited at their location; v_l and $\bar{v}_l$, with $l \in V_L$, denote locker l at the vehicle's entry and exit points, respectively; and $\hat{v}_l^i$, for $l \in V_L$ and $i \in \gamma_l$, indicates the possibility of assigning customer i to locker l.

In this context, the set of arcs A_1 represents possible arrivals at and departures from the depot, while arcs A_2 represent connections between customers. Arcs A_3 and A_4 denote arrivals at and

departures from the lockers, respectively. The set of arcs A_5 represents parallel connections a^+_{lk} and a^-_{lk} arriving at vertex $\hat{v}^i_l$, for some $l \in V_L$ and customer $i \in \gamma_l$, to indicate whether customer i is assigned to locker l (using arc a^+_{lk}) or not (using arc a^-_{lk}).

Each vertex $i \in V'$ is associated with a vertex in V through the function $\tilde{v}$, defined as follows: $\tilde{v}(i) = 0$ if $i = v_{so}$ or $i = v_{si}$; $\tilde{v}(i) = j$ if $i = v_j$, $i = \bar{v}_j$, or $i = \hat{v}^j_l$ for some $l \in V_L$.

To ensure that the paths correspond to routes that satisfy the vehicle capacity and time window constraints, two resources, denoted as 1 and 2, are created. Each arc $a = (i, j) \in A$ is associated with the following consumptions q^1_a and q^2_a for resources 1 and 2, respectively, as follows:

$$q^1_{(i,j)} = \begin{cases} d_{\tilde{v}(j)}, & \text{if } \tilde{v}(j) \in V_C \\ 0, & \text{otherwise.} \end{cases}$$

$$q^2_{(i,j)} = \begin{cases} t_{\tilde{v}(i),\tilde{v}(j)} + st_{\tilde{v}(j)}, & \text{if } (i,j) \in A \setminus (A_5 \cup A_6) \\ 0, & \text{otherwise.} \end{cases}$$

Let p be a path in the graph G' that starts and ends at the vertices v_{so} and v_{si}, respectively. Let $j \in V'$ be any vertex visited along the path p. We denote the accumulated consumption of resources 1 and 2 on the path p when visiting vertex j as $S^1_{j,p}$ and $S^2_{j,p}$, respectively, and define their calculation as follows: $S^1_{j,p} = S^2_{j,p} = 0$ when $j = v_{so}$. For the case where $j \neq v_{so}$, let (i, j) be the incoming arc to j on the path p. For resource 1, we define $S^1_{j,p} = S^1_{i,p} + q^1_{(i,j)}$. For resource 2, we define $S^2_{j,p} = \max\{S^2_{i,p} + q^2_{(i,j)}, a_{\tilde{v}_j}\}$. We say that p in G' is resource constrained if $S^1_{j,p} \leq Q$ and $S^2_{j,p} \in [a_{\tilde{v}(j)}, b_{\tilde{v}(j)}]$ for every $j \in p$.

To illustrate G', consider a small instance, Example 2, with $|V_C| = 2$, $|V_L| = 1$, $Q = 2$, $\nu_3 = 2$, and $\gamma_3 = \{1, 2\}$. For each edge $e = \{i, j\} \in E$, the traversal times are given by $t_{\{0,1\}} = t_{\{0,2\}} = t_{\{0,3\}} = t_{\{1,3\}} = t_{\{2,3\}} = 1$ and $t_{\{1,2\}} = 2$. The vertices are associated with the following time windows: $[a_0, b_0] = [0, 3]$, $[a_1, b_1] = [2, 3]$, $[a_2, b_2] = [1, 3]$, and $[a_3, b_3] = [0, 3]$. We consider $st_i = 0$ for $i = 0, 1, 2, 3$. It is important to note that we are omitting routing and assignment costs. Finally, we define unitary demands for the customers: $d_1 = d_2 = 1$.

Figure 2 illustrates the resource consumption over the graph G' for Example 2. The values of q_1 and q_2 on each arc represent the resource consumption of resources 1 and 2, respectively, for the corresponding arc. The time windows of the vertices are indicated in brackets. Additionally, the arcs in red and blue illustrate two resource-constrained paths. These paths correspond to a feasible solution for the proposed Example 2, constructed from two routes: (i) visiting customer 1 at home and returning to the depot, and (ii) visiting Locker 3, to which customer 2 is assigned, and returning to the depot. It is noteworthy that assignment decisions were made in the blue path, which visited Locker 2. Customer 1 is not assigned as arc a^-_{31} is traversed, while Customer 2 is assigned to Locker 3, as arc a^+_{32} is traversed.

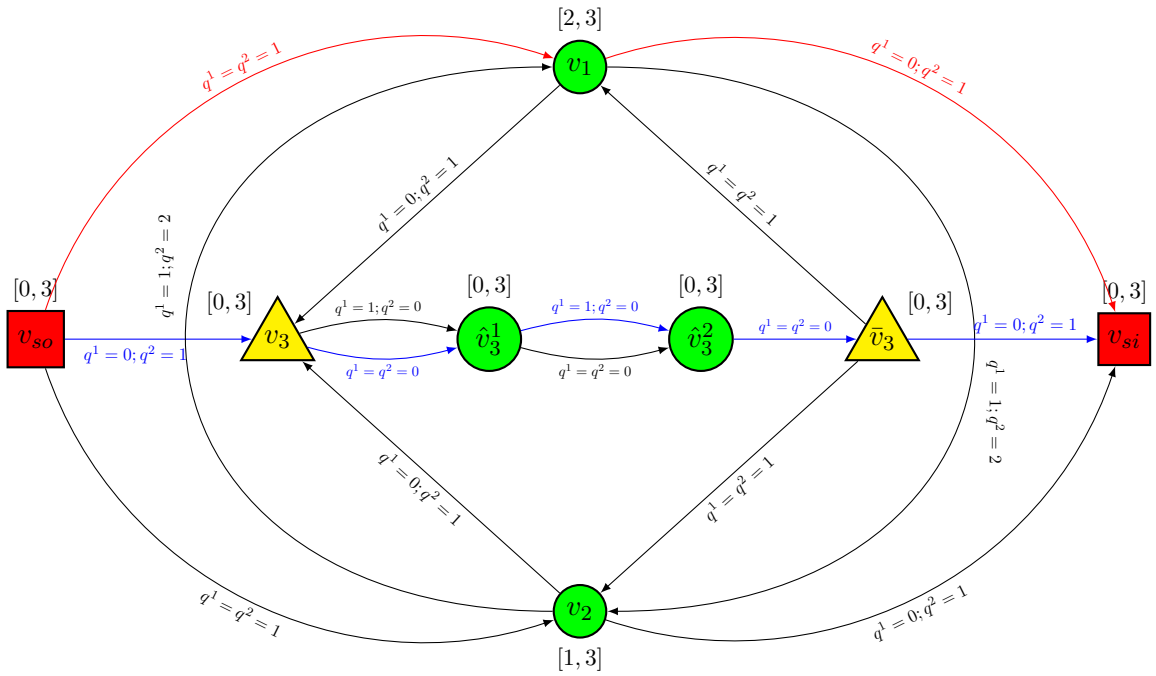

Figure 2: Example 2: illustration of resource consumption on G'.

Figure 3 illustrates the feasible solution associated with the two paths highlighted in Figure 2.

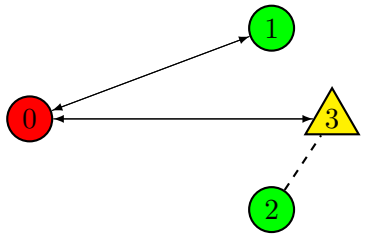

Figure 3: Example 2: a feasible solution.

4.1.2. MILP Formulation for the VRPSolver Model 1

Let P be the set of resource-constrained paths in G'. For each path $p \in P$ and each arc $a \in A$, let α_a^p denote the number of times the arc a appears in the path p. For each vertex $i \in V_C \cup V_L$, let $\delta(i)$ represent the set of edges in E adjacent to i. We now present the formulation used by the proposed VRPSolver Model 1. The formulation considers the following variables: for each edge $e \in E$, we define an integer variable x_e indicating the number of times edge e is traversed. For each locker $l \in L$ and for each customer $j \in \gamma_l$, we define a binary variable y_{jl} indicating whether customer j is assigned to locker l ($y_{jl} = 1$) or not ($y_{jl} = 0$). Additionally, for each locker $l \in V_L$, we define a binary variable t_l to indicate whether locker l is visited ($t_l = 1$) or not ($t_l = 0$). Finally, for each path $p \in P$, we define the integer variable λ_p to indicate how many times the

resource-constrained path p appears in the optimal solution. The formulation is as follows.

$$\text{Min} \sum_{e\in E} c_e x_e + \sum_{l\in V_L}\sum_{i\in\gamma_l} f_{il} y_{il} \tag{1a}$$

$$\text{S.t.} \sum_{e\in\delta(j)} x_e + \sum_{l\,:\,j\in\gamma_l} 2y_{jl} = 2 \qquad \forall j \in V_C; \tag{1b}$$

$$\sum_{e\in\delta(l)} x_e = 2t_l \qquad \forall l \in V_L; \tag{1c}$$

$$y_{il} \le t_l \qquad \forall l \in V_L\,,\, i \in \gamma_l; \tag{1d}$$

$$\sum_{j\in\gamma_l} y_{jl} \le \nu_l \qquad \forall l \in V_L; \tag{1e}$$

$$x_e = \sum_{p\in P}\left(\sum_{a\in M(x_e)} \alpha_a^p\right)\lambda_p \qquad \forall e \in E; \tag{1f}$$

$$t_l = \sum_{p\in P}\left(\sum_{a\in M(t_l)} \alpha_a^p\right)\lambda_p \qquad \forall l \in V_L; \tag{1g}$$

$$y_{il} = \sum_{p\in P}\left(\sum_{a\in M(y_{il})} \alpha_a^p\right)\lambda_p \qquad \forall l \in V_L, i \in \gamma_l; \tag{1h}$$

$$0 \le \sum_{p\in P} \lambda_p \le |V_C|; \tag{1i}$$

$$x_e \in \mathbb{Z}_+ \qquad \forall e \in E; \tag{1j}$$

$$y_{il} \in \{0,1\} \qquad \forall l \in V_L, i \in \gamma_l; \tag{1k}$$

$$t_l \in \{0,1\} \qquad \forall l \in V_L; \tag{1l}$$

$$\lambda_p \in \mathbb{Z}_+ \qquad \forall p \in P. \tag{1m}$$

The objective function (1a) aims to minimize the sum of the total traversal cost and the total assignment cost. Constraints (1b) ensure that each customer $j \in V_C$ is either served at home or through an assignment. Specifically, if $\sum_{e\in\delta(j)} x_e = 2$, then customer j is served at home, whereas if $\sum_{l\in\mathcal{L}_j} 2y_{jl} = 2$, customer j is served via an assignment.

Constraints (1c) ensure that the degree of a locker $l \in V_L$ equals 2 if the locker is visited ($t_l = 1$) or equals 0 if it is not visited ($t_l = 0$). Constraints (1d) ensure that a given customer can only be assigned to a locker if that locker is visited by some route. Constraints (1e) guarantee that the total demand assigned to any locker does not exceed its capacity.

Constraints (1f), (1g), and (1h) respectively relate the variables x, t, and y to the variables λ. This relationship is defined by a mapping function M, which is specified for each variable x, t, and y such that $M(x) \subseteq A$, $M(t) \subseteq A$, and $M(y) \subseteq A$.

Specifically, M must be defined to ensure that the value of each variable x_e, t_l, and y_{il} is determined by the number of times the arcs in $M(x_e)$, $M(t_l)$, and $M(y_{il})$ are utilized by the paths in the solution. Defining the correct mapping function is a crucial task for the modeler within the VRPSolver framework. For each variable $x_{(i,j)}$, we define $M(x_{(i,j)})$ as follows:

$$M(x_{(i,j)}) = \begin{cases} \{(v_{so}, v_j), (v_j, v_{si})\} & \text{if } i = 0, j \in V_C \\ \{(v_{so}, v_j), (\bar{v}_j, v_{si})\} & \text{if } i = 0, j \in V_L \\ \{(v_i, v_j), (v_j, v_i)\} & \text{if } i, j \in V_C \\ \{(v_i, v_j), (\bar{v}_j, v_i)\} & \text{if } i \in V_C, j \in V_L \\ \{(\bar{v}_i, v_j), (\bar{v}_j, v_i)\} & \text{if } i, j \in V_L \end{cases} \tag{2}$$

For the variables y and t, we define $M(y_{il}) = \{a^+_{lk}\}$, where $\phi^k_l = i$, and $M(t_l) = \{a^+_{l1}, a^-_{l1}\}$. Constraint (1i) establishes a lower and upper limit for the number of paths (routes) utilized in the solution. The remaining constraints ensure the domain of the variables.

Figure 4 illustrates the mapping function on G', where, on each arc of this graph, we indicate the variables that are part of the mapping set for that arc.

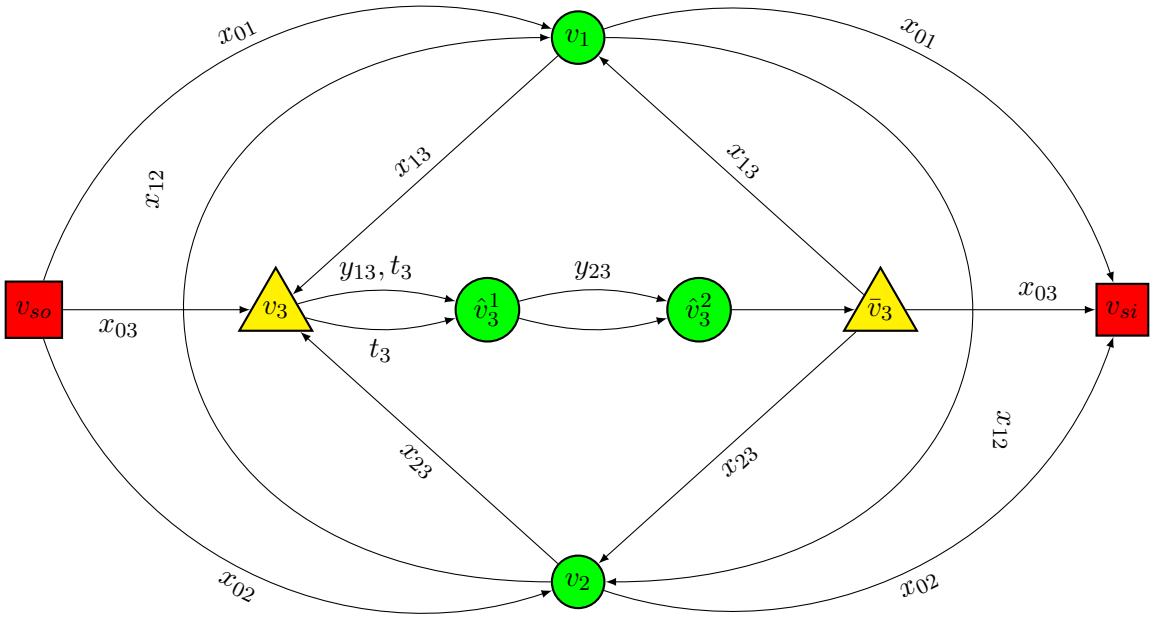

Figure 4: Example 2: illustration for mapping function on the graph G'.

4.1.3. The generic BCP algorithm

The VRPSolver employs a generic BCP algorithm based on the path-generator graph, the mapping functions, and the MILP formulation defined by the modeler. The framework solves a

formulation that exclusively utilizes path variables λ, where the variables x, t, and y are replaced according to equations (1f), (1g), and (1h), respectively. The path variables λ are generated on demand by solving a pricing subproblem, which constitutes a RCSP problem on G'. Branching on the variables x, t, and y is employed to close the primal-dual gap.

To utilize advanced BCP elements, specifically *ng*-path relaxation and limited-memory rank-1 cuts, the user must define what are known as packing sets. We briefly define packing sets here to ensure the paper is self-contained. For further details regarding packing sets, we refer the reader to Pessoa et al. (2020). A set $\mathcal{S} \subset A$ is considered a packing set if the arcs in $\mathcal{S}$ appear at most once in all paths that are part of some optimal solution. To activate the advanced BCP elements, the user must define $\mathcal{P} \subset 2^A$, a collection of mutually disjoint packing sets. The inclusion of more arcs from A in some packing set enhances the performance of the algorithm. For the proposed VRPSolver model, we define $\mathcal{P} = \mathcal{P}_C \cup \mathcal{P}_L$, where $\mathcal{P}_C = \bigcup_{c \in V_C} \{\{(i,j) : j = v_c\} \cup \{a^+_{lk} = (i,j) : l \in V_L, j = \hat{v}^c_l\}\}$ and $\mathcal{P}_L = \bigcup_{l \in V_L} \{\{(i,j) : j = v_l\}\}$. In other words, we define a packing set for each customer $c \in V_C$ composed of all arcs in A that represent the customer being directly served or assigned to some locker, as well as a packing set for each locker $l \in V_L$ composed of all arcs in A that represent arrivals at this locker.

4.2. VRPSolver Model 2 (multiple path-generator graphs)

The VRPSolver Model 2 separates routing and assignment decisions into distinct paths. Specifically, this model utilizes a path-generator graph to represent routing decisions and employs a different path-generator graph for each locker $l \in V_L$ to represent assignment decisions associated with that locker.

4.2.1. Path-generator graphs for the VRPSolver Model 2

For routing decisions, let $G^r = (V^r, A^r)$, where $V^r = V' \setminus \{\hat{v}^i_l : l \in V_L, i \in \gamma_l\}$ and $A^r = A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_7$. The sets A_1, A_2, A_3, and A_4 were defined previously, while A_7 contains a set of parallel arcs for each locker $l \in L$ that differ from each other by a third index k, which varies in D_l, where D_l represents the set of demands that can be assigned to locker l given its capacity ν_l. Formally, $A_7 = \{(v_l, \bar{v}_l, k) : l \in V_L, k \in D_l\}$. To ensure the vehicle capacity and time window constraints, we also define two resources R_1 and R_2 over G^r, as outlined in Section 4.1.1. For these resources, the consumption over arcs in A_1, A_2, A_3, and A_4 remains consistent with the definitions provided in Section 4.1.1. For each arc $a = (v_l, \bar{v}_l, k) \in A_7$, we define the consumption $q^1_a = k$ and the consumption $q^2_a = 0$. The definition of resource-constrained paths over G^r mirrors that presented for the graph G'. Figure 5 illustrates the resource consumption over G^r for Example 2.

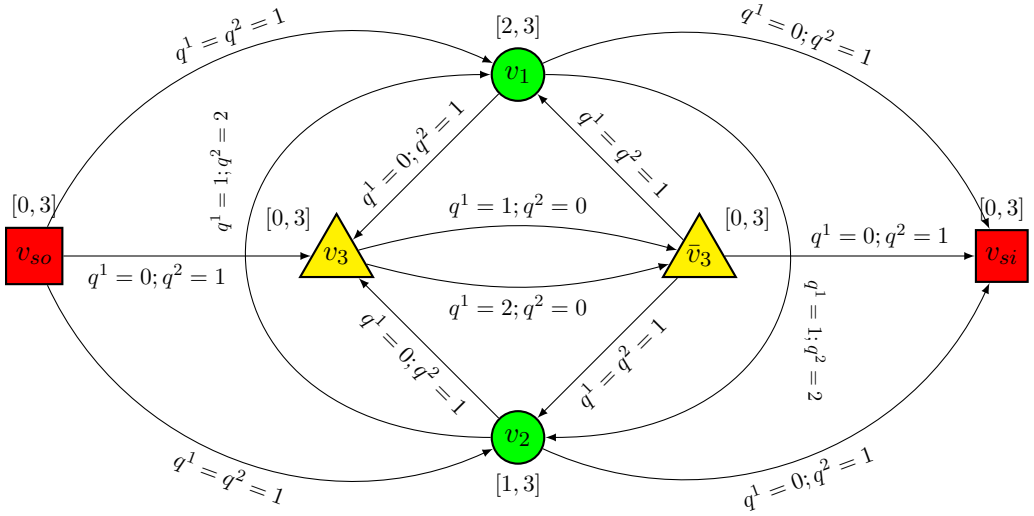

Figure 5: Example 2: illustration of resource consumption on the graph G^r.

Let $l \in V_L$ be a locker and p be a resource constrained path over G^r. Note that p indicates the total demand that is assigned to l through arcs in A_7, but p does not indicate which customers are assigned to l. To consider these decisions, we define a path-generator graph $G^l = (V^l, A^l)$, for each $l \in V_l$, in the following way. The set of vertices V^l is defined as $V^l = \{v_l, \bar{v}_l\} \cup \{\hat{v}_l^i : l \in V_L, i \in \gamma_l\}$ while the set of arcs A^l is defined as $A^l = A_5 \cup A_6$, where the sets A_5 and A_6 were defined in Section 4.1.1. Each graph G^l, for $l \in V_L$, is built in a way that there is a single node $\hat{v}_l^i$ for each customer $i \in \gamma_l$. Besides there are two parallel arcs a_{lk}^+ and a_{lk}^- arriving to $\hat{v}_l^i$, where $\phi_l^k = i$. The idea is that when a_{lk}^+ is traversed by a path then the customer i is being assigned to the locker l. On the other hand if a_{lk}^- is traversed then the customer i is not being assigned to the locker l. Figure 6 illustrates the graph G^3 for the proposed Example 2.

Figure 6: Example 2: illustration of the graph G^l.

Figure 7 illustrates the paths associated with the feasible solution presented in Figure 3 for the proposed Example 2. These paths are highlighted in red and blue.

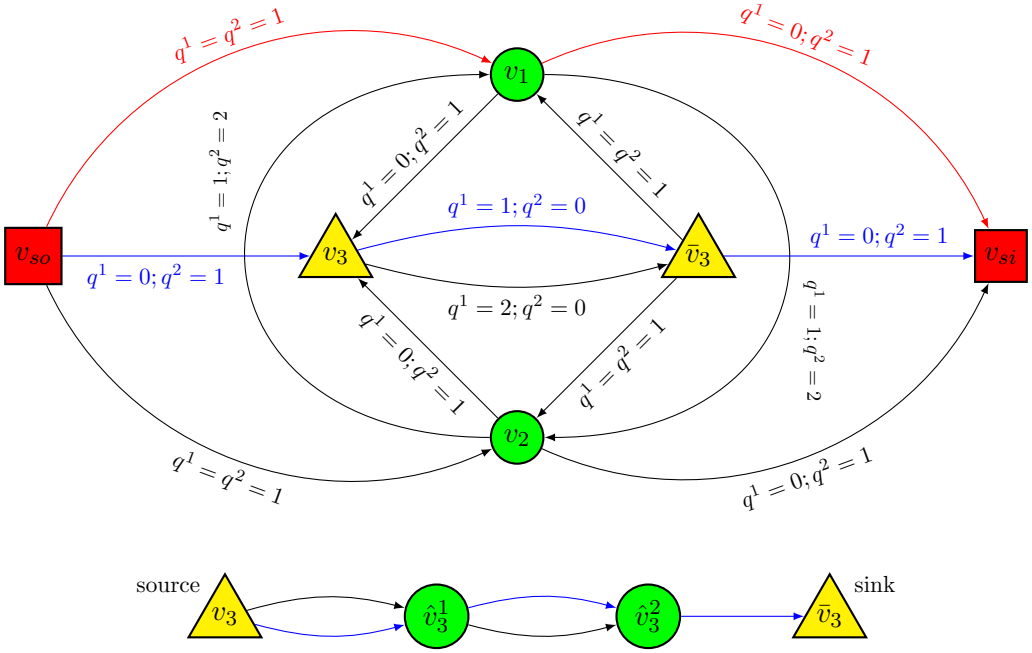

Figure 7: Example 2: paths associated with the feasible solution shown in Figure 3

4.2.2. MILP Formulation for the VRPSolver Model 2

Let P^r and P^l denote the sets of resource constrained paths over graphs G^r and G^l, respectively. The MILP formulation employs the variables x, y, and t defined in the VRPSolver model 1. Additionally, the formulation includes a binary variable z_l^k for each locker $l \in V_L$ and $k \in D_l$, which indicates whether the total demand assigned to locker l is equal to k ($z_l^k = 1$) or not ($z_l^k = 0$).

Regarding path variables, the formulation uses a variable λ_p^r for each resource constrained path $p \in P^r$, representing the number of times this path appears in the optimal solution. Similarly, the formulation uses a variable λ_p^l for each locker $l \in V_L$ and each resource-constrained path $p \in P^l$, representing the number of times this path appears in the optimal solution. The formulation is as follows.

$$\text{Min } (1a) \tag{3a}$$

$$\text{S.t.} (1b), (1c), (1d), (1e), (1j), (1k), (1l); \tag{3b}$$

$$\sum_{k \in D_l} k z_l^k = \sum_{i \in \gamma_l} d_i y_{il} \qquad \forall l \in V_L; \tag{3c}$$

$$\sum_{k \in D_l} z_l^k \leq 1 \qquad \forall l \in V_L; \tag{3d}$$

$$x_e = \sum_{p \in P^r} \left(\sum_{a \in M(x_e)} \alpha_a^p \right) \lambda_p^r; \tag{3e}$$

$$t_l = \sum_{p \in P^r} \left(\sum_{a \in M(t_l)} \alpha_a^p \right) \lambda_p^r \qquad \forall l \in V_L; \tag{3f}$$

$$y_{il} = \sum_{p \in P^l} \left(\sum_{a \in M(y_{il})} \alpha_a^p \right) \lambda_p^l \qquad \forall l \in V_L, i \in \gamma_l; \tag{3g}$$

$$z_l^k = \sum_{p \in P^r} \left(\sum_{a \in M(z_l^k)} \alpha_a^p \right) \lambda_p^r \qquad \forall l \in V_L, k \in D_l; \tag{3h}$$

$$0 \leq \sum_{p \in P^r} \lambda_p^r \leq |V_C|; \tag{3i}$$

$$\sum_{p \in P^l} \lambda_p^l = 1 \qquad l \in V_L; \tag{3j}$$

$$z_l^k \in \{0, 1\} \qquad \forall l \in V_L, k \in \mathcal{D}_l; \tag{3k}$$

$$\lambda_p^r \in \mathbb{Z}_+ \qquad \forall p \in P^r; \tag{3l}$$

$$\lambda_p^l \in \mathbb{Z}_+ \qquad \forall l \in V_L, p \in P^l; \tag{3m}$$

The formulation (3) is similar to the formulation (1). The main difference between these formulations is that in formulation (3), we require the constraints (3c) to ensure the correct relationship between the variables z and y. The constraints (3d) prevent a locker from being assigned two different demand values. Moreover, in formulation (3), for each variable x, y, t, and z, we must define the mapping function M according to the arcs from the graphs G^r and G^l, where $l \in V_L$. For each variable x_e, $e \in E$, we define $M(x_e)$ with the same definition as in equation (2), but considering the set of arcs A^r. Similarly, for the variables y and t, we also consider $M(y_{il}) = \{a_{lk}^+\}$ and $M(t_l) = \{a_{l1}^+, a_{l1}^-\}$, as was done for VRPSolver model 1. The difference is that $a_{lk}^+, a_{lk}^- \in A^l$. Finally, for the variables z, we define $M(z_l^k) = \{(v_l, \bar{v}_l, k)\}$. Figures 8 illustrate the mapping over the graphs G^r and G^l, respectively, regarding Example 2.

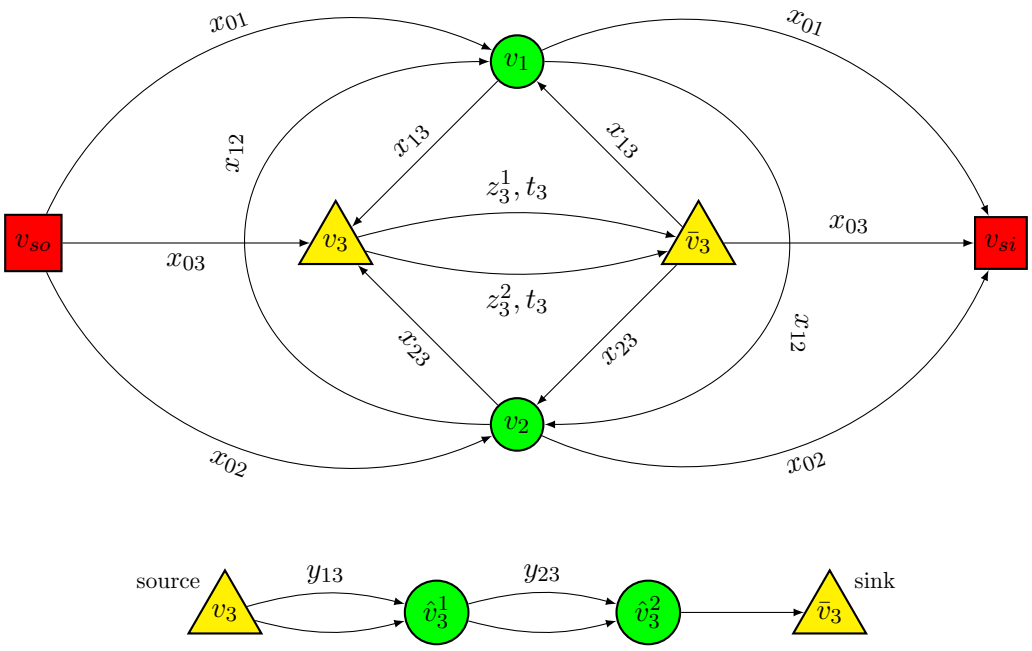

Figure 8: Example 2: illustration for the mapping on the graphs G^r and G^l

For the VRPSolver model 2, the packing sets are defined in the same way as for VRPSolver model 1.

4.3. A specialized VRPSolver model for the VRPPSDL

For the VRPPSDL, both proposed VRPSolver models 1 and 2 could also be easily adapted to solve it. However, we propose a specialized VRPSolver model based on VRPSolver Model 2 that explores two specific characteristics of the VRPPSDL: there is no capacity constraint for the vehicles, and there are fixed costs for assignments. The proposed specialized VRPSolver model for the VRPPSDL uses a single path-generator graph $G^{r'}$, which is similar to G^r. The differences between these graphs are as follows: we do not consider resource 1, and we do not construct parallel arcs in $G^{r'}$, as these are used solely to guarantee the vehicle capacity constraint. Instead of using parallel arcs between the nodes v_l and $\bar{v}_l$ for each $l \in V_L$, the graph $G^{r'}$ considers a single arc that is used solely to identify whether the locker l is visited or not.

In the MILP formulation, we do not consider the variables z, as they are used only to guarantee the vehicle capacity constraint. Additionally, we can relax the variables y due to the following property: if the variables x and t have integer values, then the remaining assignment problem to calculate the values of the variables y is equivalent to a transportation problem where the set of sources consists of all the lockers visited by some route according to the values of the variables t, while the set of customers includes all customers that are not directly visited at home according to the values of the variables x. This property is particularly important for avoiding symmetries that would arise due to the fixed assignment costs in the VRPPSDL. The mapping function used

for the variables x and t is the same as that used in VRPSolver model 2, but it is applied over the arcs from $G^{r'}$.

Figure 9 illustrates the consumption of resource 2 and the mapping on $G^{r'}$ for the proposed Example 2.

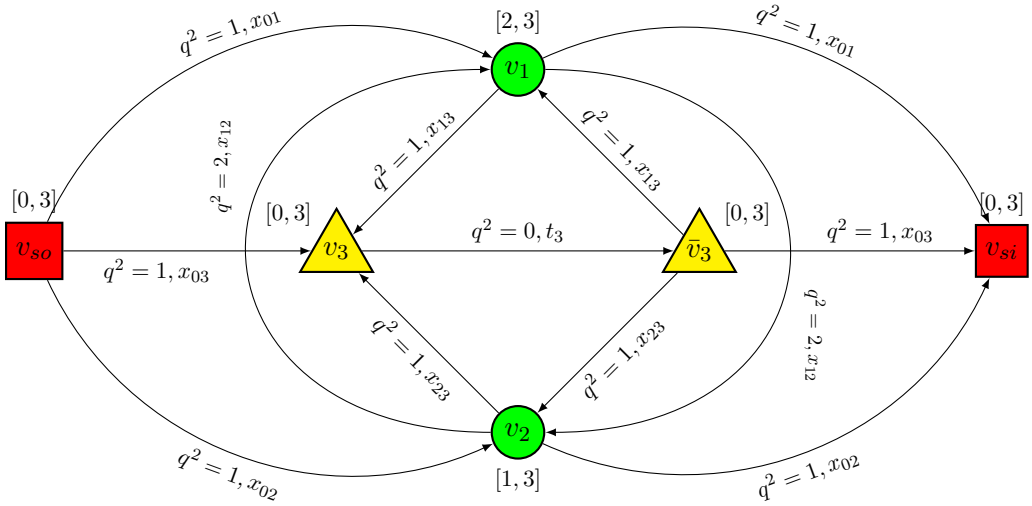

Figure 9: Example 2: illustration for the Resource Consumption and Mapping on graph $G^{r'}$.

Let $P^{r'}$ denote the sets of resource constrained paths over graphs $G^{r'}$. The MILP formulation for the VRPSolver model specialized for the VRPPSDL uses an integer variable λ_p indicating how many times p appears at optimal solution. Besides the formulation uses the variables x, t and y defined before. The formulation follows.

$$\text{Min} \sum_{i \in V_C \cup V_L} x_{0i} + \sum_{e \in E} c_e x_e + \sum_{l \in V_L} \sum_{i \in \gamma_l} f_{il} y_{il} \tag{4a}$$

$$\text{S.t.} (1b), (1c), (1d), (1e), (1j), (1l); \tag{4b}$$

$$x_e = \sum_{p \in P^{r'}} \left(\sum_{a \in M(x_e)} \alpha_a^p \right) \lambda_p; \tag{4c}$$

$$t_l = \sum_{p \in P^{r'}} \left(\sum_{a \in M(t_l)} \alpha_a^p \right) \lambda_p \qquad \forall l \in V_L; \tag{4d}$$

$$0 \leq \sum_{p \in P^{r'}} \lambda_p \leq |V_C|; \tag{4e}$$

$$0 \leq y_{il} \leq 1 \qquad \forall l \in V_L, i \in \gamma_l; \tag{4f}$$

$$\lambda_p \in \mathbb{Z}_+ \qquad \forall p \in P^{r'}. \tag{4g}$$

For the formulation (4), the objective function (4a) is very similar to the function (1a), but it additionally accounts for minimizing the number of vehicles used in the solution. The constraints in this formulation are also closely aligned with those proposed for the VRPSolver models 1 and 2.

For the specialized VRPPSDL VRPSolver model, the packing sets are defined as $\mathcal{P} = \mathcal{P}_C \cup \mathcal{P}_L$, where $\mathcal{P}_C = \bigcup_{c \in V_C} \{\{(i,j) : j = v_c\}\}$ and $\mathcal{P}_L = \bigcup_{l \in V_L} \{\{(i,j) : j = v_l\}\}$. In other words, we define a packing set for each customer $c \in V_C$ and for each locker $l \in V_L$, consisting of all arcs in A that represent a direct visit to this customer or locker.

5. Computational Experiments

In this section, we present a series of computational experiments to illustrate the robustness of the proposed models. All experiments were conducted on a computer equipped with an Intel Core i7-10700 processor running at 2.90 GHz and 16 GB of RAM. The operating system utilized was Ubuntu, and the BCP algorithms were implemented within the VRPSolver v0.4.1a framework (`https://vrpsolver.math.u-bordeaux.fr/`). All instances and VRPSolver models used in this paper are available online at `https://github.com/mcroboredo/VRPLTW-VRPPSDL`.

5.1. Results on VRPLTW/VRPL instances.

For the VRPLTW and VRPL, we follow the approach of Buzzega and Novellani (2023) and utilize the instances proposed by Jiang et al. (2020), which were generated by incorporating locker vertices into the benchmark instances of the Traveling Salesman Problem with Time Windows (TSPTW). Specifically, the original instances are divided into two sets: the first set features time windows with a width of 20, initially proposed by Dumas et al. (1995), while the second set has time windows with a width of 100, introduced by Gendreau et al. (1998) based on the instances from Dumas et al. (1995). Each set contains 10 instances. Following Buzzega and Novellani (2023), we generated distinct VRPLTW/VRPL instances as follows. The locations of the depot and the customers remain identical to those in the original TSPTW instances. For the lockers, we set $|V_L| = \lceil \frac{|V_C|}{10} \rceil$, and they are added to the first $|V_L|$ vertices using the following set of coordinates: (25,0), (12,5), (37,5), (12,5), (37,5), (12,5), (25,0), (37,5), (25,0), (25,0), (12,5), and (0,0). Furthermore, the demand of each customer is unitary ($d_i = 1, \forall i \in V_C$). The cost c_e and traversal time t_e for each edge $e \in E$ of the graph are determined by the Euclidean distance between the points, rounded to two decimal places. The cost f_{jl} of assigning a customer $j \in V_C$ to a locker $l \in V_L$ is given by $f_{jl} = 0.5 \times c_{(j,l)}$. These instances do not account for service times. For each instance, and for each locker $l \in V_L$, we have $R_l = 20$ and $\nu_l = 5$. The time windows for the depot and customers remain as in the original TSPTW instance, whereas for each locker $l \in V_L$, the time windows are set as $[a_l, b_l] = [a_0, b_0]$. The vehicle capacity is defined as $Q = \lceil \frac{n}{2} \rceil$.

5.1.1. Comparison with Buzzega and Novellani (2023)

The exact methods proposed by Buzzega and Novellani (2023) were tested on the VRPLTW/VRPL instances described earlier, with a time limit of 7200 seconds. To ensure a fair comparison, we evaluated the single-thread performance of the processors used in this study against that of the processor utilized in the referenced work via the website `https://www.cpubenchmark.net/`. Based on this comparison, we divide each literature time by 1.4 and set a time limit of 5142 seconds ($\approx 7200/1.4$) for our experiments in this subsection. Each instance tested in this subsection was executed in two sequential rounds. In the first round, the proposed model was tested without providing any valid upper bound. In the second round, the best cost obtained from the first round was used as an initial upper bound.

Table 1 presents a comparison with the literature for VRPLTW/VRPL instances with $|V_C| = 20, 40, 60$. The instances are grouped by variant (VRPLTW or VRPL) and by the number of customers. The following headers are used for the columns. The columns *Variant* and $|V_C|$ indicate, respectively, the variant and the number of customers associated with each group of instances. The column *#Inst* indicates the number of instances in the group. The column *Average Time (s)* represents the average solution time in seconds, considering all instances in the group. For unsolved instances, the time limit is used in the calculation of the average. The column *#Opt* represents the number of proven optimal solutions obtained within the time limit. The column *Literature* reports the results obtained by the best exact method proposed by Buzzega and Novellani (2023), while the columns *This work Without UB* and *This work With UB* present statistics obtained by the proposed model without and with the use of an initial upper bound.

Upon reviewing Table 1, we observe that only the Proposed VRPSolver Model 2 outperforms the best literature model in both computational time and the number of optimal solutions obtained. Additionally, VRPSolver Model 2 exhibited strong computational performance even without using an initial upper bound.

5.1.2. Statistics of VRPSolver model 2 for additional VRPLTW/VRPL instances.

We generated additional VRPLTW/VRPL instances by varying the number of lockers $|V_L|$ in the set $\{2, 4, 6, 8, 10\}$. Table 2 presents the statistics for the proposed VRPSolver Model 2 with upper bounds for VRPLTW/VRPL instances with $|V_C| = 40, 60$. The instances were categorized based on the combinations of $(|V_C|, |V_L|)$, while detailed statistics for each instance are provided in Appendix Appendix A. A time limit of 18,000 seconds was imposed for these experiments. For each category of instances, we report the following metrics: the number of instances in the corresponding group (Column *#Inst.*), the number of optimal solutions identified within the time limit by our method across all instances in the group (Column *#opt.*), the average percentage gap between the best upper bound value and the lower bound at the root node for all instances in the group (Column *Avg.* $Gap_0(\%)$), and the average total processing time in seconds for all instances in the group (Column *Avg. Time(s)*). In calculating the average time, a value of 18,000

					This paper							
					VRPSolver 1				VRPSolver 2			
			Literature		Without ub		With ub		Without ub		With ub	
Variant	n	#Inst.	Avg. Time(s)	#Opt	Avg. Time(s)	#Opt	Avg. Time(s)	#Opt	Avg. Time(s)	#Opt	Avg. Time(s)	#Opt
VRPLTW	20	10	0.9	10	1.6	10	1.3	10	1.2	10	1.0	10
	40	10	2029.1	9	754.6	9	130.2	10	80.1	10	17.3	10
	60	10	4508.7	4	5142.0	0	5142.0	0	2529.2	8	1046.3	9
VRPL	20	10	1.4	10	1.5	10	1.3	10	2.7	10	1.0	10
	40	10	4045.1	6	4115.5	2	2623.8	5	144	10	26.1	10
	60	10	4997.6	2	5142.0	0	5142.0	0	3557.7	7	868.6	9
Global		60	2597.1	41	2526.2	31	2173.4	35	1052.5	55	326.7	58

Table 1: Comparison with the literature for VRPLTW/VRPL instances.

		VRPLTW			VRPL		
$(\|V_C\|, \|V_L\|)$	#Inst.	*#opt.*	Avg. $Gap_0(\%)$	Avg. Time(s)	*#opt.*	Avg. $Gap_0(\%)$	Avg. Time(s)
(40,2)	10	10	0.7	8.3	10	1.4	15.1
(40,4)	10	10	1.5	8.1	10	0.6	30.4
(40,6)	10	10	2.9	551.8	10	2.5	64.3
(40,8)	10	9	3.7	1878.1	10	4.0	209.9
(40,10)	10	10	4.2	615.5	10	4.4	922.0
(60,2)	10	10	1.1	154.7	10	1.2	402.6
(60,4)	10	10	1.3	224.6	10	1.8	1143.4
(60,6)	10	10	2.3	2380.7	10	2.7	1704.3
(60,8)	10	9	2.6	6079.1	9	3.5	5172.0
(60,10)	10	7	3.6	6523.1	9	4.4	4369.6
Global	100	95	2.4	1567.1	98	2.6	1403.4

Table 2: Results of our approach over VRPLTW/VRPL instances with 40 and 60 customers.

seconds is used for instances that were not solved to optimality.

By analyzing Table 2, we observe that the proposed method solved 193 out of the 200 tested instances to optimality. Furthermore, we note that the instances become more challenging as the number of customers increases or the number of lockers increases.

5.2. Results on literature VRPPSDL Instances.

The benchmark instances from the VRPPSDL literature were generated by Mancini and Gansterer (2021) in the following manner.The instances are grouped into three sets based on the number of customers (25, 50, or 75), with each set containing 10 instances. Each of the 30 instances includes five lockers. Customers are randomly placed within a predefined area represented as a 10×10 km square. The depot is located in the southern part of this area, while the lockers are strategically positioned in the southeast, southwest, northeast, northwest, and center. A time horizon of 720 minutes is considered for all instances, divided into 12 time slots, each lasting 60 minutes.

For each edge $e = (i, j)$, the travel cost c_e and travel time t_e are defined as $c_e = t_e = 3 \times dist_{ij}$, where $dist_{ij}$ represents the Euclidean distance between points i and j. Each customer has a service time of 5 minutes, and each locker has a fixed service time of 10 minutes. The assignment cost for each locker l and customer i is fixed at $f_{il} = 5$. For each customer $i \in V_C$, the demand is unitary ($d_i = 1, \forall i \in V_C$). For each locker $l \in V_L$, the value of R_l is equal to 15. Regarding the capacities of the lockers, four have the same capacity, while one locker has a larger capacity, enabling routes that exclusively visit lockers to be feasible.

Table 3 presents a comparison between the VRPSolver model proposed in this paper and the best formulations available in the literature for the 30 benchmark VRPPSDL instances. The table includes the following columns: The column *Inst.* represents the identifier of the corresponding

	$\|V_C\| = 25$			$\|V_C\| = 50$			$\|V_C\| = 75$		
		Time(s)			Time(s)			Time(s)	
			This			This			This
Inst.	ub^*	Lit.	paper	ub^*	Lit.	paper		Lit.	paper
1	161.37	0.3	0.1	266.80	197.3	18.1	356.34	691.4	150.3
2	166.63	16.8	0.9	267.97	19.7	7.2	371.00	328.8	25.7
3	146.56	0.7	0.2	273.05	208.0	2.1	373.91	327.8	16.8
4	161.04	11.1	0.4	268.89	6.6	0.5	386.87	2571.4	19.1
5	157.94	1.7	0.2	271.32	151.7	2.0	378.09	515.2	23.7
6	160.83	3.5	0.4	268.32	16.8	0.8	376.93	2571.4	13.5
7	152.69	1.0	0.2	253.18	10.8	1.2	373.77	301.0	18.6
8	165.16	4.4	0.3	266.68	34.2	21.3	364.18	347.5	162.7
9	151.54	1.0	0.2	267.56	327.3	1.6	372.78	1313.0	22.4
10	151.98	2.2	0.1	273.60	93.7	0.8	360.79	200.1	2.5
Avg		4.3	0.3	267.7	106.6	5.6	371.5	916.8	45.5

Table 3: Comparison with the literature for VRPPSDL instances.

instance, which varies from 1 to 10. The column ub^* displays the optimal cost value for each instance. The columns *Time (s) Lit.* and *Time (s) This paper* report the total CPU time in seconds taken by the best literature formulation (considering all formulations proposed by Mancini and Gansterer (2021) and Buzzega and Novellani (2023)) and by our method, respectively. Each literature time is adjusted by a factor of 1.4 for consistency with the experiments detailed in Section 5.1. It is important to note that the literature employs a time limit of 3600 seconds. Therefore, when the literature time is reported as 2571.4 ($\approx 3600/1.4$), it indicates that the instance was not optimally solved by any of the literature formulations.

Upon examining Table 3, we conclude that our approach outperforms the best formulations available in the literature across all 30 tested instances. The maximum time taken by our method for any single instance was 162.7 seconds.

5.2.1. Results on additional VRPPSDL instances

To generate more challenging VRPPSDL instances, we implement the following modifications to the VRPLTW instances: we set the service time to 5 for each customer and 10 for each locker; we establish a fixed assignment cost of 5 between each customer $i \in V_C$ and each locker $l \in V_L$ such that $(f_{il} = 5, i \in V_C, l \in V_L)$; and we do not impose a capacity constraint on the vehicles. All other parameters remain identical to those of the corresponding VRPLTW instances. We generate 10 instances for each combination of $(|V_C|, |V_L|)$, where $|V_C| \in \{80, 90, 100\}$ and $|V_L| \in \{8, 9, 10\}$. The instances with 90 customers are generated by selecting the first 90 customers from the corresponding VRPLTW instance with 100 customers.

Table 4 presents the results of our VRPSolver model on the proposed additional VRPPSDL instances. The instances are grouped into sets of 10 based on the number of customers and

the number of lockers; however, the Appendix B provides detailed statistics for each individual instance. The column labels used are consistent with those in Table 1.

	$\|V_L\| = 8$			$\|V_L\| = 9$			$\|V_L\| = 10$		
$\|V_C\|$	#opt.	Avg. Gap_0	Avg. Time(s)	#opt.	Avg. Gap_0	Avg. Time(s)	#opt.	Avg. Gap_0	Avg. Time(s)
80	10	1.9	176.5	10	2.4	425.7	10	2.5	319.2
90	10	2.3	2488.9	10	2.2	2323.7	10	2.2	1899.2
100	8	2.1	5997.3	9	2.1	3941.1	8	2.2	4789.7
Global	28	2.1	2887.6	29	2.2	2230.2	28	2.3	2336.1

Table 4: Average statistics of our methdo for VRPPSDL instances with 80, 90, and 100 customers.

Upon analyzing Table 4, we observe that our approach optimally solved 85 out of the 90 tested instances. The five unsolved instances involve 100 customers. Furthermore, we note that the instances become more challenging as the number of customers increases; however, this trend does not necessarily hold true when the number of lockers increases.

6. Conclusions

In this paper, we study the VRPLTW and VRPPSDL, two VRP variants that incorporate lockers. In both problems, each customer can either be visited directly by a route or assigned to a locker. The key differences between these problems lie in factors such as time windows, vehicle capacity, and fixed assignment costs.

For the VRPLTW, we propose two Branch-Cut-and-Price (BCP) algorithms implemented within the VRPSolver framework. These algorithms generate path variables on demand by solving RCSP problems over one or more directed graphs. One model integrates routing and assignment decisions within a single path variable, while the other model separates these decisions into distinct path variables.

For the VRPPSDL, we developed a specialized VRPSolver model based on the second model proposed for the VRPLTW. This specialized model leverages specific characteristics of the VRPPSDL: the vehicle capacity constraint is not enforced through a resource, and the assignment variables can be relaxed. The latter is particularly important for avoiding symmetries that would arise due to the fixed assignment costs in the VRPPSDL.

To demonstrate the robustness of the proposed models for the VRPLTW and VRPPSDL, we conducted several computational experiments using benchmark instances from the literature for both problems. The results show that the proposed methods outperform the best exact methods from the literature for both VRPLTW and VRPPSDL, solving many instances to optimality for the first time. We then generated additional instances, with up to 60 customers and 10 lockers for VRPLTW, and 100 customers and 10 lockers for VRPPSDL. The experiments on

these additional instances demonstrated that our method was able to find optimal solutions for most of them; however, some instances were not solved to optimality. These additional instances and the computational results obtained in this work may serve as benchmarks for future heuristic and exact methods for the problems.

As future work, we intend to study the polyhedron of these problems to identify cutting planes capable of reducing the root node gap. This could potentially lower the computational time required by the algorithms, enabling the solution of even larger instances.

References

Boschetti, M. A. and Novellani, S. (2024). Last-mile delivery with drone and lockers. *Networks*, 83(2):213–235.

Buzzega, G. and Novellani, S. (2023). Last mile deliveries with lockers: Formulations and algorithms. *Soft Computing*, 27(18):12843–12861.

Costa, L., Contardo, C., and Desaulniers, G. (2019). Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science*, 53(4):946–985.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management science*, 6(1):80–91.

DellÁmico, M., Montemanni, R., and Novellani, S. (2023). Pickup and delivery with lockers. *Transportation Research Part C: Emerging Technologies*, 148:104022.

Dumas, Y., Desrosiers, J., Gelinas, E., and Solomon, M. M. (1995). An optimal algorithm for the traveling salesman problem with time windows. *Operations research*, 43(2):367–371.

Gendreau, M., Hertz, A., Laporte, G., and Stan, M. (1998). A generalized insertion heuristic for the traveling salesman problem with time windows. *Operations Research*, 46(3):330–335.

Grabenschweiger, J., Doerner, K. F., Hartl, R. F., and Savelsbergh, M. W. (2021). The vehicle routing problem with heterogeneous locker boxes. *Central European Journal of Operations Research*, 29:113–142.

Jiang, L., Dhiaf, M., Dong, J., Liang, C., and Zhao, S. (2020). A traveling salesman problem with time windows for the last mile delivery in online shopping. *International Journal of Production Research*, 58(16):5077–5088.

Mancini, S. and Gansterer, M. (2021). Vehicle routing with private and shared delivery locations. *Computers & Operations Research*, 133:105361.

Pessoa, A., Sadykov, R., Uchoa, E., and Vanderbeck, F. (2020). A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, 183:483–523.

Póvoa, C. L. R., Roboredo, M. C., Velasco, A. S., Pessoa, A. A., and Paes, F. G. (2025). A hybrid grasp and tabu-search heuristic and an exact method for a variant of the multi-compartment vehicle routing problem. *Expert Systems with Applications*, 259:125319.

Praxedes, R., Bulhões, T., Subramanian, A., and Uchoa, E. (2024). A unified exact approach for a broad class of vehicle routing problems with simultaneous pickup and delivery. *Computers & Operations Research*, 162:106467.

Roboredo, M., Lima, D., Silva, J. M. P., and Uchoa, E. (2024). Branch-cut-and-price algorithms for two routing problems with hotel selection. *Computers & Industrial Engineering*, page 110467.

Roboredo, M., Sadykov, R., and Uchoa, E. (2023). Solving vehicle routing problems with intermediate stops using vrpsolver models. *Networks*, 81(3):399–416.

Soares, V. C. and Roboredo, M. (2023). On the exact solution of the multi-depot open vehicle routing problem. *Optimization Letters*, pages 1–17.

Vincent, F. Y., Susanto, H., Jodiawan, P., Ho, T.-W., Lin, S.-W., and Huang, Y.-T. (2022). A simulated annealing algorithm for the vehicle routing problem with parcel lockers. *IEEE Access*, 10:20764–20782.

Appendix A. Results for each VRPLTW/VRPL instance with $|V_C| \in \{40, 60\}$.

Tables A.5, A.6, A.7, and A.8 present statistics of the proposed method for VRPLTW instances with 40 customers, VRPLTW instances with 60 customers, VRPL instances with 40 customers, and VRPL instances with 60 customers, respectively. The following labels are used for the columns. Column ub^* reports the best cost found within 18,000 seconds of execution. Column $Gap_0(\%)$ indicates the percentage gap at the root node. Column *Total Time(s)* provides the total CPU time in seconds consumed by the proposed algorithm. A time limit of 18,000 seconds was imposed for each experiment.

Appendix B. Results for each VRPPSDL instance with $|V_C| \in \{80, 90, 100\}$.

Tables B.9, B.10, B.11, present statistics of the proposed method for the aditional VRPPSL instances with 80, 90 and 100 customers, respectively. The column labels used are consistent with those in Tables A.5, A.6, A.7, and A.8.

	$\|V_L\| = 2$			$\|V_L\| = 4$			$\|V_L\| = 6$			$\|V_L\| = 8$			$\|V_L\| = 10$		
Instance	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0(%)	Total Time(s)	ub^*	Gap_0	Total Time(s)
n40w100_1	320.79	0.0	1.7	320.79	2.8	10.1	308.91	3.1	53.0	286.63	2.4	39.3	286.63	4.3	255.5
n40w100_2	326.95	3.2	56.8	305.70	2.9	39.2	293.59	4.8	5051.9	282.16	3.3	37.7	276.06	3.3	69.8
n40w100_3	329.26	1.9	10.2	280.52	1.8	8.5	280.52	2.8	114.5	278.20	2.9	113.5	274.12	3.4	279.6
n40w100_4	299.77	0.3	2.0	282.21	0.3	0.2	272.56	1.3	4.6	264.54	2.1	10.2	263.31	3.9	78.7
n40w100_5	320.22	1.4	6.3	288.09	1.7	6.3	268.63	3.5	110.5	259.92	5.8	18000.0	253.10	5.5	4184.1
n40w20_1	421.53	0.0	1.0	385.93	0.2	0.1	340.73	4.1	93.9	330.22	5.2	323.2	323.12	4.9	376.2
n40w20_2	391.99	0.0	1.8	334.09	1.0	1.2	305.83	2.1	9.1	294.74	4.2	130.0	291.84	3.6	120.0
n40w20_3	349.09	0.0	1.0	315.03	2.4	11.2	301.27	3.6	63.7	279.86	5.2	65.0	276.81	4.8	438.1
n40w20_4	301.56	0.1	1.1	281.99	0.6	1.5	276.13	2.3	7.7	271.58	3.1	32.2	271.58	4.3	237.3
n40w20_5	370.43	0.0	1.1	355.77	1.2	2.9	321.55	1.6	9.5	321.55	2.5	30.3	318.85	4.2	115.4
Avg.	343.16	0.7	8.3	315.01	1.5	8.1	296.97	2.9	551.8	286.94	3.7	1878.1	283.54	4.2	615.5

Table A.5: Statistics for each VRPLTW instance with 40 customers.

	$\|V_L\| = 2$			$\|V_L\| = 4$			$\|V_L\| = 6$			$\|V_L\| = 8$			$\|V_L\| = 10$		
Instance	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	Gap_0	Total Time(s)
n60w100_1	389.18	1.2	18.8	360.59	1.8	521.6	351.00	3.3	3103.0	337.11	1.7	38.0	333.30	3.3	275.9
n60w100_2	414.28	0.7	21.2	394.39	1.6	94.4	379.80	4.9	16835.2	367.52	4.3	9573.5	364.24	8.6	18000.0
n60w100_3	377.25	0.5	27.3	358.34	0.9	43.3	350.71	2.8	2939.7	341.40	3.4	17051.2	338.91	3.8	18000.0
n60w100_4	445.48	0.9	39.4	418.49	0.8	51.0	381.25	2.9	480.2	370.91	4.1	18000.0	367.58	4.3	18000.0
n60w100_5	428.35	1.6	121.7	397.95	3.2	804.8	356.96	2.4	119.1	348.21	3.5	6183.5	345.45	4.3	3846.8
n60w20_1	425.85	1.3	115.2	372.84	0.1	3.7	366.48	1.3	11.2	354.90	3.0	9327.0	347.97	3.1	3786.0
n60w20_2	484.97	1.0	146.4	464.89	2.7	641.8	429.20	1.4	90.6	402.22	1.7	195.6	398.74	2.1	1936.5
n60w20_3	468.75	0.0	3.5	415.73	0.5	12.8	400.77	1.7	133.9	394.51	1.6	227.4	385.94	1.6	83.8
n60w20_4	532.17	2.3	1021.6	441.23	1.2	48.4	405.90	0.1	10.0	381.17	0.9	108.5	381.17	0.9	98.7
n60w20_5	523.76	2.0	31.6	428.13	0.7	24.6	405.87	1.8	84.2	388.93	1.6	86.2	380.21	4.3	1203.7
Avg.	449.00	1.1	154.7	405.26	1.3	224.6	382.79	2.3	2380.7	368.68	2.6	6079.1	364.35	3.6	6523.1

Table A.6: Statistics for each VRPLTW instance with 60 customers.

	$\|V_L\| = 2$			$\|V_L\| = 4$			$\|V_L\| = 6$			$\|V_L\| = 8$			$\|V_L\| = 10$		
Instance	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	Gap_0	Total Time(s)
n40w100_1	283.59	1.7	24.9	283.16	0.1	48.3	279.45	3.5	65.5	270.54	4.7	1068.0	270.41	5.3	4758.9
n40w100_2	280.99	0.0	1.1	271.93	0.2	0.3	265.17	0.5	2.6	264.64	1.9	5.0	261.87	2.4	37.5
n40w100_3	274.29	1.4	13.5	261.37	0.0	114.1	261.37	3.1	268.8	261.37	4.5	691.6	261.37	4.6	3057.7
n40w100_4	275.97	2.1	58.9	262.59	0.0	14.0	261.79	1.9	36.9	257.39	2.3	14.2	255.75	3.4	51.0
n40w100_5	256.84	1.2	7.3	244.00	1.4	0.8	240.47	0.1	1.6	240.47	4.4	26.6	240.47	5.1	155.5
n40w20_1	309.04	0.9	4.2	308.00	2.0	11.2	296.28	3.8	41.2	294.18	5.5	108.4	289.74	4.6	183.1
n40w20_2	282.00	0.6	1.9	280.13	0.2	6.1	276.42	2.6	12.7	269.86	3.8	8.5	269.86	3.8	32.1
n40w20_3	285.58	1.3	17.6	282.35	2.5	95.1	273.33	5.0	199.1	268.09	6.8	142.0	263.06	5.6	780.4
n40w20_4	257.62	2.0	13.8	250.77	0.0	4.7	248.05	1.9	9.4	248.05	4.1	28.6	248.05	5.9	133.8
n40w20_5	288.40	2.6	7.7	276.11	0.0	9.5	276.11	2.1	5.6	276.11	2.2	6.4	276.11	3.3	29.9
Avg.	279.43	1.4	15.1	272.04	0.6	30.4	267.84	2.5	64.3	265.07	4.0	209.9	263.67	4.4	922.0

Table A.7: Statistics for each VRPL instance with 40 customers.

	$\|V_L\| = 2$			$\|V_L\| = 4$			$\|V_L\| = 6$			$\|V_L\| = 8$			$\|V_L\| = 10$		
Instance	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	Gap_0	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	Gap_0	Total Time(s)
n60w100_1	299.94	0.7	94.9	299.94	0.8	126.4	299.94	1.0	115.2	299.94	1.9	575.6	298.25	3.6	1350.0
n60w100_2	329.37	2.3	1065.5	328.86	4.1	5538.3	328.86	4.8	11360.2	322.67	5.8	18000.0	322.05	9.4	18000.0
n60w100_3	303.28	0.0	6.7	301.16	1.3	32.1	301.16	3.0	53.5	301.16	5.1	1836.6	299.69	5.2	1976.1
n60w100_4	330.44	1.5	150.0	329.80	1.4	184.1	327.92	2.6	759.8	324.45	2.6	2204.7	321.18	2.6	1310.3
n60w100_5	315.75	1.5	200.6	310.51	1.9	52.1	307.53	2.6	296.7	307.53	3.6	2059.3	307.53	4.7	2645.2
n60w20_1	331.45	1.3	217.4	317.66	1.7	173.8	317.66	3.0	280.9	316.98	3.4	3671.7	316.98	4.2	5437.1
n60w20_2	350.30	2.4	2036.9	350.30	2.6	4210.6	345.23	2.7	2077.1	344.21	3.3	15819.3	341.88	2.9	3883.6
n60w20_3	352.83	1.2	192.8	351.63	2.4	894.1	351.06	3.0	926.2	349.71	4.0	4034.9	347.77	4.3	3485.7
n60w20_4	342.67	0.3	14.6	336.14	0.6	33.9	336.14	1.2	180.3	333.54	2.2	403.3	333.55	2.1	337.2
n60w20_5	341.22	0.5	47.0	335.71	1.3	188.3	335.71	2.7	993.4	332.73	2.9	3115.2	331.97	5.2	5270.9
Avg.	329.72	1.2	402.6	326.17	1.8	1143.4	325.12	2.7	1704.3	323.29	3.5	5172.0	322.08	4.4	4369.6

Table A.8: Statistics for each VRPL instance with 60 customers.

	$\|V_L\| = 8$			$\|V_L\| = 9$			$\|V_L\| = 10$		
Instance	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)
n80w100_1	424.42	1.3	89.1	424.42	1.3	93.6	424.42	1.3	85.6
n80w100_2	415.55	2.6	155.7	413.04	2.0	86.6	413.04	2.0	88.5
n80w100_3	435.97	1.0	61.6	435.97	1.1	34.4	435.97	1.1	28.7
n80w100_4	440.68	1.2	6.5	438.66	1.3	9.2	438.66	1.3	9.7
n80w100_5	394.27	1.2	24.0	394.27	1.2	32.9	394.27	1.2	35.3
n80w20_1	479.61	3.3	1038.6	477.32	4.4	2977.4	477.32	4.5	1997.7
n80w20_2	480.42	1.0	33.8	479.27	2.7	88.3	479.27	3.2	77.9
n80w20_3	481.29	2.1	103.6	474.49	3.2	72.8	474.49	3.4	70.1
n80w20_4	444.23	4.1	204.2	443.23	4.3	757.9	443.23	4.3	683.9
n80w20_5	500.13	1.4	48.0	497.00	2.4	104.4	497.00	2.7	115.0
Avg.		1.9	176.5		2.4	425.7		2.5	319.2

Table B.9: Statistics for each VRPPSDL instances with 80 customers.

	$\|V_L\| = 8$			$\|V_L\| = 9$			$\|V_L\| = 10$		
Instance	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)
n90w100_1	470.63	2.9	9800.5	464.60	2.3	2059.7	464.60	2.3	2370.6
n90w100_2	475.00	1.8	291.3	469.87	1.5	302.3	469.87	1.5	319.0
n90w100_3	449.42	1.6	582.6	449.42	1.7	11853.2	449.42	1.7	10868.3
n90w100_4	442.45	1.0	68.8	442.45	1.0	113.6	442.45	1.0	102.6
n90w100_5	441.04	3.0	2705.9	435.42	2.0	357.1	435.42	2.0	321.0
n90w20_1	551.72	1.6	135.4	540.93	1.7	98.8	540.93	1.7	64.8
n90w20_2	515.61	4.2	7793.9	510.59	4.2	6153.5	510.59	4.2	2412.1
n90w20_3	521.71	1.6	48.4	517.39	1.8	32.5	517.39	1.9	37.8
n90w20_4	535.94	0.8	23.8	531.35	1.8	50.6	531.35	1.8	42.6
n90w20_5	544.70	4.1	3438.4	537.04	4.1	2215.3	537.04	4.1	2453.3
Avg.		2.3	2488.9		2.2	2323.7		2.2	1899.2

Table B.10: Statistics for each VRPPSDL instances with 90 customers.

	$\|V_L\| = 8$			$\|V_L\| = 9$			$\|V_L\| = 10$		
Instance	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)	ub^*	$Gap_0(\%)$	Total Time(s)
n100w100_1	506.42	3.7	18000.0	500.08	2.8	18000.0	498.77	2.5	18000.0
n100w100_2	516.42	1.8	650.6	513.33	1.7	427.0	512.95	1.6	500.5
n100w100_3	492.49	1.5	13122.7	492.49	1.6	11295.5	492.97	1.6	18000.0
n100w100_4	483.17	1.5	1129.7	483.17	1.6	1161.0	482.18	1.4	990.5
n100w100_5	464.11	2.2	601.2	459.97	1.5	380.1	460.48	1.6	439.1
n100w20_1	589.94	0.5	27.0	581.59	0.9	51.0	582.29	1.0	53.5
n100w20_2	556.23	3.8	18000.0	552.70	4.2	5827.3	552.62	4.2	6861.1
n100w20_3	580.37	1.9	416.9	574.92	2.0	171.8	574.88	2.5	226.0
n100w20_4	579.45	1.2	134.6	573.45	1.9	200.4	573.27	1.9	174.9
n100w20_5	595.57	3.4	7889.8	585.33	3.2	1896.6	584.97	3.2	2651.7
Avg.		2.1	5997.3		2.1	3941.1		2.2	4789.7

Table B.11: Statistics for each VRPPSDL instances with 100 customers.