

Branch-Price-and-Cut for the Vehicle Routing Problem With Simultaneous Delivery and Pickup, Time Windows, and Load-Dependent Cost

Carolin Hasse^{a,*}, Stefan Irnich^a

^a*Chair of Logistics Management, Department of Business & Economics, Johannes Gutenberg University Mainz,
Jakob-Welder-Weg 9, 55128 Mainz, Germany.*

Abstract

The vehicle routing problem with load-dependent cost is an extension of the classical capacitated vehicle routing problem in which the cost of traveling along an arc is dependent on the load carried by the vehicle. For the benefit of generalization, this work considers the vehicle routing problem with simultaneous delivery and pickup, time windows, and load-dependent cost (VRPSDPTW-LDC). We utilize both continuous and discontinuous monotonically non-decreasing load-dependent cost functions. These cost structures are justified by real-life applications: First and foremost, transportation cost rises in load due to increasing fuel cost. In addition, cost functions may also show discontinuities due to toll-by-weight schemes, weight restricted passage, and lift axles that may be raised when the vehicle is empty or lightly loaded, therefore decreasing tire wear. We employ a fully equipped branch-price-and-cut algorithm to solve the VRPSDPTW-LDC. A major complication in its development is the consistent handling of the load-dependent cost in the column-generation subproblem when solved by bidirectional labeling algorithms. Indeed, in the VRPSDPTW-LDC, the precise load on board is not known when a partial path is constructed. We provide a unifying description of the associated resource extension function for forward and backward labeling. In several computational experiments, we analyze algorithmic components of the branch-price-and-cut algorithm, and give managerial insights on the impact of the cost structure on key metrics such as total cost, the number of routes, and the average load carried in an optimal solution.

Keywords: routing, load-dependent cost, simultaneous delivery and pickup, branch-price-and-cut, dynamic-programming labeling algorithm

1. Introduction

In many variants of the (*capacitated*) *vehicle routing problem* ((C)VRP, [Toth and Vigo, 2014](#); [Irnich et al., 2014](#)) the cost of traversing an arc (i, j) is constant and known at the time of planning. This includes important variants such as the *VRP with time windows* (VRPTW, [Solomon, 1987](#)), the *capacitated arc routing problem* (CARP, [Golden and Wong, 1981](#)), the *pickup-and-delivery problem (with time windows)* (PDP(TW), [Dumas et al., 1991](#)), the *split-delivery VRP* (SDVRP, [Dror and Trudeau, 1989](#)), and many more. Even many variants of stochastic VRPs are based on the assumption that routing costs are deterministic and constant ([Bertsimas, 1992](#)). In this paper, however, we consider VRPs in which the routing cost depends on the load onboard the vehicle and the resulting total weight of the vehicle. This weight typically differs before and after visiting a customer and is influenced by the sequence in which customers are visited. More precisely, we assume that for each possible direct connection between two points, i.e., for every arc (i, j) , a non-decreasing cost function is given. This function $c_{ij}(q_{ij})$ describes the routing cost depending on the quantity q_{ij} which is the vehicle's load when traversing the arc (i, j) .

*Corresponding author.

Email address: chasse@uni-mainz.de (Carolin Hasse)

A load-dependent cost structure can be justified in several real-life applications. First and foremost, transportation cost rises in weight and load due to increasing fuel consumption (e.g., [Xiao et al., 2012](#); [Bousonville et al., 2022](#)). Additionally, toll-by-weight schemes may induce piecewise cost functions when no cost or a smaller cost is charged below a given weight threshold, and a higher cost, which may be also load-dependent, is charged above the threshold (e.g., [Shen et al., 2009](#)). Cost structures may also include discontinuities. One example for this is the case of weight-restricted passage, which authorities impose to ensure structural integrity of bridges, roads, etc. Then, vehicles must follow different itineraries when traveling from point i to point j , depending on which road segments are admissible given their actual weight ([Drexel, 2012](#), Sect. 2.2.1). Additionally, some trucks are equipped with lift axles, which can be raised when traveling with a light load¹, thereby rapidly decreasing tire wear and rolling resistance ([Surcel and Bonsi, 2015](#)).

The handling of load-dependent costs on arcs has important algorithmic implications. Many heuristics and exact algorithms for VRPs construct routes by extending a partial path, one arc at a time. A partial path begins at the depot and must be extended further to reach the depot again to form a feasible route. In the case of load-dependent VRPs, cost computation for partial paths must consider the quantities q_{ij} . However, for problems involving collections and deliveries at customers, the quantities q_{ij} are not precisely known for the segments of a partial path. Even for VRPs with only deliveries (from depot to customers), the partial paths in the backward direction have unknown q_{ij} -values on their arcs. Symmetrically, for VRPs with only collections, the unknown quantity q_{ij} occurs in forward direction. To generalize, we will consider the *VRP with simultaneous delivery and pickup, time windows, and load-dependent cost* (VRPSDPTW-LDC).

Consideration of load-dependent routing cost is not a new concept in the literature. However, we found that many different terms have been used to describe VRPs that include this aspect. Section 2 will provide a comprehensive overview of VRP variants with load-dependent cost. In addition, there is a commonly accepted nomenclature describing the weight of a vehicle and its onboard load that we will summarize in Section 3.1. Most importantly, a unified and simple description of load-dependent costs that can be used in different algorithmic frameworks is missing. This is the main contribution of this paper.

Branch-price-and-cut (BPC) algorithms are the leading methodology for solving many VRPs exactly ([Costa et al., 2019](#); [Desrosiers et al., 2025](#)). Here, we describe the implementation of a complete BPC algorithm for solving the VRPSDPTW-LDC. A major complication in its development is a consistent handling of the load-dependent cost, which complicates the solving of the pricing subproblem, which is a variant of the *shortest path problem with resource constraints* (SPPRC, [Irnich and Desaulniers, 2005](#)). SPPRCs are usually solved using dynamic-programming labeling algorithms. Thus, the quantities q_{ij} are variables in the pricing subproblem that must be considered, since reduced costs depend on them. In particular, dominance between labels depends on these quantities. We discuss different classes of functions $c_{ij}(q_{ij})$ and their impact on the type of SPPRC. We present a generic labeling algorithm that is applicable to any class of functions that is closed under addition and shift. For example, we show that for (piecewise) linear cost functions on arcs, the resulting label in the pricing subproblem also has a (piecewise) linear cost function.

Our contributions can be summarized as follows:

- We formally define the VRPSDPTW-LDC as the general variant of a VRP with load-dependent cost.
- We formally describe the load-dependent cost functions of forward and backward partial paths together with resource extension functions to be used to propagate the (reduced) cost information when a partial path is extended. As a result, the VRPSDPTW-LDC pricing can be solved with state-of-the-art dynamic-programming labeling algorithms, including techniques such as the consideration of unreachable customers, the *ng*-path relaxation, bidirectional labeling, the integration of non-robust cuts, etc.
- In computational experiments conducted with a BPC algorithm tailored to the VRPSDPTW-LDC, we demonstrate the effect of the load-dependent extension on the algorithm’s computational efficiency.
- On the basis of further experiments, we provide managerial insights into the impact of load-dependent cost on optimal routing strategies.

¹We would like to thank Nils Boysen from the University of Jena, Germany, for pointing us to this aspect.

The remainder of this work is structured as follows: We provide an overview of the related scientific literature in Section 2. In Section 3, we formally define the VRPSDPTW-LDC. Section 4 summarizes the BPC algorithm with a focus on the solution of the pricing problem. Section 5 presents the computational study, and Section 6 discusses managerial insights. Final conclusions are drawn in Section 7.

2. Literature Review

Dumas et al. (1991) were among the first to consider a type of load-dependent objective function in the context of the PDPTW. The objective only differentiates between an empty vehicle and one with an arbitrary positive load. They solve the PDPTW using a BPC algorithm (the term BPC was not yet introduced), implementing a forward labeling algorithm to solve the subproblem. Note that effective bidirectional labeling for PDP was later developed (Righini and Salani, 2006; Ropke and Cordeau, 2009; Gschwind et al., 2018). From the perspective of specifying the onboard load for a partial path, PDP variants are trivial, since the quantities q_{ij} are known precisely at any point in a forward and a backward partial path.

To the best of our knowledge, Kara et al. (2007) were the first to define an objective function that rises linearly in load for the VRP. They introduce the *energy minimizing VRP* (EMVRP, also referred to as the *cumulative VRP* or the *fuel consumption rate considered CVRP*) as an extension of the CVRP, considering both the case of delivery demands and that of pickup demands, albeit separately. In the EMVRP, the energy consumed by a vehicle traveling along any distance is dependent on the load it is carrying. A linear cost function is defined as a multiple of the length of the arc and the load carried by the vehicle on this arc. Kara et al. formulate *mixed integer programs* (MIPs) for both the collection EMVRP and the delivery EMVRP, and show how to extend them to include distance constraints. Comparing distance-minimization and energy-minimization, they highlight differences in the solutions regarding routes, energy consumed, and total distance traveled. Since its introduction in 2007, the EMVRP has been solved both using heuristics (e.g., Xiao et al., 2012; Frías et al., 2023) and exact approaches (e.g., Fukasawa et al., 2016; Mulati et al., 2022). Some authors, such as Xiao et al. (2012), elaborate on the objective function, explicitly including the curb weight of each vehicle. They employ a *simulated annealing* (SA) approach to solve the EMVRP, while Frías et al. (2023) combine *machine learning* (ML) with *ant colony optimization* (ACO). An exact approach to the EMVRP has been presented by Fukasawa et al. (2016) with a BPC algorithm, formulating the subproblem as a shortest q -route problem over a multigraph and solving it through dynamic programming. Mulati et al. (2022) extend that research by proposing several formulations for the EMVRP and using the BPC algorithm of Fukasawa et al. (2016) to solve and compare the new formulations.

The linear objective function of the EMVRP has also been applied to several extensions which are all solved by heuristic approaches. Zhang et al. (2011) extend the *multi-depot VRP* (MDVRP) to consider weight-related cost, where the objective function incorporates both a fixed cost for each vehicle dispatched and a load-dependent and distant-dependent function defined by a cost factor for each arc traversed. They use a *scatter search* (SS) to solve the problem heuristically. Liu et al. (2015) introduce the *load-dependent VRPTW* and solve it using a combination of *tabu search* (TS) and *adaptive large neighborhood search* (ALNS). The *VRP with simultaneous delivery and pickup* (VRPSDP) was first extended by Zachariadis et al. (2015) to consider load-dependent cost and solved using a *local search* (LS) heuristic.

Exact approaches are limited to smaller instances. In the same work, Zachariadis et al. (2015) present a *branch-and-cut* (B&C) approach, solving VRPSDP instances with up to 50 customers and load-dependent cost. Luo et al. (2017) introduce the *SDVRP with time windows* (SDVRPTW) in a version with linear weight-related cost and solve it using a BPC algorithm. Eydi and Alavi (2018) study a variant of the SDVRP which uses the objective of the EMVRP to minimize fuel consumption. They propose a MIP model and validate it by solving some small instances using a MIP solver. For larger instances, they employ an SA approach.

Motivated by toll structures on expressways in China, Shen et al. (2009) consider the *CVRP with toll-by-weight rule*. Here, the cost function can incorporate both vehicle costs (such as driver or maintenance costs) and the toll-by-weight rules that exist on some highways in China, in which the tolls are calculated dependent on load carried and distance traveled. Depending on the province and highway, tolls may be calculated with linear function in load and distance, or piecewise linear and quadratic functions. Shen et al.

(2009) develop an SA algorithm to solve the problem. Building on that, Zhang et al. (2010) consider a single-vehicle VRP, i.e., a *traveling salesman problem* (TSP) with customer demand, and a load-dependent cost function modeling a toll-by-weight scheme. They are the first to optimally solve this problem for any toll structure that rises monotonically in load using a *branch-and-bound* (B&B) approach. Refinements of their work are published in Zhang et al. (2012). Luo et al. (2016) extend the *VRP with stochastic demands* (VRP-SD) to consider weight related cost. They solve it using an ALNS heuristic. Most recently, Trang et al. (2025) introduce the *VRP with load-dependent distance and route time limit* (VRPLDRT). Their cost function is defined by the multiple of the cumulative customer demand plus a base load carried over the entire tour and the distance traveled with said load. They also consider the impact of the load on the service time. To solve the problem, they employ both a MIP model, solved via MIP solver, and a two-step heuristic. For the latter, they construct a tour using a modified (load-dependent) savings heuristic and improve it using outlier removal and *neighborhood search* (NS) with inter-cluster insertion.

The *pollution-routing problem* (PRP, Bektaş and Laporte, 2011) is another VRP extension with multiple variants. All PRP variants consider factors that contribute to greenhouse gases being emitted by vehicles, one of them being the load carried. Other factors are the speed of vehicle and (expected) congestion on the road. *VRPs with electric fleet* (EVRP) may also use load-dependent cost, as battery energy consumption increases with load. EVRPs are also often solved using heuristics, e.g., *large neighborhood search* (LNS) is used by Rastani and Çatay (2023). However, VRPs that consider battery energy are often constrained by additional restrictions such as the need to visit recharging stations. They can have multiple objectives such as making decisions on fleet composition. As the focus of our paper is not on environmental impact, we refer to (Garside et al., 2024) for an overview of *green VRPs* such as the PRP and EVRP.

Table 1 summarizes the literature presented in this section. For each reference, it specifies the load-dependent VRP variant considered, the type of solution approach (i.e., exact, heuristic, or both), the solution method(s) used, and the form of the load-cost relationship in the objective function.

Reference	Problem	Type	Method	Load-Cost Relation
Dumas et al. (1991)	PDPTW	exact	BPC	empty/full
Kara et al. (2007)	EMVRP	exact	MIP	linear
Xiao et al. (2012)	EMVRP	heuristic	SA	linear
Fukasawa et al. (2016)	EMVRP	exact	BPC	linear
Mulati et al. (2022)	EMVRP	exact	MIP/BPC	linear
Frías et al. (2023)	EMVRP	heuristic	ML+ACO	linear
Zhang et al. (2011)	MDVRP	heuristic	SS	linear + fixed cost per vehicle
Liu et al. (2015)	VRPTW	heuristic	TS+ALNS	linear
Zachariadis et al. (2015)	VRPSDP	both	LS/B&C	linear
Luo et al. (2017)	SDVRPTW	exact	BPC	linear
Liu and Jiang (2018)	VRPTW	heuristic	LS/TS	linear
Eydi and Alavi (2018)	SDVRP	both	MIP/SA	linear
Trang et al. (2025)	VRPLDRT	both	MIP/NS	linear + penalty
Shen et al. (2009)	CVRP	heuristic	SA	piecewise (non-)linear
Zhang et al. (2012)	TSP	exact	B&B	piecewise (non-)linear
Luo et al. (2016)	VRP-SD	heuristic	ALNS	piecewise (non-)linear, stochastic
Bektaş and Laporte (2011)	PRP	exact	MIP	broader objective
Rastani and Çatay (2023)	EVRP	both	MIP/LNS	battery consumption

Table 1: Overview of load-dependent VRP variants.

3. The Vehicle Routing Problem With Simultaneous Delivery and Pickup, Time Windows, and Load-Dependent Cost

This section is dedicated to the VRPSDPTW-LDC, the basic VRPTW variant that includes goods deliveries and pickups together with a load-dependent cost function per arc. This model is obviously also applicable in case of sole deliveries, sole collection, and single demand (customers have either a delivery or a pickup quantity) as well as mixed demand (both is allowed), see (Gribkovskaia and Laporte, 2008) for a classification scheme. We first present definitions for load and weight in Section 3.1 and discuss the classes of cost functions considered in this paper in Section 3.2. In Section 3.3, we formally define the VRPSDPTW-LDC.

3.1. Vehicle's Weight and Onboard Load

It is important to distinguish between the weight of a vehicle and its onboard load. Both weight and load are masses and therefore measured in kilogram [kg] or (metric) ton [t]=[10³· kg]. The *payload* q_{ij} is the mass of cargo carried when traveling along the arc (i, j) , which is dependent of the sequence in which the vehicle visits the customers (see Section 3.3). For the sake of brevity, we often speak of *load* instead of payload.

There are different weights relevant for a given vehicle. The *curb weight* w_0 refers to the total mass of a vehicle that is ready to operate, i.e., a vehicle with standard equipment and all necessary operating consumables such as motor and transmission oil, brake fluid, coolant, and (sometimes included) a full tank of fuel. The *operating weight* w_{ij} of the vehicle is the sum of its curb weight w_0 and its payload q_{ij} when traveling between i and j . Also, the operating weight is dependent on the route traveled. The *gross vehicle weight* w_{max} refers to the maximum operating weight of a vehicle as specified by the manufacturer, including curb weight, payload, and driver, but excluding that of any possibly attached trailers.

The vehicle capacity Q , which limits the maximum payload, can be derived by subtracting the curb weight w_0 and driver weight w_{drv} from the gross vehicle weight w_{max} , i.e., $Q = w_{max} - w_0 - w_{drv}$. This capacity may be further restricted by traffic regulations. Using the above definitions, it is straightforward to convert the payload $q_{ij} \in [0, Q]$ to the operating weight $w_{ij} \in [w_0, w_{max} - w_{drv}]$ via $w_{ij} = w_0 + q_{ij}$. Note that the literature (see Section 2) is not uniform and specifies the routing cost per arc by either the payload or the operating weight.

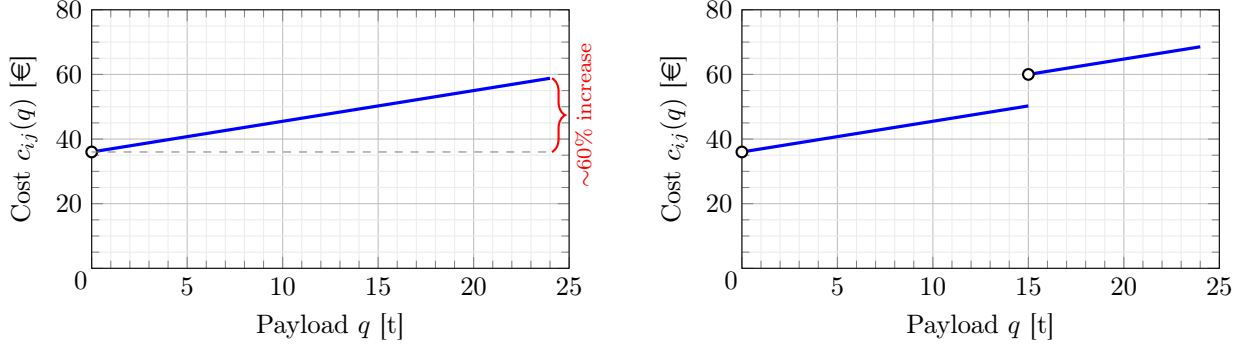
Later, we will also consider vehicles of different types $k \in K$. In this case, vehicle type-specific weights and coefficients are written with a superscript k , i.e., w_0^k for the curb weight, w_{max}^k for the gross weight, and Q^k for the capacity.

3.2. Cost Functions

We assume that the VRP under consideration is defined over a simple directed graph $D = (V, A)$ and that the routing cost $c_{ij}(q)$ per arc $(i, j) \in A$ is strictly positive and monotonically non-decreasing in payload q (to lighten the notation, we omit the arc index from now on). The considered cost functions can account for the distance between two locations that i and j refer to, characteristics of the road, the fuel consumption rate of the truck, and the payload q carried on the corresponding arc $(i, j) \in A$. In particular, when traveling along an arc (i, j) without any payload, i.e., the operating weight is the curb weight, the cost incurred is given by the intercept $\underline{c}_{ij} := c_{ij}(0)$ of the cost function. For the basic continuous linear case, we consider a linear load-dependent cost function of the form

$$c_{ij}(q) = \underline{c}_{ij} \cdot (1 + q/Q \cdot \rho), \quad (1)$$

where ρ is a non-negative parameter in percent, defining the influence of the load on the cost. More succinctly, it displays the percentage-wise increase in the arc's routing cost when a vehicle is fully loaded, i.e., $c_{ij}(Q) = \underline{c}_{ij}(1 + \rho)$ equivalent to $\rho = (c_{ij}(Q) - c_{ij}(0))/c_{ij}(0)$, see Figure 1a. This linearity assumption is justified by the studies of Xiao et al. (2012) and Bousonville et al. (2022), who observed (empirically) a linear relation between the fuel consumption rate of a vehicle and its operating weight $w_{ij} = w_0 + q$ (implying linearity also for payload and cost).



(a) Linear cost function $c_{ij}(q)$ with routing cost of $\underline{c}_{ij} = 36$ when the vehicle travels empty (with load $q = 0$ and curb weight w_0) and a load-impact factor of $\rho = 60\%$.

(b) Piecewise linear cost function with a discontinuous cost increase at payload $q = 15$ and segments having base cost multipliers $m_1 = 1$ and $m_2 = 1.625$ and slopes $\delta_1 = \delta_2 = 0.95 = 19/20$.

Figure 1: Example of cost functions $c_{ij}(q)$ dependent on payload $q \in [0, 24] = [0, Q]$.

Discontinuities. In the introduction, we already mentioned different reasons why cost functions may be discontinuous: First, itineraries between two points i and j are typically computed by navigation or geographical information systems using a shortest-path algorithm over a road network of much larger detail than the digraph $D = (V, A)$. In these underlying networks, road segments are equipped with additional attributes that allow to decide whether a truck may or may not traverse this segment, e.g., due to its weight, width, height, and number of axles. Some roads may not allow trucks pulling a trailer. As a result, when a vehicle does not comply with one of the restrictions, it must be re-routed along a ‘detour’ that avoids said bridge or road. Note that in many areas, bridges, tunnels, and roads have such restrictions, some of which might be weight-dependent. Then, the underlying shortest path problem that computes the actual itinerary becomes load-dependent. This can be modeled by a set of parallel arcs between i and j , each specifying a cost and load limit (similar to time and distance tradeoffs considered in Ben Ticha et al., 2018). We do not follow this modeling option but directly integrate re-routing requirements into a single cost function.

A second reason for discontinuities is related to trucks equipped with a lift axle, which must be lowered when reaching a specified weight limit. The lowering of the axle increases costs through increased fuel consumption due to higher rolling resistance, increased tire wear and, in some cases, tolls. In some US states for example, tolls are calculated by number of axles with ground contact. Lift axles are therefore only counted when lowered, saving costs when traveling with a light load (NACFE, 2024). For a vehicle with a lift axle, its cost function will show a discontinuity at the payload level at which it must lower the axle. An example is shown in Figure 1b.

We formalize piecewise linear cost functions with S pieces by defining (integer) breakpoints $0 = Q_1 < Q_2 < \dots < Q_S \leq Q_{S+1} = Q$ and intervals $I_s = [Q_s, Q_{s+1})$ for $1 \leq s < S$ and $I_S = [Q_S, Q_{S+1}]$ (all half-open except for the last interval which is closed). Note that the S intervals are non-overlapping and that they partition the domain $[0, Q]$ of the cost function. Furthermore, for each interval I_s , $1 \leq s \leq S$, let

- m_s be the *base cost multiplier* at Q_s , i.e., the cost is $m_s \cdot \underline{c}_{ij}$ when the load is equal to Q_s ;
- δ_s be the (non-negative) *slope* indicating the per-unit cost increase within the segment. The relationship between the load-impact factor ρ and a slope δ is $\delta = \rho/Q$.

Then, the S triplets (Q_s, m_s, δ_s) for $1 \leq s \leq S$ uniquely define

$$c_{ij}(q) = \underline{c}_{ij} \cdot (m_s + (q - Q_s) \cdot \delta_s) \quad \text{for } q \in I_s. \quad (2)$$

(note the similarity to eq. (1)). In Figure 1b, the two pieces are defined with the same basic data as in Figure 1a, i.e., $\underline{c}_{ij} = 36$, and the two triplets $(Q_1, m_1, \delta_1) = (0, 1, 0.95)$ and $(Q_2, m_2, \delta_2) = (15, 1.6, 0.95)$ (note that $c_{ij}(15) = 60$).

Third and lastly, when toll-by-weight schemes such as the ones used in China (see Shen et al., 2009) are considered, cost functions are no longer linear and have several pieces. It is no problem to approximate such a cost function with a piecewise linear function, where the quality of the approximation increases with the number of pieces allowed in the approximation. Section 5.3 analyzes the impact of having more pieces on the computational burden.

Heterogeneous Fleet. For heterogeneous fleets with different types $k \in K$ of vehicles, we can derive arc and vehicle type-specific cost functions $c_{ij}^k(q)$ in the same manner as explained above. For this purpose, the triplets become $(Q_s^k, m_s^k, \delta_s^k)$ for $1 \leq s \leq S^k$. Recall that vehicle type-specific capacities are denoted by Q^k for $k \in K$. To be able to compare vehicles of different types, we also use a *base capacity* Q , which we define as the capacity given in the VRP instance, see Section 5.

3.3. Problem Description

The VRPSDPTW-LDC is defined on a complete and simple directed graph $D = (V, A)$. The vertices V comprise the set of customers N and two copies o and o' of the depot, which represent the start and end point of each route. Each customer $i \in N$ has a non-negative delivery demand d_i and pickup demand p_i , and a time window $[e_i, \ell_i]$ representing the feasible service start times. Additionally, we define $d_o = d_{o'} = p_o = p_{o'} = 0$ and a planning horizon $[e_o, \ell_o] = [e_{o'}, \ell_{o'}]$ for both nodes representing the depot.

As in the VRPTW, each arc $(i, j) \in A$ is associated with a travel time t_{ij} . We assume that it includes the service duration at customer i . A route is a path $P = (i_0, i_1, \dots, i_h)$ in D from $i_0 = o$ to $i_h = o'$ of length $h \geq 2$. A path P respects time window constraints if there exists a schedule $(T_0^{time}, T_1^{time}, \dots, T_h^{time}) \in \mathbb{R}^{h+1}$ with $T_j^{time} \in [e_{i_j}, \ell_{i_j}]$ for all $j \in \{0, 1, \dots, h\}$ and $T_j^{time} + t_{i_j, i_{j+1}} \leq T_{j+1}^{time}$ for all $j \in \{0, 1, \dots, h-1\}$.

In the computational studies, we will consider both homogeneous and heterogeneous fleets. Thus, in general, each vehicle type $k \in K$ consists of n^k vehicles of that type (this number can be restrictive or not). The vehicle types differ in their maximum capacities Q^k and their load-dependent non-decreasing cost functions $c_{ij}^k(q)$ for all $(i, j) \in A$. For a vehicle of type $k \in K$, the capacity constraint is fulfilled if $\sum_{j=0}^g p_{i_j} + \sum_{j=g+1}^h d_{i_j} \leq Q^k$ for all $g \in \{0, 1, \dots, h-1\}$. A route r is feasible if it respects the time-window and capacity constraints. The cost of a route is

$$c_r^k = \sum_{g=0}^{h-1} c_{i_g, i_{g+1}}^k(q_{i_g, i_{g+1}}) \quad \text{with} \quad q_{i_g, i_{g+1}} = \sum_{j=0}^g p_{i_j} + \sum_{j=g+1}^h d_{i_j}. \quad (3)$$

A set of routes is a feasible solution to the VRPSDPTW-LDC if every customer is included in exactly one of the routes, all routes are feasible, and not more than n^k routes of type k are in the set for all $k \in K$. A feasible set of routes is an optimal solution to the VRPSDPTW-LDC if the sum of the cost of all routes is minimal.

4. Branch-Price-and-Cut Algorithm for the VRPSDPTW-LDC

We employ a BPC algorithm to solve the VRPSDPTW-LDC since these are the leading exact algorithms for solving many variants of the VRP (Costa et al., 2019). We introduce an extended set-partitioning model for the VRPSDPTW-LDC in Section 4.1, which is the *master problem* (MP) in the BPC approach. Section 4.2 presents the column-generation procedure, in particular, we highlight the necessary adjustment (compared to the *VRPSDP with time windows* (VRPSDPTW) with static cost) to handle load-dependent cost of the forward, backward, and bidirectional labeling. Additionally, we discuss the acceleration techniques used in the labeling algorithm. In Section 4.3, valid inequalities and the applied branching rules used to obtain integer solutions are presented.

4.1. Route-based Formulation

For each vehicle type $k \in K$, let Ω^k denote the set of all feasible VRPSDPTW-LDC routes. The following route-based formulation uses binary variables λ_r^k to indicate whether a route $r \in \Omega^k$ of type $k \in K$ is chosen. Additionally, let the integer coefficient b_{ir} indicate how often route $r \in \Omega^k$, $k \in K$ includes vertex $i \in V$.

$$\min \quad \sum_{k \in K} \sum_{r \in \Omega^k} c_r^k \lambda_r^k \quad (4a)$$

$$\text{subject to} \quad \sum_{k \in K} \sum_{r \in \Omega^k} b_{ir} \lambda_r^k = 1 \quad \forall i \in N \quad (4b)$$

$$\sum_{r \in \Omega^k} \lambda_r^k \leq n^k \quad \forall k \in K \quad (4c)$$

$$\lambda_r^k \in \{0, 1\} \quad \forall k \in K, r \in \Omega^k \quad (4d)$$

The objective (4a) is the minimization of the total cost of all routes. Constraints (4b) enforce that each customer is served exactly once. Constraints (4c) restrict the number of routes of each vehicle type to the fleet size. The domain of the routing variables is specified in (4d).

Model (4) is an extended set-partitioning formulation that typically contains a very large number of variables, since the number of feasible routes grows quickly in $|N|$ when the capacity and time-window constraints are not strongly binding. Hence, instead of solving the model (4) directly with a general purpose MIP solver, a BPC approach is used. Herein, model (4) is the MP, while the *linear-programming* (LP) relaxation restricted to a (small) subset of feasible routes $\bar{\Omega}^k \subset \Omega^k$ is denoted as the *restricted master program* (RMP). *Column generation* (Desaulniers et al., 2005; Desrosiers et al., 2025) iterates between solving the RMP and the vehicle type-specific pricing problems that provide routes with negative reduced costs to be added to the RMP. The column-generation process terminates when no more routes with negative reduced cost exist.

4.2. Column Generation

The pricing decomposes into $|K|$ column-generation subproblems, one for each vehicle type $k \in K$. In this section, we assume that a vehicle type $k \in K$ is given and fixed. Moreover, let $(\pi_i)_{i \in N}$ denote the dual prices of the set-partitioning constraints (4b) and let μ_k be the dual price of the fleet-size constraint (4c) for vehicle type k . For convenience, we define $\pi_o := \pi_{o'} := \mu_k/2$. Then, the reduced cost of an arbitrary route $r \in \Omega^k$ is

$$\bar{c}_r^k = c_r^k - \sum_{i \in V} b_{ir} \pi_i,$$

where c_r^k is load-dependent as defined by (3). The load-dependency of the (reduced) cost distinguishes the VRPSDPTW-LDC pricing problem from other standard SPPRCs. It is *not* possible to assign a constant cost to each arc of the network $D = (V, A)$.

We will show that the pricing problem can be modeled as an SPPRC (Irnich and Desaulniers, 2005) and solved using a bidirectional dynamic-programming labeling algorithm over D . Recall that labeling algorithms construct partial paths that are iteratively extended. In forward labeling, a first partial path is the trivial path $P_o = (o)$ that is extended, arc by arc, into partial paths of the form $P_i = (o, \dots, i)$ ending at a vertex i . *Resource extension functions* (REFs, Desaulniers et al., 1998; Irnich, 2008) describe how the attributes describing a partial path $P_i = (o, \dots, i)$ are to be updated when the path is extended along an arc $(i, j) \in A$, resulting in the new path $P_j = (o, \dots, i, j)$. In VRPs with deliveries, the payload to transport along each arc of the partial path is such an attribute. However, the payload is not known precisely during the extension and can only be determined when the last customer is reached. The attributes depend on the completion of the partial path, i.e., which customers are visited in the completion. Hence, also the reduced cost of the partial path is not known precisely. Symmetrically, backward labeling starts with a partial path $P_{o'} = (o')$ and partial paths $P_j = (j, \dots, o')$ are extended along arcs $(i, j) \in A$ in opposite direction into partial paths $P_i = (i, j, \dots, o')$. Here, all load-related information is known when only deliveries occur.

Conversely, for VRPs with pickups, the payload is always known precisely in forward labeling, but not in backward labeling.

When considering the VRPSDP variant where both pickup demand is shipped from customer to end depot o' and delivery demand is shipped from start depot o to customer, the load is not known in either labeling direction. This complicates feasibility checks, dominance conditions, and merging forward and backward labels. In the following sections, we discuss the forward and backward extensions of labels and the handling of these unknown quantities in the bidirectional labeling.

4.2.1. Forward Labeling

A forward label F_i represents a partial path $P_i = (o, \dots, i)$ and includes several attributes (a.k.a. resources) to describe the state in which the vehicle is. For the VRPSDPTW-LDC, we use the following attributes:

$F_i^{rdc}(\bar{q})$: Reduced cost function of P_i , depending on the unknown delivery demand $\bar{q} = \bar{d}$

F_i^{time} : Earliest start time of service at the last vertex i

$F_i^{visit,n}$: Number of times the customer $n \in N$ is visited in P_i

F_i^{pick} : Total load picked up from customers on P_i

F_i^{ml} : Maximum load on the vehicle along P_i or after leaving the last vertex i

The resources F_i^{time} and $F_i^{visit,n}$ for $n \in N$ are standard resources in a VRPTW (Desaulniers et al., 2014). In a VRPSDP, the interdependent resources F_i^{pick} and F_i^{ml} are both needed to check for feasibility regarding capacity (Halse, 1992; Irnich, 2008).

The initial label for the trivial path $P_o = (o)$ including only the origin depot is given by $F_o = (F_o^{rdc}(\bar{q}), F_o^{time}, (F_o^{visit,n})_{n \in N}, F_o^{pick}, F_o^{ml}) = ((\bar{q} \mapsto 0), e_o, \mathbf{0}, 0, 0)$, where $(\bar{q} \mapsto 0)$ is the null function and $\mathbf{0}$ the null vector of dimension $|N|$.

To propagate the attributes of a forward label F_i over an arbitrary arc $(i, j) \in A$, thus creating a new label F_j at vertex j for the corresponding partial path $P_j = (o, \dots, i, j)$, we define the following REFs:

$$F_j^{rdc}(\bar{d}) = F_i^{rdc}(d_j + \bar{d}) + c_{ij}^k(F_i^{pick} + d_j + \bar{d}) - (\pi_i + \pi_j)/2, \quad \text{for all } \bar{d} \in [0, Q^k - F_j^{ml}] \quad (5a)$$

$$F_j^{time} = \max\{e_j, F_i^{time} + t_{ij}\} \quad (5b)$$

$$F_j^{visit,n} = \begin{cases} F_i^{visit,n} + 1, & \text{if } n = j \\ F_i^{visit,n}, & \text{otherwise} \end{cases}, \quad \text{for all } n \in N \quad (5c)$$

$$F_j^{pick} = F_i^{pick} + p_j \quad (5d)$$

$$F_j^{ml} = \max\{F_i^{pick} + p_j, F_i^{ml} + d_j\} \quad (5e)$$

The partial path P_j is feasible if $F_j^{time} \leq \ell_j$, $F_j^{visit,n} \leq 1$ for all $n \in N$, $F_j^{pick} \leq Q^k$, and $F_j^{ml} \leq Q^k$, i.e., testing the as-early-as-possible schedule, elementarity, and the capacity. Regarding the propagation of reduced costs via (5a), the payload on each arc of the partial path P_j is unknown due to the additional delivery demand that has to be transported from the origin depot o to the respective customer of a feasible extension. Therefore, we describe the unknown payload with the symbol $\bar{d}$ to make this aspect clearer. The REF (5a) can then be interpreted as follows: the delivery demand d_j becomes the precisely known part of the unknown delivery demand so that the function F_i^{rdc} must be shifted to the left by this amount. In addition, on the arc (i, j) extending P_i , the known payload $F_i^{pick} + d_j$ together with the unknown delivery demand $\bar{d}$ must be transported. Lastly, the dual prices $(\pi_i + \pi_j)/2$ can be attributed to the added arc (i, j) . The domain of the new REF (5a), i.e., the new unknown delivery demand, is restricted to a value not exceeding $Q^k - F_j^{ml}$, which can be interpreted as the resulting residual capacity. In summary, the new REF (5a) is the result of three types of operations, i.e.,

- (1) left shift applied to the load-dependent arc cost function $c_{ij}^k(\cdot)$ and to the label's cost function $F_i^{rdc}(\cdot)$,
- (2) addition of the two resulting functions, and
- (3) addition/subtraction of a constant.

We will later use two classes of functions that are closed under these three types of operations. The first class is *linear functions* as the one depicted in Figure 1a. The second class is piecewise linear functions with possible discontinuities as the one depicted in Figure 1b. Here, operations (1) can decrease the number of pieces, operations (2) can increase it (to the sum of the number of pieces), while operations (3) are neutral. Other classes of functions may better approximate non-linear load-to-cost relationships. Examples of such classes closed under operations (1) to (3) are *splines* (Schumaker, 2007) and *Fourier series* (Seeley, 2006).

Dominance tests avoid the enumeration of all feasible paths. Since all forward REFs (5) are non-decreasing and resources are only bounded from above, standard dominance rules are applicable (Irnich and Desaulniers, 2005). Let labels F_i and F'_i be two labels referring to the same vertex $i \in V$. Then, F_i dominates F'_i if

$$F_i^{rdc}(\bar{d}) \leq F'_i{}^{rdc}(\bar{d}) \quad \text{for all } \bar{d} \in [0, Q^k - F'_i{}^{ml}], \quad (6)$$

$F_i^{time} \leq F'_i{}^{time}$, $F_i^{visit,n} \leq F'_i{}^{visit,n}$ for all $n \in N$, $F_i^{pick} \leq F'_i{}^{pick}$, and $F_i^{ml} \leq F'_i{}^{ml}$. Conditions (6) can be checked in different ways. Assuming integer demands, the trivial test is to compare $F_i^{rdc}(\bar{d})$ and $F'_i{}^{rdc}(\bar{d})$ for every integer point $\bar{d} \in \{0, 1, \dots, Q^k - F'_i{}^{ml}\}$. If $F_i^{rdc}(\bar{d})$ and $F'_i{}^{rdc}(\bar{d})$ are linear function, it suffices to implicitly test dominance using the points $\bar{d} = 0$ and $\bar{d} = Q^k - F'_i{}^{ml}$. If they are piecewise linear functions, additional tests at every break point are sufficient. Even non-linear functions allow for efficient implicit tests as exemplified for quadratic cost functions in (He et al., 2019). Here, the implicit test boils down to determine all intersection points of two functions. Since dominance tests are usually the bottleneck of the BPC algorithm, it is preferable to use efficient implicit dominance tests rather than trivial tests, the latter of which are typically much slower. Our implementation evaluated in Section 5 uses implicit dominance tests.

4.2.2. Backward Labeling

One of the most important acceleration techniques for labeling algorithms is to apply bidirectional labeling (Righini and Salani, 2006). Bidirectional labeling algorithms use both forward and backward labeling algorithms, in which labels are respectively extended until a halfway point, at which they are merged. We present the backward labeling now, which is particularly interesting for VRPs with collection, since the precise quantity picked up along an extension of a backward partial path is not known precisely.

A backward label B_j represents a partial path $P_j = (j, \dots, o')$ and includes corresponding attributes to describe in which state a vehicle is or has to be.

$B_j^{rdc}(\bar{q})$: Reduced cost function of P_j , depending on the unknown pickup demand $\bar{q} = \bar{p}$

B_j^{time} : Latest start time of service at the first vertex j

$B_j^{visit,n}$: Number of times the customer $n \in N$ is visited in P_j

B_j^{ml} : Maximum load on the vehicle along P_j or before reaching the first vertex j

B_j^{dlvr} : Total load delivered to customers on P_j

As in the forward labeling, the resources B_j^{time} and $B_j^{visit,n}$ for $n \in N$ are known from the VRPTW, and the two resources B_j^{dlvr} and B_j^{ml} from the VRPSDP as detailed in (Irnich, 2008). The initial label for the trivial path $P_{o'} = (o')$ including only the destination depot is given by $B_{o'} = (B_{o'}^{rdc}, B_{o'}^{time}, (B_{o'}^{visit,n})_{n \in N}, B_{o'}^{ml}, B_{o'}^{pick}) = ((\bar{q} \mapsto 0), \ell_{o'}, \mathbf{0}, 0, 0)$.

To propagate the attributes of a label B_j backward over an arbitrary arc $(i, j) \in A$, thus creating a new label B_i at vertex i for the corresponding partial path $P_i = (i, j, \dots, o')$, we define the following backward REFs:

$$B_i^{rdc}(\bar{p}) = B_j^{rdc}(p_i + \bar{p}) + c_{ij}^k(B_j^{dlvr} + p_i + \bar{p}) - (\pi_i + \pi_j)/2, \quad \text{for all } \bar{p} \in [0, Q^k - B_i^{ml}] \quad (7a)$$

$$B_i^{time} = \min\{B_j^{time} - t_{ij}, \ell_i\} \quad (7b)$$

$$B_i^{visit,n} = \begin{cases} B_j^{visit,n} + 1, & \text{if } n = i \\ B_j^{visit,n}, & \text{otherwise} \end{cases}, \quad \text{for all } n \in N \quad (7c)$$

$$B_i^{ml} = \max\{B_j^{dlvr} + d_i, B_i^{ml} + p_i\} \quad (7d)$$

$$B_i^{dlvr} = B_j^{dlvr} + d_i \quad (7e)$$

The partial path P_i is feasible if $B_i^{time} \geq e_i$, $B_i^{visit,n} \leq 1$ for all $n \in N$, $B_i^{dlvr} \leq Q^k$, and $B_i^{ml} \leq Q^k$, i.e., using the as-late-as-possible schedule and respecting elementarity and the capacity. In the backward case, the payload on the vehicle on each arc is not known precisely due to the unknown pickup demand $\bar{p}$ that is carried back to the destination depot o' from each additional customer that the partial path is extended to. Note that we describe the unknown payload in (7a) with the symbol $\bar{p}$, since it is the quantity that is picked up along the extension before arriving at vertex i .

Also, the dominance checks are defined similarly to the forward labeling. Let labels B_j and B'_j be two labels referring to the same vertex $j \in V$. Then, B_j dominates B'_j , if

$$B_j^{dc}(\bar{p}) \leq B'_j{}^{dc}(\bar{p}) \quad \text{for all } \bar{p} \in [0, Q^k - B'_j{}^{ml}], \quad (8)$$

$$B_j^{time} \geq B'_j{}^{time}, B_j^{visit,n} \leq B'_j{}^{visit,n} \text{ for all } n \in N, B_j^{dlvr} \leq B'_j{}^{dlvr}, \text{ and } B_j^{ml} \leq B'_j{}^{ml}.$$

4.2.3. Bidirectional Labeling

Let $P_i = (o, \dots, i)$ be a forward partial path with associated label F_i , and likewise, let $P_j = (j, \dots, o')$ be a backward partial path with associated label B_j . Then, the concatenation $P = (P_i, P_j) = (o, \dots, i, j, \dots, o')$ over the arc $(i, j) \in A$ is a feasible path if and only if

$$F_i^{time} \leq B_j^{time}, \quad (9a)$$

$$F_i^{visit,n} + B_j^{visit,n} \leq 1, \quad \text{for all } n \in N \quad (9b)$$

$$F_i^{pick} + B_j^{ml} \leq Q^k, \quad \text{and} \quad (9c)$$

$$F_i^{ml} + B_j^{dlvr} \leq Q^k. \quad (9d)$$

Herein, the two capacity constraints (9c) and (9d) have a left-hand side that describes the maximum load on P_j and P_i within P , respectively. The reduced cost of the complete path P resulting from the merge can be calculated by

$$\tilde{c}_r^k = F_i^{rdc}(B_j^{dlvr}) + \left(c_{ij}(F_i^{pick} + B_j^{dlvr}) - (\pi_i + \pi_j)/2 \right) + B_j^{rdc}(F_i^{pick}), \quad (9e)$$

where the first term is the reduced cost on P_i , the second the reduced cost on the arc (i, j) , and the third the reduced cost on P_j .

Constraints and equations (9) describe a merge over an arc $(i, j) \in A$. It is typically more efficient to perform a merge at a vertex i , i.e., using forward and backward labels that both refer to the same merge vertex $i \in N$. In our case, the forward and backward attributes *pick*, *dlvr*, and *ml* already anticipate what happens at the merge vertex i . The respective conditions would have to correct the otherwise double counting at the merge vertex. Therefore, it is simpler to test the merge of F_i and B_i by using a merge over the arc (i, j) considering F_i and the predecessor label $\text{pred}(B_i)$, assuming that the latter refers to the vertex j . Of course, using the predecessor label $\text{pred}(F_i)$ and B_i is also possible.

The effectiveness of bidirectional labeling results from limiting the extension of all labels using a so-called half-way point (Righini and Salani, 2006). The idea is to choose a monotone resource such as *time* and a half-way point value H (in the “middle” of the time horizon). Forward labels F_i are only extended if $F_i^{time} \leq H$, and backward labels B_j are only extended if $B_j^{time} > H$ (or with reversed roles in case of $= H$). To improve computational efficiency, we use the dynamic half-way point technique for *time* as suggested by Tilk et al. (2017). The dynamic choice of H better balances the forward and backward labeling effort.

Finally, we mention that the load-related attributes can also be modified to become the monotone resource, as described by Bianchessi et al. (2023). To this end, B_j^{ml} and B_j^{dlvr} have to be replaced by residual capacities $Q^k - B_j^{ml}$ and $Q^k - B_j^{dlvr}$, respectively. Moreover, to not double count deliveries and pickups at a vertex, the propagation along an arc (i, j) has to be delayed, e.g., in the backward REF. Here, instead of considering d_i and p_i in (7d) and (7e), the demands d_j and p_j have to be used. Since the capacity is not always a strongly binding resource in the computational study presented in Section 5, we refrain

from using load-related attributes as the monotone resource. Instead, we refer to (Bianchessi et al., 2023, Sections 3.3.1 and 6.2) for further explanation and detailed formulas.

4.2.4. Acceleration of Labeling

In addition to the use of bidirectional labeling, we employ the following acceleration techniques. Feillet et al. (2004) replace $F_i^{visit,n}$ by new attributes that set the value to 1 if a customer n is unreachable, e.g., due to time-window or load-related constraints.

Instead of solving the *elementary* SPPRC, Baldacci et al. (2011) suggest solving a computationally easier *ng*-path relaxation. Given neighborhoods $(N_i)_{i \in V}$ which are all subsets of N , a path $P = (i_0, i_1, \dots, i_h)$ is feasible if for each cycle $(i_j, \dots, i_\ell)$ with $i := i_j = i_\ell$ in P (if any), there exists an index p with $j < p < \ell$ and $i \notin N_p$. The choice of the neighborhoods controls the trade-off between a tighter linear relaxation of the MP (for larger neighborhoods) and faster labeling (for smaller neighborhoods). Since the *ng*-relaxation uses a limited rather than full memory of previously visited nodes, the resource $F_i^{visit,n}$ must be replaced with a new resource reflecting this change. The REFs (5c) and (7c) must be adjusted resetting $F_j^{visit,n}$ and $B_i^{visit,n}$ to 0, if $j \notin N_n$ or $i \notin N_n$, respectively. In our BPC implementation, we use *ng*-neighborhoods of size 10, because these have lead to a comparatively fast computation time in preliminary testing and in (Bianchessi et al., 2023).

Our implementation also uses a one-dimensional bucketing on the *time* resource as suggested by Sadykov et al. (2021). Pricing heuristics (a.k.a. partial pricing) are also employed (Desaulniers et al., 2008), where instead of the full digraph, arc-reduced networks are constructed. In these networks, we allow two, five, ten or fifteen arcs per customer vertex, respectively. As a result, the full pricing subproblem is only solved exactly when the faster heuristic fails to find a route with negative reduced cost.

4.3. Cutting and Branching

The cutting component of a BPC algorithm strengthens the linear relaxation of the MP by adding violated valid inequalities. These cuts are of two types, robust and non-robust cuts (Desrosiers et al., 2025). For VRPSDPTW-LDC instances with a homogeneous fleet, we use the robust *capacity cuts* as suggested in (Subramanian et al., 2012). They are also valid for heterogeneous fleets with identical capacity. Otherwise, for heterogeneous instances with different capacities, they cannot be applied.

Limited-memory subset-row inequalities (lm-SRIs, Pecin et al., 2017) are non-robust cuts that are often very effective. We only use subsets referring to three rows. Note that the introduction of each violated lm-SRI into the RMP requires the addition of a (binary) resource to the labels. We separate violated lm-SRIs only at the root node of the branch-and-bound tree, and add only those with a violation exceeding 0.05.

To ensure integrality, we embed the column-generation approach into a branch-and-bound search tree. The tree is explored using a best-bound-first node-selection strategy to encourage small search trees and tight lower bounds when the BPC algorithm is terminated before proving optimality. As in many other BPC algorithms, we use a two-level hierarchical branching scheme. We prioritize branching on the number of vehicles and select the vehicle type $k \in K$ for which the branching variable has a fractional part closest to 0.5. On the second level, we branch on arcs, i.e., an arc (i, j) with flow value closest to 0.5 is chosen, and in the first branch, the arc (i, j) is removed. In the second branch, the arc is enforced by removing incompatible arcs (other arcs entering j or leaving i). Both rules must also be implemented into the RMP where all routes that use forbidden arcs are temporarily removed from the RMP.

5. Computational Results

In this section, we detail the design of the computational experiments and present their results. Section 5.1 describes the computational setup and the set of instances used, Section 5.2 analyzes the impact of the load-dependent extension to the BPC algorithm on computational efficiency, and Section 5.3 the impact of discontinuities in the cost functions.

5.1. Computation Setup and Instances

All algorithms are implemented in C++ using the callable library of CPLEX 22.1.2 and compiled into 64-bit single-thread release code with GCC 8.5.0. The computational study was performed on the high performance computing cluster MOGON KI of the Johannes Gutenberg University Mainz. The cluster consists of several AMD EPYC 7713 processors running at 2.0 GHz (the performance of a single thread is slightly lower than that of a standard personal computer). In all calls to CPLEX, default values of all parameters are kept except for setting the number of available threads to one. Moreover, the RMPs are re-optimized with the primal simplex algorithm of CPLEX. The time limit was set to one hour unless stated otherwise.

To study the impact of the load-dependent routing cost, we consider the VRPSDPTW benchmark instances with 100 customers of Solomon (1987), adapted by Wang and Chen (2012), available at <https://oz.nthu.edu.tw/~d933810/test.htm>. The 56 instances are classified by customer distribution as random (R), clustered (C), and mixed (RC). The instances can also be divided into Series 1 with the subsets R1, C1, and RC1 with tight constraints and Series 2 with the subsets R2, C2 and RC2 with loose constraints. The latter are on average harder to solve.

The computation of travel times t_{ij} and basic intercept $c_{ij}(0) = \underline{c}_{ij}$ of the cost function follows exactly the same rules as applied to Solomon’s VRPTW benchmark and the VRPSDPTW benchmark. Depot and customers have associated coordinates, from which the distances $\underline{c}_{ij}$ are computed as Euclidean distance and rounded up to one decimal place (for details, we refer to Kohl et al., 1999). The travel time t_{ij} is the sum of $\underline{c}_{ij}$ and the service time at i . The base capacity Q is the one defined in the Solomon instances.

In addition to these basic instances, we consider various load-dependent cost functions and fleets of vehicles. The fleets can be classified by number of vehicle types, number of pieces and their respective load-impact factor in the cost function as described in Section 3.2. The fleet files used to implement the representation of the cost functions for arcs can be found at <https://logistik.bwl.uni-mainz.de/research/benchmarks/> together with a description of their format.

5.2. Impact of Load-Dependent Cost Functions

In the first experiment, we study the impact of an increasing load-dependent routing cost to the performance of the BPC algorithm. Compared to the BPC algorithm for the VRPSDPTW, the VRPSDPTW-LDC must replace the reduced cost attribute, i.e., a single number, by a cost function. Even if this function is represented implicitly, it results in an increased effort. To measure this effort, we compare the original BPC implementation for the VRPSDPTW which does not support any load-dependent cost functions (denoted by “Base”) to the VRPSDPTW-LDC-tailored implementation that can handle load-dependencies (denoted by “LD”). In the latter, piecewise linear cost functions are used to store load-independent reduced cost values by setting the load-impact factor ρ to 0.

	C1 ($n = 9$)		R1 ($n = 12$)		RC1 ($n = 8$)		C2 ($n = 8$)		R2 ($n = 11$)		RC2 ($n = 8$)		Total ($n = 56$)	
	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time
Base	5	2363.5	12	525.4	6	1091.5	3	2511.1	4	2639.7	5	1392.3	35	1724.5
LD	3	2555.0	10	807.0	6	1239.5	3	2611.5	3	2702.1	5	1410.9	30	1866.0
Increase:		8.1%		53.6%		13.6%		4.0%		2.4%		1.3%		8.2%

Table 2: Impact of the load-dependent BPC extension on computational performance.

For the two otherwise identical BPC implementations, Table 2 shows the following key metrics:

Opt: the number of instances solved to proven optimality within the given time limit and

Time: the solution time in seconds (arithmetic average; instances not solved to optimality are taken into account with the time limit of 3600 seconds).

Comparing Base and LD, the results show a slight increase in computation time for LD due to the ability to handle cost functions and a heterogeneous fleet. This effect can be seen across all classes, although the increase in computation time varies between just 1.3 percent for RC2 and 53.6 percent for R1. For LD,

there is also a slight decrease in the number of instances solved to optimality within the one-hour time limit. Overall, the integration of the load-dependent cost functions and heterogeneous fleet results in a slight, acceptable deterioration of the performance of the more general BPC algorithm.

In the second experiment, we analyze the linear continuous cost functions $c_{ij}(q) = \underline{c}_{ij} \cdot (1 + q/Q \cdot \rho)$, in which the load-impact factor ρ is varied. The impact of ρ on the key measures Opt and Time can be seen in Table 3.

ρ [%]	C1 ($n = 9$)		R1 ($n = 12$)		RC1 ($n = 8$)		C2 ($n = 8$)		R2 ($n = 11$)		RC2 ($n = 8$)		Total ($n = 56$)	
	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time
0	3	2555.0	10	807.0	6	1239.5	3	2611.5	3	2702.1	5	1410.9	30	1866.0
20	3	2489.1	10	967.5	7	1343.4	6	1516.2	4	2761.6	6	1575.3	36	1783.4
40	4	2170.6	10	978.4	7	1603.1	7	1550.4	4	2586.9	6	1468.8	38	1727.0
60	9	1355.4	12	532.2	6	1559.9	4	2611.0	4	2435.1	6	1235.3	41	1582.5
80	9	329.8	12	873.1	6	1409.4	5	1984.6	5	2327.8	6	1119.4	43	1342.1
100	9	175.6	12	566.7	8	484.1	6	1219.3	5	2440.9	6	1126.5	46	1033.4
120	9	118.0	12	413.1	8	358.8	7	1038.7	5	2549.0	6	1153.3	47	972.6

Table 3: Effect of the load-impact factor ρ in the linear continuous load-dependent cost function $c_{ij}(q) = \underline{c}_{ij} \cdot (1 + q/Q \cdot \rho)$.

We note that the average solution time is often dominated by the time limit of 3600 seconds. However, the general trend is clear: The higher the load-impact factor ρ , the simpler is the solution for the BPC algorithm. The most difficult instances are those of classes R2 and RC2 where differences regarding ρ are less pronounced compared to the other classes. The general behavior is rather expected, and we interpret the results as follows. When the load-impact factor ρ increases, carrying high loads becomes increasingly expensive. This incentivizes shorter routes with a lower average load, which in turn simplifies the labeling. As a result, VRPSDPTW-LDC instances become easier to solve.

5.3. Impact of Discontinuities in the Cost Function

We also study the impact of discontinuities in the routing cost function on the computational performance. Even if these functions only have a single discontinuity, the labels can have functions with multiple discontinuities as a result of the addition operation (2) of two discontinuous functions, see Section 4.2.1. The number of discontinuities can however be comparatively small when many discontinuities are lost during the left shift (operation (1)) of the cost functions in the REFs (5a) and (7a), i.e., demands are high compared to the capacity.

To measure the impact of discontinuities, we compare five different homogeneous fleets for each of the given 56 instances. All fleets have a linear positive slope ($\delta_s^k = 1/Q$ for all $s \in S$) and differ in the load-dependent *cost functions* (CFs) that have

(CF₀): no discontinuities,

(CF₁₀): a single discontinuity at 10 percent of the instance's capacity Q ,

(CF₅₀): a single discontinuity at 50 percent of the instance's capacity Q ,

(CF₉₀): a single discontinuity at 90 percent of the instance's capacity Q , and

(CF_{multi}): multiple discontinuities at 30, 50, 70 percent of the instance's capacity Q .

We solve only the root node, setting the time limit to 3600 seconds. Table 4 reports the following key metrics as arithmetic averages grouped by instance class:

#Pcs: the average number S of pieces in the reduced cost function of all labels generated in the root node of the BPC algorithm;

Time_{rn}: the computation time for solving the root node in seconds (arithmetic average; instances not solved to optimality are taken into account with the time limit of 3600 seconds); and

#RS: the number of instances for which the root node was solved within the given time limit.

Metric	Cost function	C1 ($n = 9$)	R1 ($n = 12$)	RC1 ($n = 8$)	C2 ($n = 8$)	R2 ($n = 11$)	RC2 ($n = 8$)	Total ($n = 56$)
#Pcs	CF ₀	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	CF ₁₀	1.0	1.1	1.0	1.2	4.4	2.6	1.9
	CF ₅₀	2.9	4.9	3.7	5.9	10.2	7.8	6.0
	CF ₉₀	4.2	5.9	4.8	5.9	10.2	7.7	6.6
	CF _{multi}	5.7	10.8	7.6	13.7	28.4	21.2	14.9
Time _{rn}	CF ₀	9.3	14.8	15.2	713.4	440.8	969.0	333.8
	CF ₁₀	15.0	18.8	19.8	611.5	938.7	1040.1	429.6
	CF ₅₀	13.8	72.7	57.9	773.2	979.9	1091.0	484.9
	CF ₉₀	14.4	47.0	49.3	1317.1	1088.7	1091.8	577.4
	CF _{multi}	7.4	82.9	78.4	442.3	1273.5	1174.2	511.2
#RS	CF ₀	9	12	8	7	11	6	53
	CF ₁₀	9	12	8	8	11	6	54
	CF ₅₀	9	12	8	7	10	6	52
	CF ₉₀	9	12	8	7	9	6	51
	CF _{multi}	9	12	8	8	9	6	52

Table 4: Impact of discontinuity location in the load-dependent cost functions.

The results are clear: More pieces in the routing cost functions lead, on average, to even more pieces in the resulting label cost functions due to the addition operation (2). This is most evident in CF_{multi}, which reached an average of 28.4 pieces for the subset R2, which is an increase by a factor of more than seven. A deeper analysis of the results shows that some labels have cost functions with up to 76 pieces (for RDP204 in combination with CF_{multi}). The root node of the BPC for said instance could not be solved within the time limit, highlighting the trade-off between precision, generalization, and solvability.

A discontinuity in the middle or at the right-hand side (CF₅₀ and CF₉₀) leads to a higher average piece count than breakpoints further to the left (CF₁₀). In particular, the breakpoints at 10 percent of the instance’s capacity in CF₁₀ often completely disappear under the (first) left-shift operations applied during labeling. An increasing number of discontinuities also correlates with longer average root node solution times Time_{rn}. Likewise, more discontinuities also lead to a reduction in the number #RS of instances for which the root node could be solved within the time limit.

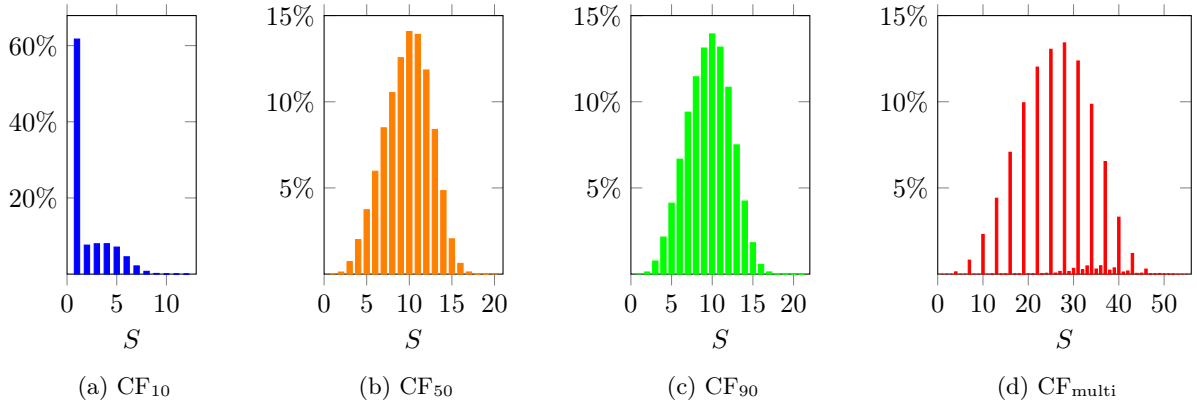


Figure 2: Percentage of labels with a cost function having S pieces for RCDP203. Note the different scales of the axes.

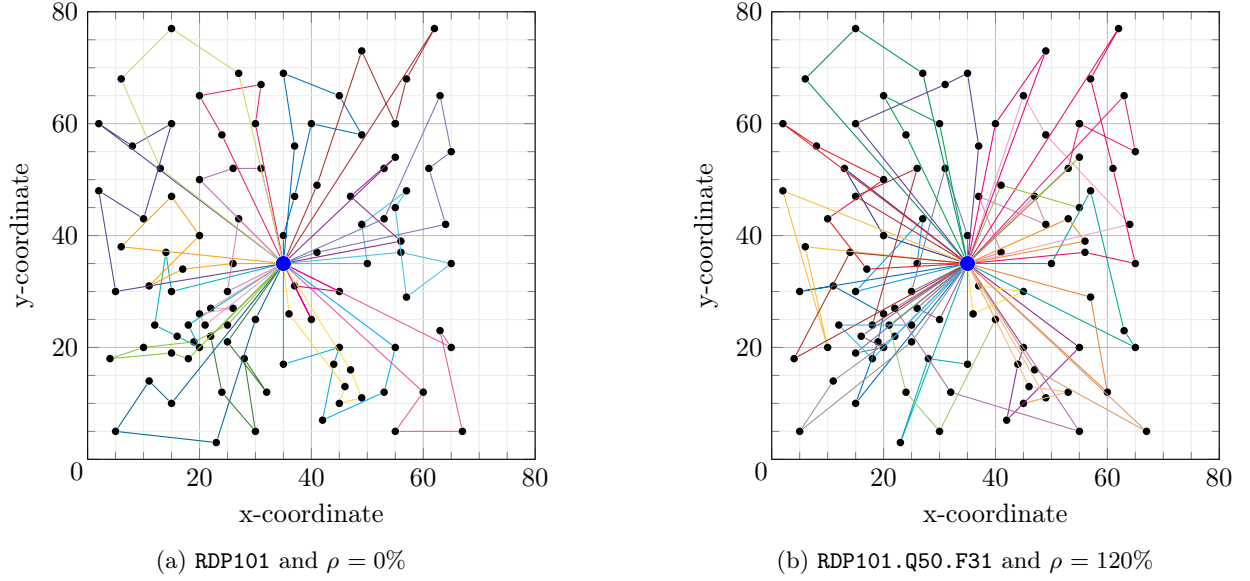


Figure 3: Optimal solutions of instance RDP101 in the unrestricted vs. restricted case.

We can make these general statements more concrete by considering, e.g., the instance RCDP203. For CF_{10} , CF_{50} , CF_{90} , and CF_{multi} , Figure 2 shows histograms regarding the number of pieces in the cost functions of all generated labels. The effect of vanishing breakpoints can be clearly seen from Figure 2a. The placement of the breakpoint in the middle (CF_{50}) or to the right (CF_{90}) does not produce a substantial difference (compare Figures 2b and 2c). For CF_{multi} , the number of breakpoints however increases drastically: up to 55 pieces occur in the cost functions.

Finally, it is interesting to point out that in the case of CF_{multi} , the labels' cost functions most often have $3n + 1$ pieces (for a positive integer n), i.e., $3n$ breakpoints. We can observe the repeated pattern 'large–small–small' of bars in Figure 2d. Note that the first propagated labels have exactly three breakpoints and four pieces. Due to discontinuities occurring at positions between 30 and 70 percent of an instance's capacity, left-shift operations eliminate only a very small fraction of breakpoints. This is particularly true because many customers have relatively small demands, which allows for many different, longer routes. Thus, the addition of two cost functions with $3i$ and $3j$ breakpoints tends to give a cost function with $3(i + j)$ breakpoints, resulting again in $3n + 1$ pieces for the integer $n = i + j$.

6. Managerial Insights

In this section, we analyze the impact of the load-impact factor on the structure of optimal solutions (Section 6.1) and present insights on cost savings comparing a homogeneous fleet of trucks with liftable axles and a heterogeneous fleet with only stationary axles (Section 6.2). For the following experiments, we use the same computational setup as before.

6.1. Impact of Fleet

To showcase the effect of the load-impact factor ρ on the structure of optimal solutions, we exemplarily consider the instance RDP101 with 100 customers that are distributed randomly, see Figure 3.

The fleet of the original instance RDP101 has $n = 25$ homogeneous vehicles, each with a capacity of $Q = 200$. This standard fleet size is not restrictive. Without load-dependent costs (i.e., $\rho = 0$), only 20 vehicles are needed in the optimal solution shown in Figure 3a. With increasing load-impact factor $\rho > 0$, the number of routes in the optimal solution rises slowly. In the extreme case with load-impact factor reaching 120 percent (recall that arc costs increase by this amount only when the vehicle is fully loaded),

only 24 vehicles are needed in the respective optimal solution. With the rising number of routes, the average load on each vehicle decreases so that the average and maximum capacity utilization are lower.

To also see the effect of the load-impact factor ρ for vehicles loaded close to full capacity, we define a modified instance RDP101.Q50 in which the number of vehicles remains unlimited, but the capacity is restrictive (set to $Q = 50$). Here, for $\rho = 0$ the optimal solution already utilizes 34 vehicles. This number increases to 38 vehicles for a load-impact factor of 120 percent. In addition, a second modified instance named RDP101.Q50.F31 is derived from the former by restricting the fleet size to $n = 31$. In preliminary tests, we tried different fleet-size limits and found that a fleet of 31 vehicles is the smallest possible.

For the three instances RDP101, RDP101.Q50, and RDP101.Q50.F31, Figure 4 shows the effect of the load-impact factor ρ on the four metrics: number of vehicles used, capacity utilization, total distance, and total cost. The figure depicts the optimal solution values in pink ($\bullet$), orange ($\blacksquare$), and cyan ($\blacklozenge$), respectively. As

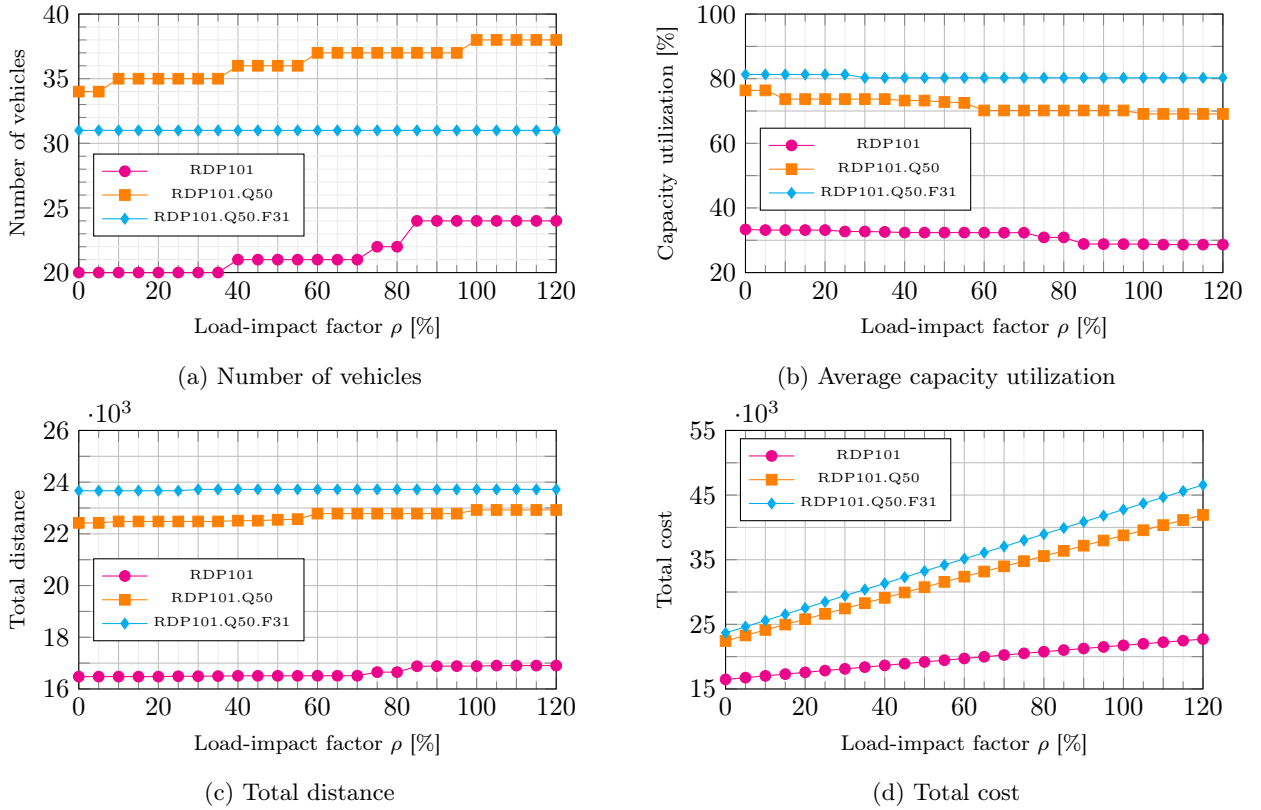


Figure 4: Effects of an increasing load-impact factor ρ on four solution metrics.

expected, the number of vehicles used, see Figure 4a, rises with a higher impact of load on cost. However, the increase remains moderate, with only four additional routes needed when doubling the arc costs for a fully loaded vehicle, i.e., for $\rho = 100$. For RDP101, the average load of approximately 30 percent of the capacity is substantially lower than what is probably expected, see Figure 4b. Further analyzes showed that for RDP101, the maximum capacity utilization is only 63 percent (not depicted). Similar observations can be made for most VRPSDPTW instances defined in (Wang and Chen, 2012). For the modified instances RDP101.Q50 and RDP101.Q50.F31, the average payload is closer to the capacity limit.

The total distance increases slightly with more, albeit shorter routes, see Figure 4c. Interestingly, the total cost rises almost linear in the load-impact factor ρ , see Figure 4d. This effect is amplified by restricted capacities and fleets, since then many arcs are traversed with a load that approaches the capacity so that almost the full load-dependent markup results. The only visible increase in distance occurs when an additional vehicle is included (compare Figures 4a and 4c, at $\rho = 80$ percent for RDP101 and at $\rho = 60$ percent

for RDP101.Q50). This increase is not visible in the total cost. We observe a similar behavior for many instances, which can be explained by the fact that the lower average load in the solution with an additional vehicle balances the increase in distance (compare Figures 4b and 4c).

6.2. Lifiable Axles

In this section, we analyze the use of heavy-duty commercial vehicles that are equipped with liftable axles, see Section 3.2. Axles can be raised when a vehicle is only partially loaded or empty, reducing rolling resistance and therefore both tire wear and fuel consumption. In practice, liftable axles are automatically lowered to the ground when a vehicle’s weight exceeds a certain threshold and retracted otherwise. These thresholds are often defined in national traffic laws. We will now compare fleets of vehicles with and without liftable axles.

Recall from the previous section that the original Solomon-based instances have a capacity that is almost never binding. In optimal solutions, the average capacity utilization is rather low (see Figure 4b). For the following experiment, we therefore drastically reduce the capacity to 25 percent of the original value given in the Solomon instances. In these new instances, the lift axles must be lowered when a realistic weight threshold is reached. We choose all cost functions or cost function pieces in this study so that weight-dependent factors approximate the impact of load on fuel costs.

We compare two different fleets. The first fleet is homogeneous, and all vehicles are equipped with lift axles (the fleet is denoted by “fleet_{lift}”). The vehicles have a capacity of Q^{lift} (which is a quarter of the original capacity specified in the Solomon instances, see above). Realistic load-impact factors can be extrapolated from the study conducted by Bousonville et al. (2022), which seem to hover at around 66 percent. To approximate this value and to account for increasing rolling resistance (which also rises with mass) we use a load-impact factor of 60 percent for any vehicle with two axles with ground contact and 70 percent for a truck that has three axles on the ground. To approximate the discontinuity in a fuel cost function due to a lift axle, we use the following data: In Germany, vehicles with three axles are allowed to have a gross weight of up to 25 tons (see §34 StVZO). A standard European truck with a lift axle, such as the Scania R590 6x2, exhibits a curb weight of around 10 tons (Scania). Payload is therefore capped at 15 tons, equivalent to our choice of Q^{lift} . For a vehicle with two axles, German traffic laws define a gross weight of 20 tons (see §34 StVZO). Thus, the lift axles must be lowered at a payload of 10 tons or 67 percent of the full capacity Q^{lift} . The resulting cost function

$$c_{ij}^{\text{lift}}(q) = \begin{cases} \underline{c}_{ij} \cdot (1.04 + q/Q^{\text{lift}} \cdot 0.6), & \text{if } q < 0.67 \cdot Q^{\text{lift}} \\ \underline{c}_{ij} \cdot (1.06 + q/Q^{\text{lift}} \cdot 0.7), & \text{if } q \geq 0.67 \cdot Q^{\text{lift}} \end{cases}$$

is shown by the blue (—) piecewise-linear function in Figure 5.

The second fleet is heterogeneous (denoted by “fleet_{het}”) and comprises two types of vehicles which differ in the number of axles (2 or 3; not liftable) and therefore in the load they are allowed to carry. A standard two-axle truck, e.g., the Scania R580 4x2, exhibits a curb weight of around 9 tons, leaving 11 tons for payload, i.e., $Q^{\text{het},1} = 0.73 \cdot Q^{\text{lift}}$ (Scania). As the two lifting devices needed for one lift axle weigh about 26 kg in total (Cargobull), we considered it negligible and exclude it from the analysis. Here, vehicles with a larger capacity are only needed when the capacity of a smaller vehicle is exceeded. Hence, we assume standard three-axle trucks to also weigh approximately 10 tons, again leading to the payload limit $Q^{\text{het},1} = Q^{\text{lift}}$. The difference in cost at curb weight has two reasons: (1) to approximate the one-ton difference in curb weight and (2) to add an average fuel cost increase of 2 percent in rolling resistance due to the additional axis (FleetOwner). The two resulting cost functions

$$\begin{aligned} c_{ij}^{\text{het},1}(q) &= \underline{c}_{ij} \cdot (1 + q/Q^{\text{lift}} \cdot 0.6), & q \leq Q^{\text{het},1} = 0.73 \cdot Q^{\text{lift}} \\ c_{ij}^{\text{het},2}(q) &= \underline{c}_{ij} \cdot (1.06 + q/Q^{\text{lift}} \cdot 0.7), & q \leq Q^{\text{het},2} = 1.0 \cdot Q^{\text{lift}} \end{aligned}$$

of the two vehicle types are shown in orange (—) and red (—) in Figure 5.

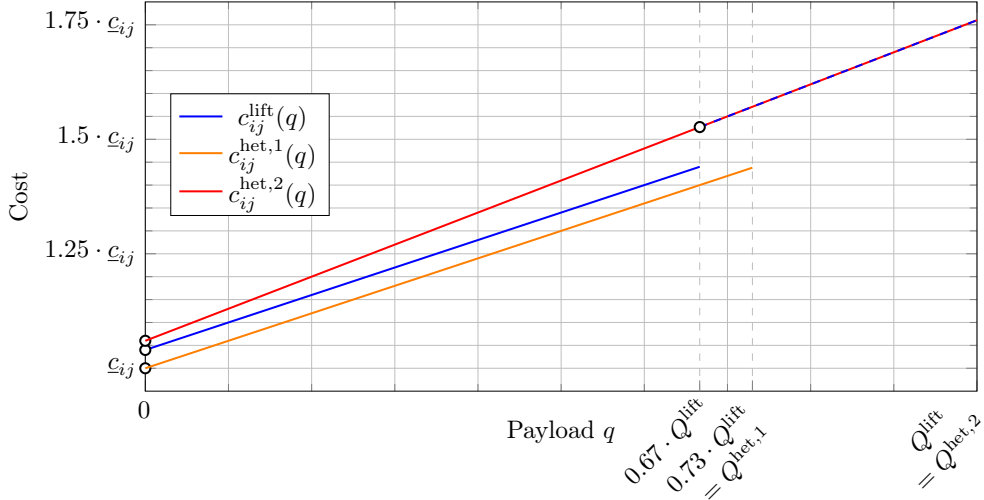


Figure 5: Load-dependent cost functions for comparing fleets with and without lift axes.

We grant a maximum computation time of 7200 seconds to each of the 56 instances, solved with both $\text{fleet}_{\text{lift}}$ and $\text{fleet}_{\text{het}}$. With the assumption that both fleets are unrestricted in the number of vehicles, 43 instances with configuration $\text{fleet}_{\text{lift}}$ and 37 instances with configuration $\text{fleet}_{\text{het}}$ are solved to optimality, respectively, with an overlap of all 37 instances (see [Appendix A](#) for details). To make results fully comparable, we report only on the subset of these 37 instances.

Metric	Fleet	C1 ($n = 5$)	R1 ($n = 12$)	RC1 ($n = 8$)	C2 ($n = 5$)	R2 ($n = 3$)	RC2 ($n = 4$)	Total ($n = 37$)
Fleet size	$\text{fleet}_{\text{het}}$	44.2	35.3	39.4	13.2	11.3	11.3	29.8
	Type 1 [%]	2.3	15.2	11.0	26.5	100.0	88.8	28.9
	Type 2 [%]	97.7	84.8	89.0	73.5	0.0	11.2	71.1
	$\text{fleet}_{\text{lift}}$	44.2	34.4	38.9	12.6	11.3	11.0	29.4
	Savings [%]	0.0	2.3	1.2	4.4	0.0	2.3	1.9
Total cost	$\text{fleet}_{\text{het}}$	46166.2	34386.3	50065.4	17814.3	14546.2	17487.1	33693.2
	$\text{fleet}_{\text{lift}}$	45470.8	33974.0	49469.9	17783.3	15014.9	17953.7	33421.0
	Savings [%]	1.5	1.2	1.2	0.2	-3.2	-2.7	0.3
Total distance	$\text{fleet}_{\text{het}}$	29977.3	22890.2	32568.3	11773.5	11353.7	12916.4	22424.6
	$\text{fleet}_{\text{lift}}$	29458.6	22388.8	31952.4	11416.1	11819.6	13103.5	22068.4
	Savings [%]	1.7	2.2	1.9	3.0	-4.1	-1.4	1.3

Table 5: Comparison of fleets $\text{fleet}_{\text{het}}$ and $\text{fleet}_{\text{lift}}$.

Table 5 presents averages of the number of vehicles used, the total cost, and the total distance. We can see a consistent reduction in the number of vehicles when the fleet with liftable axles is used. The fleet with liftable axles also outperforms the heterogeneous fleet in most of the instance classes in both total cost and distance. The reduction in cost and distance is here mainly due to the high utilization of the vehicles with three axles (with or without liftable axle) between 2/3 and full capacity in both fleets. For these vehicles with three axles, the lower cost for a load between 0 and 67 percent of the full capacity is decisive. Solely the classes R2 and RC2 lead to higher cost for the $\text{fleet}_{\text{lift}}$. In these instances, we see a larger share or even the exclusive use of vehicles with two axles (see “Type 1 [%]” and “Type 2 [%]”).

In summary, a fleet with liftable axles can reduce operational costs while minimizing environmental impact by decreasing travel distances and fleet size. A truck with a liftable axle costs slightly more in procurement and maintenance than a vehicle with only stationary axles (395€ per device; two needed per axle ([Cargobull](#))). However, this comparatively small investment is offset by reductions in fuel costs and tire wear. The most important aspect is that fleet sizing decisions become simpler when only considering one type of vehicle (with a liftable axle). While these results demonstrate the generally positive impact of lift axles on cost, it is important to note that they depend heavily on the specific cost parameters and the customer data. All results should be interpreted with caution, because actual cost structures may differ significantly in real-world applications.

7. Conclusions

Empirical studies show that transportation costs rise almost linearly with load due to increased fuel consumption. Non-linear effects become relevant when toll-by-weight schemes and weight-restricted passage are considered. In this paper, we introduced the VRPSDPTW-LDC, which generalizes the classical VRP by considering load-dependent costs. We demonstrate that arc-specific continuous or discontinuous non-decreasing load-dependent cost functions can capture the above and other aspects. Furthermore, we computed the routing costs of forward and backward partial paths in such networks from the arc-specific cost functions using only three types of operations: left shift of a function, addition of two functions, and addition of a function and a constant. This provides the theoretical foundation for computing costs in many types of heuristic and exact vehicle routing algorithms.

We described a VRPSDPTW-LDC-tailored BPC algorithm that handles the load-dependent cost functions in pricing problems involving the construction of partial paths using dynamic-programming labeling algorithms. In this case, the cost function of a label depends on a single variable: the quantity to be delivered to (or collected from) customers in the extension of the forward (or backward) path to which the label refers. These cost functions allow us to implement effective algorithmic techniques used by modern BPC algorithms for VRPs, such as bidirectional labeling with dynamic halfway points, robust and non-robust cuts, and the *ng*-path relaxation.

The first computational study showed that extending a BPC algorithm to handle arc-specific cost functions moderately increases computation time. Additionally, a higher load-impact factor makes instances easier to solve, as evidenced by the increasing number of optimal solutions and the decreasing computation time. Lastly, the impact of the number and location of the discontinuities in the cost function can be significant. We observed labels with cost functions that had more than 50 breakpoints, whereas the arc-specific routing cost functions had only three. Breakpoints located farther to the right of the cost functions' domains can lead to substantially higher computation times.

This paper also provides managerial insights into the structure of optimal solutions when additional fleet restrictions are imposed, such as a more binding vehicle capacity and a limited number of available vehicles. As the load-impact factor ρ increases, so do the optimal fleet (if not limited), the traveled distance, and the total cost (the latter increases strongly, while the former two increase slowly). Interestingly, the overall increase in total cost is more pronounced when additional fleet restrictions are imposed. Conversely, the average capacity utilization decrease slowly with ρ . Additional experiments with two fleets of vehicles indicate that trucks equipped with liftable axles are an opportunity to reduce transportation costs.

Heavily loaded trucks accelerate more slowly and may have a lower top speed. Therefore, travel times can also be load-dependent. Future work could focus on developing heuristic and exact solution approaches that simultaneously consider the interrelation of load, travel time, service time, cost, and emissions. Introducing speed decisions as additional independent variables per road segment would certainly make routing a much more challenging task.

Acknowledgement

The authors gratefully acknowledge the computing time granted on the high performance computing cluster MOGON NHR Süd-West at Johannes Gutenberg University Mainz (hpc.uni-mainz.de).

CRedit authorship contribution statement

Carolyn Hasse: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Stefan Irnich:** Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Writing – review & editing, Visualization, Supervision, Project administration.

References

- R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, 59(5):1269–1283, 2011. doi:[10.1287/opre.1110.0975](https://doi.org/10.1287/opre.1110.0975).
- T. Bektaş and G. Laporte. The pollution-routing problem. *Transportation Research Part B: Methodological*, 45(8):1232–1250, 2011. doi:[10.1016/j.trb.2011.02.004](https://doi.org/10.1016/j.trb.2011.02.004).
- H. Ben Ticha, N. Absi, D. Feillet, and A. Quilliot. Vehicle routing problems with road-network information: State of the art. *Networks*, 72(3):393–406, 2018. doi:[10.1002/net.21808](https://doi.org/10.1002/net.21808).
- D. J. Bertsimas. A vehicle routing problem with stochastic demand. *Operations Research*, 40(3):574–585, 1992. doi:[10.1287/opre.40.3.574](https://doi.org/10.1287/opre.40.3.574).
- N. Bianchessi, T. Gschwind, and S. Irnich. Resource-window reduction by reduced costs in path-based formulations for routing and scheduling problems. *INFORMS Journal on Computing*, 36(1):224–244, 2023. doi:[10.1287/ijoc.2022.0214](https://doi.org/10.1287/ijoc.2022.0214).
- T. Bousonville, D. C. Kamga, T. Krüger, and M. D. and. Data driven analysis and forecasting of medium and heavy truck fuel consumption. *Enterprise Information Systems*, 16(6):1856417, 2022. doi:[10.1080/17517575.2020.1856417](https://doi.org/10.1080/17517575.2020.1856417).
- Cargobull. Schmitz cargobull the trailer company. <https://www.cargobull-serviceportal.de/applications/serviceportal/default.aspx>, n.d. Accessed: 2025-22-10.
- L. Costa, C. Contardo, and G. Desaulniers. Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science*, 53(4):946–985, 2019. doi:[10.1287/trsc.2018.0878](https://doi.org/10.1287/trsc.2018.0878).
- G. Desaulniers, J. Desrosiers, I. Ioachim, M. M. Solomon, F. Soumis, and D. Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T. G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 57–93. Springer, 1998. ISBN 978-0-7923-8161-7. doi:[10.1007/978-1-4615-5755-5_3](https://doi.org/10.1007/978-1-4615-5755-5_3).
- G. Desaulniers, J. Desrosiers, and M. Solomon, editors. *Column Generation*. Springer, New York, NY, 2005. ISBN 0-387-25485-4.
- G. Desaulniers, F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized k -path inequalities for the vehicle routing problem with time windows. *Transportation Science*, 42(3):387–404, 2008. doi:[10.1287/trsc.1070.0223](https://doi.org/10.1287/trsc.1070.0223).
- G. Desaulniers, O. B. Madsen, and S. Ropke. The vehicle routing problem with time windows. In P. Toth and D. Vigo, editors, *Vehicle Routing*, chapter 5, pages 119–159. Society for Industrial & Applied Mathematics (SIAM), 2014. doi:[10.1137/1.9781611973594.ch5](https://doi.org/10.1137/1.9781611973594.ch5).
- J. Desrosiers, M. Lübbecke, G. Desaulniers, and J. B. Gauthier. *Branch-and-Price*. Springer, Heidelberg, 2025. ISBN 978-3031969164.
- M. Drexler. Rich vehicle routing in theory and practice. *Logistics Research*, 5(1-2):47–63, 2012. doi:[10.1007/s12159-012-0080-2](https://doi.org/10.1007/s12159-012-0080-2).
- M. Dror and P. Trudeau. Savings by split delivery routing. *Transportation Science*, 23(2):141–145, 1989. doi:[10.1287/trsc.23.2.141](https://doi.org/10.1287/trsc.23.2.141).
- Y. Dumas, J. Desrosiers, and F. Soumis. The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54(1):7–22, 1991. doi:[10.1016/0377-2217\(91\)90319-Q](https://doi.org/10.1016/0377-2217(91)90319-Q).
- A. Eydi and H. Alavi. Vehicle routing problem in reverse logistics with split demands of customers and fuel consumption optimization. *Arabian Journal for Science and Engineering*, 44(3):2641–2651, 2018. doi:[10.1007/s13369-018-3311-2](https://doi.org/10.1007/s13369-018-3311-2).
- D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004. doi:[10.1002/net.20033](https://doi.org/10.1002/net.20033).
- FleetOwner. Fleets explained: How to spec trucks. <https://www.fleetowner.com/fleets-explained/article/55283332/fleets-explained-how-to-spec-trucks>, n.d. Accessed: 2025-22-10.
- N. Frías, F. Johnson, and C. Valle. Hybrid algorithms for energy minimizing vehicle routing problem: Integrating clusterization and ant colony optimization. *IEEE Access*, 11:125800–125821, 2023. doi:[10.1109/ACCESS.2023.3325787](https://doi.org/10.1109/ACCESS.2023.3325787).
- R. Fukasawa, Q. He, and Y. Song. A branch-cut-and-price algorithm for the energy minimization vehicle routing problem. *Transportation Science*, 50(1):23–34, 2016. doi:[10.1287/trsc.2015.0593](https://doi.org/10.1287/trsc.2015.0593).
- A. K. Garside, R. Ahmad, and M. N. B. Muhtazaruddin. A recent review of solution approaches for green vehicle routing problem and its variants. *Operations Research Perspectives*, 12:100303, 2024. doi:[10.1016/j.orp.2024.100303](https://doi.org/10.1016/j.orp.2024.100303).
- B. L. Golden and R. T. Wong. Capacitated arc routing problems. *Networks*, 11(3):305–315, 1981. doi:[10.1002/net.3230110308](https://doi.org/10.1002/net.3230110308).
- I. Gribkovskaia and G. Laporte. One-to-many-to-one single vehicle pickup and delivery problems. In B. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces*, chapter 16, pages 359–377. Springer US, 2008. doi:[10.1007/978-0-387-77778-8_16](https://doi.org/10.1007/978-0-387-77778-8_16).
- T. Gschwind, S. Irnich, A.-K. Rothenbächer, and C. Tilk. Bidirectional labeling in column-generation algorithms for pickup-and-delivery problems. *European Journal of Operational Research*, 266(2):521–530, 2018. doi:[10.1016/j.ejor.2017.09.035](https://doi.org/10.1016/j.ejor.2017.09.035).
- K. Halse. *Modelling and Solving Complex Vehicle Routing Problems*. PhD dissertation No. 60, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark, Lyngby, Denmark, 1992.

- Q. He, S. Irnich, and Y. Song. Branch-and-cut-and-price for the vehicle routing problem with time windows and convex node costs. *Transportation Science*, 53(5):1409–1426, 2019. doi:[10.1287/trsc.2019.0891](https://doi.org/10.1287/trsc.2019.0891).
- S. Irnich. Resource extension functions: properties, inversion, and generalization to segments. *OR Spectrum*, 30(1):113–148, 2008. doi:[10.1007/s00291-007-0083-6](https://doi.org/10.1007/s00291-007-0083-6).
- S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, pages 33–65. Springer US, Boston, MA, 2005. ISBN 978-0-387-25486-9. doi:[10.1007/0-387-25486-2_2](https://doi.org/10.1007/0-387-25486-2_2).
- S. Irnich, P. Toth, and D. Vigo. The family of vehicle routing problems. In *Vehicle Routing*, chapter 1, pages 1–33. Society for Industrial & Applied Mathematics (SIAM), 2014. doi:[10.1137/1.9781611973594.ch1](https://doi.org/10.1137/1.9781611973594.ch1).
- İ. Kara, B. Y. Kara, and M. K. Yetis. Energy minimizing vehicle routing problem. In A. Dress, Y. Xu, and B. Zhu, editors, *Combinatorial Optimization and Applications*, pages 62–71, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-73556-4. doi:[10.1007/978-3-540-73556-4_9](https://doi.org/10.1007/978-3-540-73556-4_9).
- N. Kohl, J. Desrosiers, O. B. G. Madsen, M. M. Solomon, and F. Soumis. 2-Path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999. doi:[10.1287/trsc.33.1.101](https://doi.org/10.1287/trsc.33.1.101).
- R. Liu and Z. Jiang. A constraint relaxation-based algorithm for the load-dependent vehicle routing problem with time windows. *Flexible Services and Manufacturing Journal*, 31(2):331–353, 2018. doi:[10.1007/s10696-018-9323-0](https://doi.org/10.1007/s10696-018-9323-0).
- R. Liu, Z. Jiang, and B. Yuan. A heuristic algorithm for the load-dependent capacitated vehicle routing problem with time windows. In *2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 843–847. IEEE, 2015. doi:[10.1109/ieem.2015.7385767](https://doi.org/10.1109/ieem.2015.7385767).
- Z. Luo, H. Qin, D. Zhang, and A. Lim. Adaptive large neighborhood search heuristics for the vehicle routing problem with stochastic demands and weight-related cost. *Transportation Research Part E: Logistics and Transportation Review*, 85: 69–89, 2016. doi:[10.1016/j.tre.2015.11.004](https://doi.org/10.1016/j.tre.2015.11.004).
- Z. Luo, H. Qin, W. Zhu, and A. Lim. Branch and price and cut for the split-delivery vehicle routing problem with time windows and linear weight-related cost. *Transportation Science*, 51(2):668–687, 2017. doi:[10.1287/trsc.2015.0666](https://doi.org/10.1287/trsc.2015.0666).
- M. H. Mulati, R. Fukasawa, and F. K. Miyazawa. The arc-item-load and related formulations for the cumulative vehicle routing problem. *Discrete Optimization*, 45:100710, 2022. doi:[10.1016/j.disopt.2022.100710](https://doi.org/10.1016/j.disopt.2022.100710).
- NACFE. North American Council for Freight Efficiency: Trailer Lift Axles — [nacfe.org](https://nacfe.org/research/technology/trailer-general/trailer-lift-axles/). <https://nacfe.org/research/technology/trailer-general/trailer-lift-axles/>, 2024. [Accessed 13-05-2025].
- D. Pecin, C. Contardo, G. Desaulniers, and E. Uchoa. New enhancements for the exact solution of the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 29(3):489–502, 2017. doi:[10.1287/ijoc.2016.0744](https://doi.org/10.1287/ijoc.2016.0744).
- S. Rastani and B. Çatay. A large neighborhood search-based matheuristic for the load-dependent electric vehicle routing problem with time windows. *Annals of Operations Research*, 324(1-2):761–793, 2023. doi:[10.1007/s10479-021-04320-9](https://doi.org/10.1007/s10479-021-04320-9).
- G. Righini and M. Salani. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006. doi:[10.1016/j.disopt.2006.05.007](https://doi.org/10.1016/j.disopt.2006.05.007).
- S. Ropke and J.-F. Cordeau. Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science*, 43(3):267–286, 2009. doi:[10.1287/trsc.1090.0272](https://doi.org/10.1287/trsc.1090.0272).
- R. Sadykov, E. Uchoa, and A. Pessoa. A bucket graph-based labeling algorithm with application to vehicle routing. *Transportation Science*, 55(1):4–28, 2021. doi:[10.1287/trsc.2020.0985](https://doi.org/10.1287/trsc.2020.0985).
- Scania. Scania corporate website. <https://www.scania.com/>, n.d. Accessed: 2025-22-10.
- L. L. Schumaker. *Cambridge mathematical library: Spline functions: Basic theory*. Cambridge University Press, Cambridge, England, 3 edition, 2007.
- R. T. Seeley. *An introduction to Fourier series and integrals*. Dover Books on Mathematics. Dover Publications, Mineola, NY, 2006.
- C. Shen, H. Qin, and A. Lim. A capacitated vehicle routing problem with toll-by-weight rule. In B.-C. Chien and T.-P. Hong, editors, *Opportunities and Challenges for Next-Generation Applied Intelligence*, pages 311–316. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-92814-0. doi:[10.1007/978-3-540-92814-0_48](https://doi.org/10.1007/978-3-540-92814-0_48).
- M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987. doi:[10.1287/opre.35.2.254](https://doi.org/10.1287/opre.35.2.254).
- StVZO. Straßenverkehrs-Zulassungs-Ordnung (German road traffic licensing regulations), n.d.
- A. Subramanian, E. Uchoa, A. A. Pessoa, and L. S. Ochi. Branch-cut-and-price for the vehicle routing problem with simultaneous pickup and delivery. *Optimization Letters*, 7(7):1569–1581, 2012. doi:[10.1007/s11590-012-0570-9](https://doi.org/10.1007/s11590-012-0570-9).
- M.-D. Surcel and A. K. Bonsi. The impact of lift axles on fuel economy and GHG emissions reduction. *SAE International Journal of Commercial Vehicles*, 8(2):673–681, 2015. doi:[10.4271/2015-01-2874](https://doi.org/10.4271/2015-01-2874).
- C. Tilk, A.-K. Rothenbächer, T. Gschwind, and S. Irnich. Asymmetry matters: Dynamic half-way points in bidirectional labeling for solving shortest path problems with resource constraints faster. *European Journal of Operational Research*, 261(2):530–539, 2017. doi:[10.1016/j.ejor.2017.03.017](https://doi.org/10.1016/j.ejor.2017.03.017).
- P. Toth and D. Vigo, editors. *Vehicle Routing: Problems, Methods, and Applications*. Society for Industrial and Applied Mathematics, Philadelphia, 2014. ISBN 978-1-61197-358-7.
- N. T. Trang, P. Parthanadee, J. Buddhakulsomsiri, and P. D. Tai. Load-dependent vehicle routing problem with route time constraint: a heuristic approach. *Cogent Engineering*, 12(1):2533415, 2025. doi:[10.1080/23311916.2025.2533415](https://doi.org/10.1080/23311916.2025.2533415).
- H.-F. Wang and Y.-Y. Chen. A genetic algorithm for the simultaneous delivery and pickup problems with time window. *Computers & Industrial Engineering*, 62(1):84–95, 2012. doi:[10.1016/j.cie.2011.08.018](https://doi.org/10.1016/j.cie.2011.08.018).
- Y. Xiao, Q. Zhao, I. Kaku, and Y. Xu. Development of a fuel consumption optimization model for the capacitated vehicle routing problem. *Computers & Operations Research*, 39(7):1419–1431, 2012. doi:[10.1016/j.cor.2011.08.013](https://doi.org/10.1016/j.cor.2011.08.013).
- E. E. Zachariadis, C. D. Tarantilis, and C. T. Kiranoudis. The load-dependent vehicle routing problem and its pick-up and

- delivery extension. *Transportation Research Part B: Methodological*, 71:158–181, 2015. doi:[10.1016/j.trb.2014.11.004](https://doi.org/10.1016/j.trb.2014.11.004).
- J. Zhang, J. Tang, and R. Y. K. Fung. A scatter search for multi-depot vehicle routing problem with weight-related cost. *Asia-Pacific Journal of Operational Research*, 28(03):323–348, 2011. doi:[10.1142/S0217595911003260](https://doi.org/10.1142/S0217595911003260).
- Z. Zhang, H. Qin, A. Lim, and S. Guo. Branch and bound algorithm for a single vehicle routing problem with toll-by-weight scheme. In N. García-Pedrajas, F. Herrera, C. Fyfe, J. M. Benítez, and M. Ali, editors, *Trends in Applied Intelligent Systems*, pages 179–188, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. doi:[10.1007/978-3-642-13033-5_19](https://doi.org/10.1007/978-3-642-13033-5_19).
- Z. Zhang, H. Qin, W. Zhu, and A. Lim. The single vehicle routing problem with toll-by-weight scheme: A branch-and-bound approach. *European Journal of Operational Research*, 220(2):295–304, 2012. doi:[10.1016/j.ejor.2012.01.035](https://doi.org/10.1016/j.ejor.2012.01.035).

Appendix A. Instances for Managerial Insights on Lifiable Axle

The following tables show whether the Solomon benchmark instances were solved to optimality by the two fleet configurations $\text{fleet}_{\text{het}}$ and $\text{fleet}_{\text{lif}}$. A checkmark indicates that the corresponding fleet configuration solved the instance to optimality within a time limit of 7200 seconds. Instances solved to optimality by both $\text{fleet}_{\text{het}}$ and $\text{fleet}_{\text{lif}}$ are highlighted in bold and are included in the analysis presented in Section 6.2.

	Cdp101	Cdp102	Cdp103	Cdp104	Cdp105	Cdp106	Cdp107	Cdp108	Cdp109	Cdp201	Cdp202	Cdp203	Cdp204	Cdp205	Cdp206	Cdp207	Cdp208
Het				✓	✓	✓	✓	✓		✓		✓		✓	✓	✓	
Lift	✓		✓	✓	✓	✓	✓	✓		✓	✓	✓		✓	✓	✓	✓

Table A.6: Solution status of Solomon C instances with $\text{fleet}_{\text{het}}$ and $\text{fleet}_{\text{lif}}$

	RCdp101	RCdp102	RCdp103	RCdp104	RCdp105	RCdp106	RCdp107	RCdp108	RCdp201	RCdp202	RCdp203	RCdp204	RCdp205	RCdp206	RCdp207	RCdp208
Het	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓		
Lift	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓		

Table A.7: Solution status of Solomon RC instances with $\text{fleet}_{\text{het}}$ and $\text{fleet}_{\text{lif}}$

	Rdp101	Rdp102	Rdp103	Rdp104	Rdp105	Rdp106	Rdp107	Rdp108	Rdp109	Rdp110	Rdp111	Rdp112	Rdp201	Rdp202	Rdp203	Rdp204	Rdp205	Rdp206	Rdp207	Rdp208	Rdp209	Rdp210	Rdp211
Het	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓						
Lift	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓				✓		

Table A.8: Solution status of Solomon R instances with $\text{fleet}_{\text{het}}$ and $\text{fleet}_{\text{lif}}$

Appendix B. Abbreviations & Notation

In this section of the appendix, we give a tabular overview of used abbreviations and notation.

Acronym	Meaning
ACO	ant colony optimization
ALNS	adaptive large neighborhood search
B&B	branch-and-bound
B&C	branch-and-cut
BPC	branch-price-and-cut
CARP	capacitated arc routing problem
CF	cost function
CVRP	capacitated VRP
EMVRP	energy minimizing vehicle routing problem
EVRP	VRP with electric fleet
lm-SRI	limited-memory subset-row inequality
LNS	large neighborhood search
LP	linear programming
LS	local search
MDVRP	multi-depot VRP
MIP	mixed integer program
ML	machine learning
MP	master program
NS	neighborhood search
PDP	pickup-and-delivery problem
PDPTW	PDP with time windows
PRP	pollution routing problem
REF	resource extension function
RMP	restricted master program
SA	simulated annealing
SDVRP	split-delivery vehicle routing problem
SDVRPTW	SDVRP with time windows
SPPRC	shortest path problem with resource constraints
SS	scatter search
TS	tabu search
TSP	traveling salesman problem
VRP	vehicle routing problem
VRPLDRT	VRP with load-dependent distance and route time limit
VRP-SD	VRP with stochastic demands
VRPTW	VRP with time windows
VRPSDP	VRP with simultaneous delivery and pickup
VRPSDPTW	VRPSDP and time windows
VRPSDPTW-LDC	VRPSDP, time windows, and load-dependent cost

Table B.9: Abbreviations

Symbol	Description
A	set of arcs
B_i	a backward label at vertex $i \in V$
B_i^{res}	a resource res in a backward label at vertex $i \in V$
b_{ir}	route-vertex coefficient, indicating how often route $r \in \Omega^k$ serves customer $i \in N$
$c_{ij}^{(k)}(q)$	cost function (for a vehicle of type k) when traversing arc $(i, j) \in A$ carrying load q
$c_r^{(k)}$	cost of a route $r \in \Omega^k$ (for a vehicle of type k)
$\tilde{c}_r^{(k)}$	reduced costs of route $r \in \Omega^k$ (for a vehicle of type k)
$\underline{c}_{ij}^{(k)}$	cost incurred when traveling empty (for a vehicle of type k)
D	directed graph
d_i	delivery demand of vertex $i \in V$
$\bar{d}$	unknown additional delivery demand in forward labeling
$\delta_s^{(k)}$	slope of the s th segment in a piecewise defined cost function (for a vehicle of type k)
e_i	beginning of service time window of vertex $i \in V$
F_i	a forward label at vertex $i \in V$
F_i^{res}	a resource res in a forward label at vertex $i \in V$
H	half-way point value
I_s	interval or domain of segment s in a piecewise linear cost function
K	set of vehicle types
k	a vehicle type $k \in K$
ℓ_i	end of service time window of vertex $i \in V$
λ_r^k	binary route variable of route $r \in \Omega^k$ for a vehicle of type k
m_s	base cost multiplier of the s th segment in a discontinuous cost function
μ_k	dual price of fleet-size constraint (4c) of vehicle type $k \in K$
N	set of customers
n^k	number of available vehicles of type $k \in K$
o, o'	copies of the depot, representing start and end of a route, respectively
Ω^k	set of feasible routes of vehicle type $k \in K$
P_i	partial path to vertex $i \in V$
p_i	pickup demand of vertex $i \in V$
π_i	dual price of set-partitioning constraint (4b) of customer $i \in N$
$\bar{p}$	unknown additional pick demand in backward labeling
$Q^{(k)}$	capacity of a vehicle (of type $k \in K$)
$Q_s^{(k)}$	s th breakpoint in a piecewise defined cost function (of type $k \in K$)
q or q_{ij}	payload, carried when traversing arc $(i, j) \in A$
r	a route $r \in \Omega^k$ (of a vehicle of type $k \in K$)
ρ	load-impact factor in a continuous cost function
S	number of segments in a piecewise linear cost function
s	segment index $s, 1 \leq s \leq S$, in a piecewise linear cost function
t_{ij}	travel time from i to j , including service time at customer $i \in N$
V	set of vertices
$w_0^{(k)}$	curb weight of a vehicle (of type $k \in K$)
$w_{drv}^{(k)}$	weight of driver of a vehicle (of type $k \in K$)
$w_{ij}^{(k)}$	operating weight of a vehicle (of type $k \in K$)
$w_{max}^{(k)}$	gross vehicle weight (of a vehicle of type $k \in K$)

Table B.10: Notation