Nested Branch-and-Price-and-Cut for Vehicle Routing Problems with Multiple Resource Interdependencies

Christian Tilk*,[a], Michael Drexl[b], Stefan Irnich[a]

[a] *Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, D-55128 Mainz, Germany.*
[b] *Faculty of Applied Natural Sciences and Industrial Engineering, Deggendorf Institute of Technology, D-94469 Deggendorf, Germany.*

Abstract

This paper considers vehicle routing problems (VRPs) with multiple resource interdependencies and addresses the development and computational evaluation of an exact branch-and-price-and-cut algorithm for their solution. An interdependency between two resources means that the two resource consumptions influence one another in such a way that a tradeoff exists between them. This impacts the feasibility and/or the cost of a solution.

The subproblem in branch-and-price-and-cut procedures for VRPs is very often a variant of the shortest-path problem with resource constraints (SPPRC). For the exact solution of many SPPRC variants, dynamic-programming based labeling algorithms are predominant. The tradeoffs in problems with multiple resource interdependencies, however, render the application of labeling algorithms unpromising. This is because complex data structures for managing the tradeoff curves are necessary and only weak dominance criteria are possible, so that the labeling algorithm becomes almost a pure enumeration. Therefore, we propose to solve also the SPPRC subproblem with branch-and-price-and-cut. This results in a two-level, nested branch-and-price-and-cut algorithm. We analyze different variants of the algorithm to enable the exchange of columns and also rows between the different levels.

To demonstrate the computational viability of our approach, we perform computational experiments on a prototypical VRP with time windows, minimal and maximal delivery quantities for each customer, a customer-dependent profit paid for each demand unit delivered, and temporal synchronization constraints between some pairs of customers. In this problem, tradeoffs exist between cost and load and between cost and time.

Key words: Routing, Vehicle routing, Synchronization, Nested decomposition, Branch-and-price-and-cut

1. Introduction

In this paper, we study how *vehicle routing problems* (VRPs, Golden *et al.*, 2008; Toth and Vigo, 2014; Lahyani *et al.*, 2015; Braekers *et al.*, 2016) with multiple resource interdependencies can be solved exactly by branch-and-price-and-cut. *Resources* are measurable quantities that are consumed when a vehicle moves through a network, for example, cost, time, and load. By *resource interdependency* we mean that the consumption of one resource influences the consumption of another one and vice versa, so that there exists a tradeoff between the two. Such tradeoffs impact the feasibility and/or the cost of a solution. We use the *vehicle routing problem with time windows, soft demand, and synchronized visits* (VRPTW-SD-SV) as a prototypical example for a VRP with multiple resource interdependencies. The VRPTW-SD-SV is an extension of the well-known *VRP with time windows* (VRPTW, Desaulniers *et al.*, 2014) and can be described as follows. In addition to the hard time windows of the VRPTW, the VRPTW-SD-SV specifies minimal and maximal delivery quantities for each customer and a customer-dependent profit that is paid for each demand unit delivered. Furthermore, temporal synchronization between some pairs of customers is required in the sense that a minimal offset between the beginning of service at such customer pairs is given. To our knowledge, this particular VRP variant has not yet been studied in the literature.

*Corresponding author.
Email address: tilk@uni-mainz.de (Christian Tilk)

The VRPTW-SD-SV shares characteristics with routing problems with profits (Feillet *et al.*, 2005; Jepsen *et al.*, 2014), though these consider a fixed profit for visiting a certain customer, independently of the amount delivered. It also bears some similarity with split delivery VRPs (Archetti and Speranza, 2008; Irnich *et al.*, 2014), where also decisions must be taken on how much load to deliver to which customer by which vehicle. Finally, the temporal synchronization aspect in the VRPTW-SD-SV is present in the context of routing and scheduling service technicians (Dohn *et al.*, 2009) or nurses in home care (Mankowska *et al.*, 2014) as well as in synchronized aircraft routing and scheduling (Ioachim *et al.*, 1999).

In this paper, however, we are not interested in the VRPTW-SD-SV from an application perspective. As mentioned, our main motivation for studying the problem is that it constitutes a basic example of a VRP with more than two interdependent resources. In the VRPTW-SD-SV, there are three interdependent resources, namely, cost, time, and load, with two tradeoffs. The first is cost versus load: the more is delivered at a customer, the lower the cost of the route is, but the less can be delivered to other customers on this route. Hence, without considering all customers on a route, it is impossible to specify the optimal amount to deliver to a customer. The second tradeoff is cost versus time: on the one hand, it is good to start service early at a customer in order to have more temporal flexibility for subsequent customers along the route. On the other hand, it may be good or even necessary to start service at a customer later than at the earliest possible point in time in order to be able to fulfill the temporal synchronization constraints with other customers. Consequently, without knowing the visiting time for vertices in other routes, it is impossible to determine an optimal or even a feasible visiting time for a given customer.

Such interdependencies have far-reaching consequences for heuristic as well as exact solution approaches. *Branch-and-price-and-cut* is the predominant exact approach for time-constrained VRPs (see Desaulniers *et al.*, 2005; Lübbecke and Desrosiers, 2005; Feillet, 2010). The pricing problem in a branch-and-price-and-cut algorithm for VRPs is very often a variant of the *shortest-path problem with resource constraints* (SPPRC). It is usually solved by means of a dynamic-programming based labeling algorithm (cf. Irnich and Desaulniers, 2005). Such an algorithm creates partial paths by moving forward from an origin to a destination vertex in a network. Herein, a label stores information on resource consumption of a partial path from the origin to the endpoint vertex. Labels are propagated along the network arcs by resource extension functions (Desaulniers *et al.*, 1998; Irnich, 2008).

In many VRP variants, the propagation of a label along an arc creates a unique state that represents the only undominated resource values for this partial path. This happens, for example, when the lowest cost is obtained and the farthest feasible extensions are possible with the lowest possible resource values. The role of dominance procedures is to eliminate any unnecessary labels. In case of a unique undominated state per label, the dominance procedures can rely on a pointwise comparison of the resource values making dominance highly effective. Indeed, the simplicity of resource updates and the existence of powerful dominance procedures are the two main reasons for the successful application of labeling algorithms for solving SPPRCs.

However, the presence of resource interdependencies leads to tradeoffs between resources. These, in turn, result in infinitely many undominated states per partial path, i.e., per label. In the VRPTW-SD-SV, a single state describes the amount of load that is delivered, the point in time when the service starts, and the associated cost (taking profits of delivered amounts into account). The difficulty is not only to design a compact data structure that implicitly describes all associated feasible and undominated states of a label. The more severe problem is that typically only very weak dominance relationships exist between these labels, so that labeling becomes almost a pure enumeration.

Some cases of two interdependent resources can still be handled in a labeling algorithm, albeit at a significant algorithmic and computational overhead. For vertex-specific linear tradeoffs, e.g., between cost and time, the approach of Ioachim *et al.* (1998) stores tradeoff curves representing infinitely many states as labels. Moreover, it applies a sophisticated dominance procedure that allows several labels to prove that a certain label is dominated. Due to its complexity, only a few authors have adopted this approach. We are aware of the following pertinent papers: Christiansen and Nygreen (1998) (for a ship routing and scheduling problem with inventory constraints), Tagmouti *et al.* (2007) (for arc routing problems with time-dependent service cost), Liberatore *et al.* (2011) (for a VRP with soft time windows), Spliet and Gabor (2014) (for a VRP with time windows that have to be assigned before customer demand is known), Visser (2015) (for a simultaneous vehicle and driver routing and scheduling problem with driving and break time restrictions), and Tilk *et al.* (2017b) (for a pickup-and-delivery problem with discrete demands and autonomous and non-autonomous vehicles). An implicit tradeoff between delivered load and reduced cost has been efficiently encoded in the labeling algorithm for the split-delivery VRPTW by Desaulniers (2010) and Archetti *et al.* (2011). Another tradeoff between load and time is considered for the exact solution of a variant of the truck-and-trailer routing problem with time windows, in which the transfer of load from truck to trailer consumes a

quantity-dependent amount of time (Rothenbächer *et al.*, 2017). Finally, when partial recharging of battery electric commercial vehicles is allowed in a VRPTW, there exists a tradeoff between time and the battery's energy level (Desaulniers *et al.*, 2016).

When there are more than two interdependent resources, i.e., when there is more than one tradeoff, matters become even more complicated, and we are unaware of any work on exact labeling algorithms for this case when resources may take continuous values. One option for solving SPPRCs with tradeoffs is to discretize the resources (Drexl, 2007; Dohn *et al.*, 2011; Fink *et al.*, 2016) and create networks in which each vertex corresponds to a certain resource state at a customer. This is equivalent to creating several, but finitely many new labels from a single label in the original network with one vertex per customer. The issue with discretization is that, one the one hand, too fine discretization may lead to intractably large networks, but on the other hand, too coarse discretization may provide insufficient solution quality. This dilemma can be mitigated by not explicitly creating the whole discretized network. Instead, it is in some cases possible to compute optimal paths using only a partial network (Fischer and Helmberg, 2014; Boland *et al.*, 2017). In any case, if an SPPRC shall be solved exactly and discretization is not an option, one possibility is to formulate it as a mixed-integer program and solve it with LP-based methods. We take this approach in the present paper.

Our contribution is the development and computational evaluation of a nested branch-and-price-and-cut algorithm for the VRPTW-SD-SV, i.e., we solve both the original problem and the pricing problem in the column-generation algorithm for the original problem by branch-and-price-and-cut. This is of considerable relevance, as SPPRCs with multiple (more than two) resource interdependencies that cannot be solved with traditional labeling routines occur as subproblems of several types of VRPs. Examples include VRPs with quantity-dependent (un)loading times, with load-dependent travel times, with load transfer possibilities or requirements between vehicles, or with autonomous and non-autonomous vehicles (e.g., trucks and trailers) that must coordinate their movements in space and time. Generally speaking, VRPs where, in addition to the customer covering constraints, further synchronization requirements between the vehicles are present, which may concern spatial, temporal, load, or other aspects, possess multiple resource interdependencies (see the survey by Drexl, 2012). As the VRPTW-SD-SV is an example of such a VRP, it constitutes a suitable object of study for our purposes and allows us to gain fundamental insights that can possibly be generalized to related VRP variants.

The remainder of the paper is structured as follows. In the next section, we review related work on nested decomposition. After that, in Section 3, we present the extended set-partitioning formulation that forms the basis of our solution approach. The latter is described in Section 4. Section 5 describes the computational results, and Section 6 concludes the paper.

2. Earlier Work on Nested Decomposition

As described by Caprara *et al.* (2016), the nested application of column generation or Dantzig-Wolfe decomposition for solving multi-stage linear programs dates back to the early 1970s. Nevertheless, the literature on this topic is still rather scarce. This section briefly reviews pertinent recent papers.

Vanderbeck (2001) and Song (2009) apply nested column-generation approaches to two-dimensional cutting-stock problems motivated by practical applications. In their respective first-level master programs, both authors use formulations with one variable per cutting pattern. The resulting subproblems are essentially two-stage multiple-knapsack problems with additional constraints. Vanderbeck (2001) solves the first-level pricing problem heuristically by determining lower and upper bounds on the minimal reduced cost. The second-level pricing problem is solved a priori by complete enumeration, which is possible due to some special problem features stemming from the practical application context. A rounding heuristic is used to obtain feasible solutions to the overall problem. Song (2009) applies a column-generation heuristic to solve the first-level pricing problem. If the heuristic fails to identify a negative reduced cost column, a compact formulation of the pricing problem is solved exactly. After computing the linear relaxation of the original problem, branch-and-bound is applied using the columns found so far to obtain an integer solution.

Cordeau *et al.* (2001) describe a Benders decomposition approach for a simultaneous aircraft routing and crew scheduling problem. They give a path variable formulation of the problem with one variable for each sequence of flight legs of an aircraft and one for each crew pairing. In the Benders decomposition, both the relaxed Benders master program and the primal subproblem (which involves only crew variables) are solved by column generation.

Elhedhli and Goffin (2005) study a two-echelon multi-commodity facility location (production plant and distribution center) problem with single-sourcing constraints at the distribution centers. To solve the problem, they apply nested Lagrangean relaxation. First, they relax the commodity flow conservation constraints at the distribution centers. This leads to two subproblems: a transportation-type problem and a capacitated facility location problem with additional single sourcing constraints. The sum of the objectives of the two subproblems yields a lower bound on the optimal solution value of the overall problem. The former subproblem is solved with a standard MIP solver. The latter, which is computationally much more difficult, is again Lagrangean relaxed. The dual of the Lagrangean dual is then solved with column generation, using an analytic center cutting-plane method for the relaxed master programs. To improve the quality of the lower bound for the original problem, integer solutions are computed for the subproblems, which, for the second subproblem, requires embedding the process just described into branch-and-bound.

Karabuk (2009) uses nested decomposition to solve dial-a-ride problems. The first-level master program is a set-partitioning formulation where each variable corresponds to a complete vehicle route. The subproblem for computing such routes, i.e., the second-level master program, is a network flow problem with side constraints. It links vertices corresponding to single transport requests by arcs corresponding to partial routes. At the beginning and at the end of each such partial route, the vehicle is empty. An arc from i to j in the subproblem network represents a partial route starting at the pickup location of request i, visiting the pickup and the delivery vertices of zero or more other requests and the delivery location of i, and ending at the pickup location of request j. The second-level pricing problem is to generate arcs, i.e., partial routes for the second-level master. This problem is solved by dynamic programming. Due to the special structure of the network underlying the second-level master, it is possible to generate all relevant arcs of the network with the second-level pricing problem, which is solved only once in each first-level iteration, before the first-level pricing problem is solved, using the dual prices from the first-level master.

Akça (2010) develops several exact solution methods for the location routing problem, among them a branch-and-price algorithm based on an extended formulation with one variable for each so-called pairing, i.e., for each feasible sequence of routes starting and ending at the same depot. A possible approach for solving the corresponding pricing problems, one per potential depot location, is to decompose them further into the problem of generating routes and the problem of combining routes to pairings. To generate the routes, the author solves a standard *elementary SPPRC (ESPPRC)* on a network with a source and a sink vertex corresponding to the respective depot and one vertex for each customer. After solving this problem, a set of undominated routes is available. From these, a source-sink network is generated with one vertex for each undominated route. On this network, pairings are computed by again solving an ESPPRC. This means that the second-level pricing problem, i.e., the generation of routes, is solved (i) only once in each first-level iteration, (ii) using only dual prices from the first-level master program, and (iii) before the first-level pricing problem, i.e., before the generation of pairings. The procedure can be interpreted as nested column generation, because the solution of the pricing problem is implicitly based on an extensive formulation for which only a subset of all columns/variables is determined, namely, the undominated routes.

Uygur (2011) presents a nested column-generation algorithm that is part of a heuristic for a service technician scheduling problem with stochastic demand. Nested column generation is used to solve a relaxation of the deterministic equivalent of a stochastic integer program for assigning working-week schedules of eight-hour shifts to technicians, given a sample of equally likely request scenarios and various constraints such as request time windows, technician qualifications, and work rules. In principle, the first-level master program would be to assign a weekly schedule consisting of one or more shifts to each technician. However, such a formulation would have too many constraints to be computationally tractable. Therefore, the problem is relaxed and decomposed by shift, so that the actual first-level master program has binary variables corresponding to assignments of technicians to shifts. The first-level pricing problems, of which there is one per technician, determine shifts as combinations of requests in different scenarios. The second-level pricing problems, one for each scenario, seek a suitable assignment of requests to shifts. The problems on all levels are solved with a MIP solver. The author applies several acceleration and stabilization techniques to speed up the column-generation process and reports that these are indispensable to achieve acceptable computation times.

Hennig *et al.* (2012) study a crude oil tanker routing and scheduling problem. Specified amounts of different types of crude oil must either be picked up at or delivered to ports by tanker ships within given hard time windows. Split pickups and split deliveries are allowed. The amount of load picked up or delivered during a visit of a ship at a port determines the duration of the visit. In addition, there are two types of capacity constraints (weight and volume) for ships as well as for network arcs between ports. These constraints induce interdependencies between the resources weight, volume, and cost. In combination with the load-dependent

port service times, this makes the use of a one-stage decomposition prohibitively difficult. Hence, Hennig *et al.* (2012) apply nested branch-and-price. The first-level master program determines a solution to the original problem as a convex combination of variables that simultaneously correspond to sailing routes for ships and to cargo patterns representing feasible quantities of load picked up at or delivered to the ports along the route. The authors exploit that it is sufficient to consider only so-called extreme cargo patterns, where at most one pickup and at most one delivery of each type of oil is served fractionally. The second-level master generates extreme cargo pattern variables. It takes into account the ship and arc capacity constraints as well as the load-dependent service times. The second-level pricing problem is then a standard elementary shortest-path problem with time windows. It is solved by dynamic programming.

Dohn and Mason (2013) study a staff rostering problem where a roster-line, i.e., the sequence of shifts and days off during the planning period, is composed of work-stretches. Work-stretches consist of an on-stretch followed by an off-stretch. On-stretches, in turn, are made up of shifts (periods of time during which an employee is at work), whereas off-stretches are periods during which an employee does not work. To solve the problem, the authors use a nested subproblem with three levels, but no explicit nested delayed column generation, i.e., no second-level master program. Also in this application, the hierarchy is reversed, i.e., the subproblems are solved in reverse hierarchical order: first, shifts are combined to on-stretches, then, on- and off-stretches are paired to work-stretches, and finally, roster-lines are constructed from work-stretches. Each of these subproblems is an SPPRC and solved only once in each first-level iteration.

Muter *et al.* (2014) study the multi-depot vehicle routing problem with interdepot routes, a generalization of the multi-depot VRP in which the vehicles are allowed to stop at any depot to replenish. They adapt the approach of Akça (2010) to their problem as follows. A solution consists of a set of so-called rotations, one per vehicle. A rotation is an ordered sequence of routes, each of which starts at a depot, visits an ordered sequence of customers, and ends at a depot that may be different from the start depot. The start depot of the first route in a rotation must be the same as the end depot of the last route. The duration of a rotation is limited, but no customer time windows are considered. Muter *et al.* (2014) solve the subproblems of generating rotations (one for each depot) using a multigraph whose vertices correspond to depots and whose arcs correspond to routes starting and ending at the depots represented by the incident vertices. The number of arcs in this multigraph is exponential in the number of customers; therefore, the subproblem is solved by column generation. To obtain an integer solution to the subproblem, it would in principle have to be solved by branch-and-price. In order to avoid this, the authors generate all undominated columns (i.e., multigraph arcs, routes) for each pair of depots, and compute a rotation on the thus limited multigraph, analogously to the approach of Akça (2010). The undominated routes are generated by solving an ESPPRC on the original graph, taking into account the dual prices from the RMP. On each multigraph created with these routes, an SPPRC is solved to look for negative reduced cost rotations, which amounts to the solution of the first-level pricing problem.

Caprara *et al.* (2016) apply nested branch-and-price with more than two levels to the *temporal knapsack problem* (TKP). The TKP is a knapsack problem where a time horizon is considered and each item consumes knapsack capacity only during an item-specific interval. The knapsack capacity must be maintained at any point in time. To achieve this, it is sufficient to enforce the capacity constraint at all start times of the item-specific intervals. The TKP has the property that selecting any subset of the constraints leads again to a TKP. In their solution approach, the authors perform a Dantzig-Wolfe decomposition (one binary variable for each lattice point of the convex hull of feasible solutions). To solve the linear relaxations at the nodes of the branch-and-price tree, they recursively split the constraint set of the problem into two equal-sized subsets. The results of computational experiments on benchmark instances showed that there exists a marked, instance-size specific optimal number of recursion levels that leads to minimal computation times. For the largest benchmark instance class, which has 1792 constraints, six levels turned out best. At the final recursion level, the resulting TKPs are solved with a standard MIP solver.

All of the above papers are significant contributions to the principle of nested decomposition. It is, however, debatable whether the approaches of Vanderbeck (2001), Cordeau *et al.* (2001), Karabuk (2009), Akça (2010), Muter *et al.* (2014), and Dohn and Mason (2013) should be called nested *delayed* column generation, as the second-level problems in these works are not solved with an iterative procedure that alternates between master and subproblem. Essentially, only Hennig *et al.* (2012) and Caprara *et al.* (2016) apply nested branch-and-price as described in the previous section, and, of these, only the latter authors solve their problem to proven optimality.

In most of the references cited in this section, the motivation for the use of nested decomposition is to exploit a quasi natural multi-layer problem structure stemming from the application context. For Hennig *et al.* (2012) and us, the reason for using nested branch-and-price-and-cut is the necessity to solve a highly

complicated subproblem for which an integrated, one-step solution approach is unpromising. Our subproblem results from the extended set-partitioning formulation presented in the next section.

3. An Extended Set-Partitioning Formulation

Similar to the VRPTW, the VRPTW-SD-SV can be formally defined on a directed graph $G = (V, A)$ with vertex set V and arc set A. The vertex set V consists of the customer set $N = \{1, \ldots, n\}$ and vertices 0 and $n+1$ that represent the depot at the beginning and the end of the planning horizon respectively. Each customer $i \in N$ is associated with a time window $[a_i, b_i]$, a non-negative service time s_i, a demand interval $[q_i^{min}, q_i^{max}]$, and a profit $e_i \geq 0$ per demand unit delivered. Moreover, a time window $[a_0, b_0] = [a_{n+1}, b_{n+1}]$ is associated with the depots to model the planning horizon. For convenience, we also define $s_0 = s_{n+1} = q_0^{min} = q_0^{max} = q_{n+1}^{min} = q_{n+1}^{max} = 0$. Each delivery at customer $i \in N$ must start within $[a_i, b_i]$, but a vehicle may arrive prior to a_i and then wait until a_i before starting the delivery. Any amount Q_i between q_i^{min} and q_i^{max} may be delivered, leading to a profit of $e_i Q_i$.

A fleet of K identical vehicles with a capacity C is available to serve the customers. The vehicles are initially located at the depot 0 and must return to the depot $n+1$. We assume a fixed cost f for each vehicle which is used for visiting customers.

Each arc $(i, j) \in A$ represents a feasible movement from i to j, and it is associated with a non-negative travel time t_{ij} and a non-negative routing cost c_{ij}. We assume that t_{ij} already includes the service time s_i. Thus, an arc $(i, j) \in A$ exists only for vertices $i, j \in V$, $i \neq j$, with $a_i + t_{ij} \leq b_j$ and $q_i^{min} + q_j^{min} \leq C$.

We use a subset $P \subset N \times N$ and values Δ_{ij} for each $(i, j) \in P$ to model the temporal synchronization constraints. The value Δ_{ij} means that the service at customer $i \in N$ must start at least Δ_{ij} time units before the service at customer $j \in N$.

In addition to the standard VRPTW preprocessing (time window reducing and arc elimination, cf. Desrosiers *et al.*, 1995), we perform another type of time window reduction exploiting the temporal synchronization. If $(i, j) \in P$ and $a_i + \Delta_{ij} > a_j$, we set $a_j = a_i + \Delta_{ij}$. Similarly, if $(i, j) \in P$ and $b_j - \Delta_{ij} < b_i$, we set $b_i = b_j - \Delta_{ij}$.

A route r is defined as an elementary 0-$(n+1)$-path in G together with a schedule and a distribution plan. Such a path $p = (i_0, i_1, \ldots, i_m, i_{m+1})$ satisfies $i_0 = 0$, $i_{m+1} = n+1$, and $i_j \in N$ are all different for $1 \leq j \leq m$. The associated schedule $T = (T_0, T_1, \ldots, T_m, T_{m+1})$ specifies the start time T_j of service at each vertex i_j of p. The associated distribution plan $Q = (Q_0, Q_1, \ldots, Q_m, Q_{m+1})$ specifies the load Q_j delivered to each vertex i_j in p. A route $r = (p, T, Q)$ is feasible if the schedule T complies with travel times and time windows, i.e., if

$$T_{j-1} + t_{i_{j-1}, i_j} \leq T_j \quad (\text{for all } 1 \leq j \leq m+1) \qquad \text{and} \qquad T_j \in [a_{i_j}, b_{i_j}] \quad (\text{for all } 0 \leq j \leq m+1),$$

and if the distribution plan Q respects the demand intervals of the visited customers as well as the vehicle capacity, i.e.,

$$Q_j \in [q_{i_j}^{min}, q_{i_j}^{max}] \quad (\text{for all } 0 \leq j \leq m+1) \qquad \text{and} \qquad \sum_{j=0}^{m+1} Q_j \leq C.$$

Let R be the set of all feasible routes $r = (p, T, Q)$.

The cost c_r of a route $r = (p, T, Q) \in R$ is defined as the sum of the fixed cost f for the vehicle and the travel cost of the traversed arcs minus the profits per delivered load unit according to the distribution plan:

$$c_r = f + \sum_{(i,j) \in A(p)} c_{ij} - \sum_{j=1}^{m} e_{i_j} Q_j$$

The goal of the VRPTW-SD-SV is to find a set of feasible routes visiting each customer exactly once and respecting the synchronization constraints in P while minimizing the overall cost. A route-based formulation of the VRPTW-SD-SV has non-negative route variables λ_r for each feasible route $r \in R$ that indicate how often route $r \in R$ is performed, and binary arc variables x_{ij} indicating how often the arc $(i, j) \in A$ is traversed by all vehicles. Moreover, we denote by a_{ir} the number of times route r visits customer $i \in N$, and by b_{ijr} the number of times route r traverses arc $(i, j) \in A$. Finally, T_{jr} is the time at which customer $j \in N$ is served in route r. Now, the model reads as follows:

$$\min \quad \sum_{r \in R} c_r \lambda_r \qquad \text{Duals:} \qquad (1a)$$

$$\text{s.t.} \quad \sum_{r \in R} a_{ir} \lambda_r = 1 \qquad i \in N \qquad [\pi_i] \tag{1b}$$

$$\sum_{r \in R} T_{jr} \lambda_r - \sum_{r \in R} T_{ir} \lambda_r \geq \Delta_{ij} \qquad (i,j) \in P \qquad [\sigma_{ij}] \tag{1c}$$

$$\sum_{r \in R} \lambda_r \leq K \qquad [\mu] \tag{1d}$$

$$\sum_{r \in R} b_{ijr} \lambda_r - x_{ij} = 0 \qquad (i,j) \in A \qquad [\eta_{ij}] \tag{1e}$$

$$\lambda_r \geq 0 \qquad r \in R \tag{1f}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A \tag{1g}$$

The objective (1a) minimizes the overall cost taking into account profits obtained by deliveries. Constraints (1b) state that each customer is visited exactly once, and constraints (1c) ensure the synchronization in time for all $(i,j) \in P$. The fleet is limited by constraint (1d), and the path and arc variables are coupled by constraints (1e). The variable domains are given in (1f) and (1g).

Note that there are no binary restrictions on the λ_r variables. The reason for this is that a solution may be fractional in terms of the λ_r variables, because several routes may use the same path with different schedules and/or different distribution plans. Note further that any convex combination of feasible schedules/distribution plans for a fixed path forms a feasible schedule/distribution plan again. Such a convex combination can impose binary arc variables, so that no branching is necessary in this case. Integrality constraints in extensive formulations are discussed in detail by Desaulniers *et al.* (1998) and Jans (2010).

4. Nested Column Generation

In the following, formulation (1) is denoted as the *first-level master program*; its linear relaxation where the set of all feasible routes is additionally replaced by a subset $\bar{R}$ is called *restricted master program* (RMP). For solving the linear relaxation of the first-level master program, a column-generation algorithm is employed to an initially constructed RMP (Desaulniers *et al.*, 2005). Given an RMP, let π_i for $i \in N$ denote the dual prices of the constraints (1b), σ_{ij} for $(i,j) \in P$ the dual prices of the synchronization constraints (1c), μ the dual price of the convexity constraint (1d), and η_{ij} for $(i,j) \in A$ the dual prices of the arc-coupling constraints associated with the current solution. Its column-generation pricing problem, in the following referred to as the *first-level subproblem*, asks for a feasible route $r = (p,T,Q)$ with negative reduced cost $\bar{c}_r$ given by $c_r - \mu - \sum_{i \in N} a_{ir}\pi_i - \sum_{(i,j) \in A} b_{ijr}\eta_{ij} + \sum_{(i,j) \in P}(\sigma_{ij}T_j - \sigma_{ij}T_i)$. The pricing problem is an ESPPRC in which a tradeoff between cost and time as well as a tradeoff between load and time exists. As mentioned in the introduction, a labeling algorithm for solving the subproblem of the VRPTW-SD-SV would have to handle two interdependent tradeoffs, resulting in a weak dominance criterion and making the algorithm almost a pure enumeration. Therefore, we also solve the first-level subproblem with branch-and-price. Its column-generation subproblem is denoted as the *second-level subproblem*.

There exist (at least) two fundamentally different possibilities for modeling the first-level subproblem. First, we present in Section 4.1 a compact and an extensive formulation in which the time variables are defined on vertices (denoted *vertex-time formulations*). Moreover, we show how the latter formulation can be used in a branch-and-price algorithm. Second, we present in Section 4.2 formulations in which the time variables are defined on arcs (denoted *arc-time formulations*). Additionally, Section 4.3 discusses cutting and branching strategies for the first-level master program as well as their implications for the first-level subproblem. Finally, Section 4.4 describes cooperation schemes between the levels.

4.1. Vertex-Time Formulation of First-Level Subproblem

Defining visiting times on vertices leads to a compact vertex-time formulation of the first-level subproblem. The formulation utilizes binary arc-flow variables x_{ij} for all $(i,j) \in A$, load variables Q_i for all customers $i \in N$ indicating the amount of load delivered to that customer, and time variables T_i for all $i \in V$ to indicate the start of service at that vertex. Setting $\pi_{n+1} := \mu - f$, the routing cost of arc $(i,j) \in A$ is defined as $\bar{c}_{ij} = c_{ij} - \pi_j - \eta_{ij}$, and the compact vertex-time formulation reads as follows:

$$\min \quad \sum_{(i,j) \in A} \bar{c}_{ij} x_{ij} + \sum_{(i,j) \in P} (\sigma_{ij}T_j - \sigma_{ij}T_i) - \sum_{i \in N} e_i Q_i \tag{2a}$$

$$\text{s.t.} \quad \sum_{(0,j)\in\delta^+(0)} x_{0j} = \sum_{(i,n+1)\in\delta^-(n+1)} x_{in+1} = 1 \tag{2b}$$

$$\sum_{(j,i)\in\delta^-(i)} x_{ji} - \sum_{(i,j)\in\delta^+(i)} x_{ij} = 0 \qquad i \in N \tag{2c}$$

$$\sum_{i\in N} Q_i \leq C \tag{2d}$$

$$q_i^{min} \sum_{(i,j)\in\delta^+(i)} x_{ij} \leq Q_i \leq q_i^{max} \sum_{(i,j)\in\delta^+(i)} x_{ij} \qquad i \in N \tag{2e}$$

$$T_i + t_{ij} \leq T_j + t^{max}(1 - x_{ij}) \qquad (i,j) \in A \tag{2f}$$

$$a_i \leq T_i \leq b_i \qquad i \in \{0, n+1\} \tag{2g}$$

$$a_i \sum_{(i,j)\in\delta^+(i)} x_{ij} \leq T_i \leq b_i \sum_{(i,j)\in\delta^+(i)} x_{ij} \qquad i \in N \tag{2h}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A \tag{2i}$$

The objective function (2a) minimizes the difference between the sum of arc and time cost (resulting from the synchronization constraints of the first-level master) and the profits of the deliveries. Flow conversation is guaranteed by constraints (2b) and (2c). Constraints (2d) and (2e) ensures that the overall load as well as the load at each customer is bounded and coupled with the flow variables. Subtours are eliminated by constraints (2f), and time-window feasibility results from constraints (2g) and (2h). The binary conditions on the x-variables are given by (2i).

4.1.1. Path-Based Reformulation

As solving formulation (2) directly with a MIP solver is only possible (in reasonable time) for small-sized instances with less than ten vertices (we performed pre-tests with this type of approach), we apply Dantzig-Wolfe reformulation using path variables. (Recall that a route is a path with a time schedule and a distribution plan.) Let Ω be the set of all elementary time-window and load feasible 0-$(n+1)$-paths. More precisely, time-window feasibility just ensures that at least one feasible schedule exists; the actual times are determined in the path formulation by the time variables T_i for $i \in V$. The situation is analogous for load: a path is load feasible if each customer $i \in N$ can receive at least the minimum demand q_i^{min}. The actual optimal amount to be delivered is then determined in the path formulation with the help of load variables Q_i for $i \in N$. Alternatively, the set of feasible paths Ω can be described as the projection of the route set R onto its first path-related component, i.e., $\Omega = \text{proj}_1(R) = \{p : (p,T,Q) \in R\}$.

Besides the already mentioned time variables T_i and load variables Q_i, the path formulation uses two types of binary variables: θ_p, for each path $p \in \Omega$, indicates whether or not path p is used in the solution, and z_i, for each vertex $i \in V$, specifies whether vertex i is visited in the chosen path. A path $p \in \Omega$ is characterized by the following three coefficients: (1) the number a_{ip} of times vertex i is visited, (2) the number b_{ijp} of times arc (i,j) is traversed, and (3) the routing cost $\bar{c}_p = f - \mu - \sum_{i\in N} a_{ir}\pi_i + \sum_{(i,j)\in A} b_{ijp}(c_{ij} - \eta_{ij})$. With these definitions, the path-based formulation of the first-level subproblem reads as follows:

$$\min \quad \sum_{p\in\Omega} \bar{c}_p\theta_p + \sum_{(i,j)\in P} (\sigma_{ij}T_j - \sigma_{ij}T_i) - \sum_{i\in N} e_iQ_i \qquad \text{Duals:} \tag{3a}$$

$$\text{s.t.} \quad \sum_{p\in\Omega} \theta_p = 1 \qquad [\nu] \tag{3b}$$

$$z_i - \sum_{p\in\Omega} a_{ip}\theta_p = 0 \qquad i \in N \qquad [\rho_i] \tag{3c}$$

$$\sum_{i\in N} Q_i \leq C \tag{3d}$$

$$q_i^{min} z_i \leq Q_i \leq q_i^{max} z_i \qquad i \in N \tag{3e}$$

$$T_i - T_j + t^{max} \sum_{p\in\Omega} b_{ijp}\theta_p \leq t^{max} - t_{ij} \qquad (i,j) \in A \qquad [\phi_{ij}] \tag{3f}$$

$$a_i \leq T_i \leq b_i \qquad i \in \{0, n+1\} \tag{3g}$$

$$a_i z_i \leq T_i \leq b_i z_i \qquad i \in N \tag{3h}$$

$$z_i \in \{0,1\} \qquad i \in N \tag{3i}$$
$$\theta_p \in \{0,1\} \qquad p \in \Omega \tag{3j}$$

The objective function (3a) has the same meaning as (2a). Constraint (3b) ensures that exactly one path is selected. The visiting variables are coupled with the path variables with the help of constraints (3c). Constraints (3d)–(3h) are (identical) reformulations of the constrains (2d)–(2h). The variable domains are given by (3i) and (3j).

4.1.2. Second-Level Branch-and-Price

Due to the huge number of variables in formulation (3), we also solve it with branch-and-price. In the following, formulation (3) is referred to as the first-level subproblem (see beginning of Section 4) and as the *second-level master program* (we use these terms interchangeably). The associated linear relaxation of formulation (3) in which the set of all feasible paths is replaced by a subset $\bar{\Omega}$ is denoted as the *second-level RMP*.

As for the first level, after solving the linear relaxation with column generation, branching may be required to ensure integer solutions. To this end, we apply a two-stage hierarchical branching scheme: first, we branch on the occurrence of individual vertices in the chosen path, i.e., on the binary z_i-variables. Second, we branch on individual arcs. Note that any route present in the current solution of the first-level RMP is a feasible solution to the second-level master and it has an objective value (reduced cost) of zero. Hence, we can initialize the second-level RMP with an initial upper bound of zero, so that branch-and-bound nodes with a lower bound above zero can be cut off.

Column Generation in the Second Level. The resulting second-level subproblem is an ESPPRC with only time- and load-related constraints. This ESPPRC can be solved with a standard bidirectional labeling algorithm on the graph $G = (V, A)$, similar to labeling algorithms used in the VRPTW pricing problem (see, e.g., Righini and Salani, 2006). The reasons are that, on the one hand, time feasibility depends only on travel times and time windows, i.e., the existence of any feasible schedule suffices. On the other hand, concerning the vehicle capacity, a feasible route must allow the delivery of at least q_i^{min} units to each visited customer $i \in N$. Hence, the q_i^{min} define the demands. Finally, there is no longer any type of tradeoff between resources to be taken into account in the second-level subproblem.

With ν the dual price of constraint (3b), ρ_i for $i \in N$ the dual prices of constraints (3c), and ϕ_{ij} for $(i,j) \in A$ the dual prices of constraints (3f), the second-level subproblem asks for a feasible path p with negative reduced cost $\hat{c}_p$ given by $\bar{c}_p - \sum_{i \in N} \rho_i a_{ip} - \sum_{(i,j) \in A} b_{ijp} t^{max} \phi_{ij} - \nu$. Hence, the (reduced) cost $\hat{c}_{ij}$ of an arc $(i,j) \in A$ can be defined as $\bar{c}_{ij} - \rho_i - t^{max}\phi_{ij}$ with $\rho_{n+1} = \nu$.

In the labeling algorithm for the second-level subproblem, we apply the following four acceleration techniques. First, we use the *ng*-path relaxation (Baldacci *et al.*, 2011) with a neighborhood size of ten. Note that the z-variables as well as the T-variables in the second-level master (3) ensure elementarity of the paths used in an integer solution. Second, a *dynamic half-way point* defined on the time resource is used to balance the time spent in forward and backward labeling (Tilk *et al.*, 2017a). Third, instead of a set of visited customers, we define a set of *unreachable customers* to strengthen the dominance as proposed by Feillet *et al.* (2004). Fourth, the labeling is tentatively solved by means of *limited discrepancy search* (LDS, see Feillet *et al.*, 2007) before using the exact labeling.

4.2. Arc-Time Formulation of First-Level Subproblem

Computational studies (see Section 5) have shown that formulations (2) and (3) can have a weak linear relaxation. This can be attributed to the Big-M coefficient t^{max} in constraints (2f) and (3f). Ideas to eliminate the Big-M were presented by van Eijl (1995) and later by Maffioli and Sciomachen (1997). They define arc-time variables T_{ij} indicating the service start time at vertex i when using arc $(i,j) \in A$. Note that there is no variable defining the arrival time at the destination vertex $n+1$, as no such variable is needed for the VRPTW-SD-SV when time windows are preprocessed as proposed in Section 3. In general, for other VRP variants, this issue can be resolved by defining an artificial destination vertex $n+2$ and an extra arc $(n+1, n+2)$ with the additional arc-time variable $T_{n+1,n+2}$ that models the arrival time at $n+1$.

Our arc-time formulation utilizes binary arc-flow variables x_{ij} for all $(i,j) \in A$ and load variables Q_i for all customers $i \in N$ indicating the amount of load delivered to that customer. Then, the compact arc-time formulation reads as follows:

$$\min \quad \sum_{(i,j) \in A} \bar{c}_{ij} x_{ij} + \sum_{(i,j) \in P} \left(\sum_{(j,k) \in \delta^+(j)} \sigma_{ij} T_{jk} - \sum_{(i,k) \in \delta^+(i)} \sigma_{ij} T_{ik} \right) - \sum_{i \in N} e_i Q_i \tag{4a}$$

$$\text{s.t.} \quad \text{(2b), (2c), (2d), (2e), and (2i)} \tag{4b}$$

$$\sum_{(h,i)\in\delta^-(i)} (T_{hi} + t_{hi}x_{hi}) \leq \sum_{(i,j)\in\delta^+(i)} T_{ij} \qquad i \in V \setminus \{n+1\} \tag{4c}$$

$$a_i x_{ij} \leq T_{ij} \leq b_i x_{ij} \qquad (i,j) \in A \tag{4d}$$

The objective function (4a) minimizes the reduced cost of the constructed route. In comparison to the vertex-time formulation, the subtour-elimination constraints (4c) and time-window constraints (4d) are now formulated with the help of the new arc-time variables.

4.2.1. Path-Based Reformulation

As for the vertex-time formulation (2), directly solving the compact arc-time formulation (4) with a MIP solver is not a viable approach, even if the Big-M coefficient t^{max} is eliminated now. Instead, we use the following Dantzig-Wolfe reformulation, again a path-based model, and solve it with branch-and-price. The path variables θ_p have the same semantics as in (3) and the load and arc-time variables are adopted from the compact model, so that the path formulation reads as follows:

$$\min \quad \sum_{p\in\Omega} \bar{c}_p \theta_p + \sum_{(i,j)\in P} \left(\sum_{(j,k)\in\delta^+(j)} \sigma_{ij} T_{jk} - \sum_{(i,k)\in\delta^+(i)} \sigma_{ij} T_{ik} \right) - \sum_{i\in N} e_i Q_i \qquad \text{Duals:} \tag{5a}$$

$$\text{s.t.} \quad \sum_{p\in\Omega} \theta_p = 1 \qquad [\nu] \tag{5b}$$

$$z_i - \sum_{p\in\Omega} a_{ip}\theta_p = 0 \qquad i \in N \qquad [\rho_i] \tag{5c}$$

$$\sum_{i\in N} Q_i \leq C \tag{5d}$$

$$q_i^{min} z_i \leq Q_i \leq q_i^{max} z_i \qquad i \in N \tag{5e}$$

$$\sum_{(h,i)\in\delta^-(i)} \left(T_{hi} + t_{hi} \sum_{p\in\Omega} b_{hip}\theta_p \right) - \sum_{(i,j)\in\delta^+(i)} T_{ij} \leq 0 \qquad i \in V \qquad [\chi_i] \tag{5f}$$

$$a_i \sum_{p\in\Omega} b_{ijp}\theta_p - T_{ij} \leq 0 \qquad (i,j) \in A \qquad [\psi_{ij}^L] \tag{5g}$$

$$b_i \sum_{p\in\Omega} b_{ijp}\theta_p - T_{ij} \geq 0 \qquad (i,j) \in A \qquad [\psi_{ij}^U] \tag{5h}$$

$$z_i \in \{0,1\} \qquad i \in N \tag{5i}$$

$$\theta_p \in \{0,1\} \qquad p \in \Omega \tag{5j}$$

Also here, the objective function (5a) minimizes the reduced cost of the constructed route. The constraints (5b) state that one path has to be selected. The other constraints are (identical) reformulations of (4b)–(4d): the visiting variables are coupled with the path variables with the help of constraints (5c), a feasible distribution plan is guaranteed by (5d)–(5e), and a feasible schedule is guaranteed by (5f)–(5h). The variable domains are given by (5i) and (5j).

4.2.2. Second-Level Branch-and-Price

We sketch the branch-and-price algorithm for the resolution of formulation (5) and compare it with the branch-and-price algorithm used to solve formulation (3).

First, the second-level subproblem is again an ESPPRC with only time windows and load constraints; no tradeoffs between resources remain.

Second, the only difference to the subproblem of Section 4.1.2 is the definition of the reduced cost: let ν be the dual price of constraint (5b), ρ_i for $i \in N$ be the dual prices of constraints (5c), χ_i for $i \in V$ be the dual prices of constraints (5f), and ψ_{ij}^L and ψ_{ij}^U for $(i,j) \in A$ be the dual prices of constraints (5g) and (5h) respectively. Then, the subproblem asks for a path $p \in \Omega$ with negative reduced cost $\hat{c}_p$ given by $\bar{c}_p - \sum_{i\in N} \rho_i a_{ip} - \sum_{(i,j)\in A} b_{ijp}(t_{ij}\chi_{ij} + a_i\psi_{ij}^L + b_i\psi_{ij}^U) - \nu$. Hence, we define the reduced cost $\hat{c}_{ij}$ of an arc $(i,j) \in A$ as $\bar{c}_{ij} - \rho_j - t_{ij}\chi_{ij} - (a_i\psi_{ij}^L + b_i\psi_{ij}^U)$ with $\rho_{n+1} = \nu$.

Third, for the solution of the second-level subproblem, we use the same bidirectional labeling algorithm as in Section 4.1.2.

Finally, we also apply the same branching strategy as in the vertex-time reformulation.

4.3. Branching and Cutting in the First Level

In this subsection, we comment on the branching strategy and the use of valid inequalities for the solution of the first-level master program (1) presented in Section 3.

Branching Strategy. We apply a two-stage hierarchical branching scheme to ensure integer solutions of (1). We first branch on the number of vehicles and then on individual arcs. These arcs are selected with a *strong branching* technique (cf. Achterberg, 2007). The idea of strong branching is to more carefully select branching variables in order to reduce the size of the branch-and-bound tree, which is often helpful to reduce computation times, in particular when bounds are weak so that trees are large. As candidates we choose the eight most fractional arcs in the current solution and perform a rough evaluation of each candidate by solving the RMPs of its two child nodes without generating further columns. A similar procedure was successfully applied for the capacitated VRP by Pecin *et al.* (2017). The resulting RMP objective values usually overestimate the true lower bounds. However, this evaluation is relatively fast and often beneficial compared to straightforward arc-selection strategies. The arc to branch on is then chosen according to the product rule (Achterberg, 2007, p. 62).

As branch-and-bound node-selection rule, we apply a best-bound-first strategy, because our primary goal is to improve the dual bound.

Cutting Strategy. To strengthen the linear relaxation of the first-level master, we use two classes of valid inequalities. The first class is 2-*path cuts*, which were introduced by Kohl *et al.* (1999). Let $W \subset N$ be a subset of customers that cannot be visited by one single vehicle due to capacity, time window or synchronization restrictions, and let $\delta^-(W)$ be the set of all arcs $(i,j) \in A$ with $i \in W$ and $j \notin W$. The corresponding 2-path inequality is given by $\sum_{r\in R} \sum_{(i,j)\in\delta^-(W)} b_{ijr}\lambda_r \geq 2$. We use the heuristic proposed by Kohl *et al.* (1999) to generate candidate sets W with up to 20 customers. Each candidate set is then tested if it has to be served by at least two vehicles, and if so, the corresponding inequality is added.

The second class is *subset-row inequalities* (Jepsen *et al.*, 2008) defined on sets $U \subset N$ of cardinality three. The corresponding inequality is given by $\sum_{r\in R} \lfloor \frac{h^r}{2} \rfloor \lambda_r \leq 1$, where h^r is the number of times route r visits a customer in U. We separate violated subset-row inequalities by enumeration of sets U.

We add all violated 2-path cuts throughout the branch-and-bound process. In contrast, we separate subset-row inequalities only at the root node and add at most ten of them in total.

Impact of Branching and Cutting Decisions. We only comment on the impact of branching and cutting decisions on the path-based reformulations (3) and (5), as these are used to solve the first-level subproblems, not the compact models (2) and (4).

Branching on the number of vehicles does obviously not alter the structure of the second-level master (only the dual price μ on the fleet size is impacted). Branching on arcs can be realized in the second-level subproblem by eliminating arcs from the underlying network.

Concerning the cutting decisions, the dual prices of 2-path cuts and subset-row inequalities need to be incorporated in the cost of a second-level path $\bar{c}_p$. In both cases, this can also be handled in the second-level subproblem. The dual prices of the 2-path cuts have to be subtracted from the arc cost $\bar{c}_{ij}$ for all arcs (i,j) contained in the cut set $\delta^-(W)$ for the respective cut-defining set W. For each subset-row inequality defined by a set $U \subset N$ with $|U| = 3$, we need an additional resource in the second-level subproblem to count the number of times a vertex of U has been visited. Then, the dual price of the corresponding subset-row inequality has to be subtracted from the cost of the path for every second visit. Dominance can be handled as in the VRPTW subproblem (see Jepsen *et al.*, 2008).

4.4. Cooperation Schemes

Partial pricing (Gamache *et al.*, 1999) means to tentatively solve pricing problems with heuristic techniques as long as they find one or several columns with a sufficiently negative reduced cost. The idea is to quickly generate negative reduced cost columns and avoid the generally very time-consuming exact solution of the pricing problem as often as possible.

Also for the second-level master, partial pricing is generally beneficial. It is not necessary to perform a full branch-and-price in every iteration. In this section, we show how cooperation between the three levels

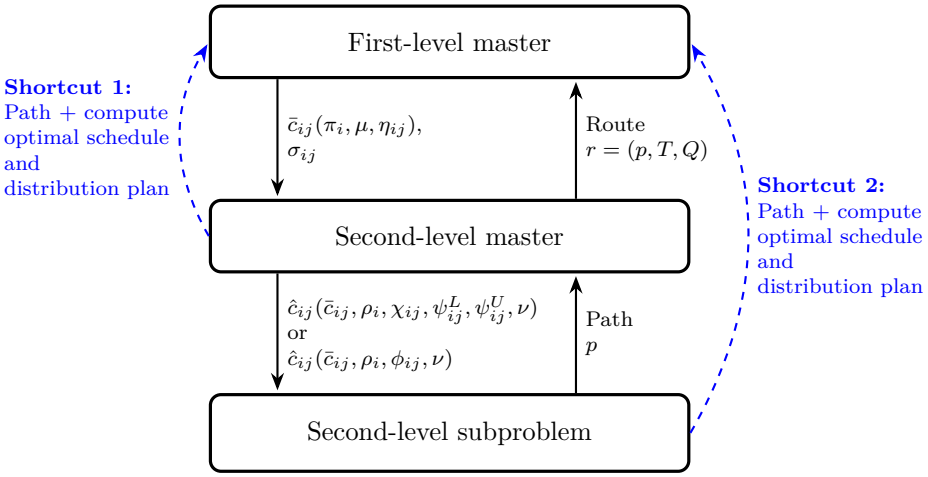

Figure 1: Cooperation scheme of the nested column-generation algorithm

of the nested column-generation approach can be exploited. There exist ample opportunities to accelerate the process. We discuss four of them.

First, note that the second-level master program does not need to be solved from scratch in each iteration of the first-level branch-and-price. We can keep the paths computed in previous iterations and only update their (reduced) cost according to the current dual prices given by the first-level master program. The updated path variables from previous iterations are then used to warm-start the second-level RMP.

Second, any negative reduced cost column suffices to stop the solution of the subproblem. We can therefore stop the second-level branch-and-price as soon as an integer solution in the second-level branch-and-bound tree with negative reduced cost is found. We pass this route $r = (p, T, Q)$ directly to the first-level master program and re-optimize the latter.

Additional possibilities for partial pricing exist as visualized in Figure 1, where the three levels of the nested column-generation algorithm are depicted. The third option is shown as *Shortcut 1* in the figure. For paths p that have been computed in previous iterations of the first-level master, there may exist a schedule T and a distribution plan Q so that the resulting route $r = (p, T, Q)$ has negative reduced cost regarding the current dual prices of the first-level master. The problem to determine an optimal schedule T and distribution plan Q will be discussed in Section 4.5.

The fourth option, shown as *Shortcut 2* in Figure 1, passes one or several paths p that are computed in the second-level subproblem directly to the first-level master. This is possible if a schedule T and a distribution plan Q for p exist so that the resulting route $r = (p, T, Q)$ has negative reduced cost $\bar{c}_r$ regarding the current dual prices of the first-level master $(\pi_i, \mu, \eta_{ij}, \sigma_{ij})$.

Transferring Bounds between Levels. Cooperation is not restricted to the generation of columns, also bounds can be transferred. Every lower bound computed for the second-level master can be used to compute a *Lagrangean lower bound* (LLB, cf. Lübbecke and Desrosiers, 2005) in the first-level master. Such a bound can be used to sooner terminate a branch-and-bound node of the first-level master. More precisely, let $RMP_{Level\,1}$ be the objective value of the current iteration of the first-level master program and $LB_{Level\,2}$ a lower bound on the current second-level master program. Then the LLB is

$$LLB = RMP_{Level\,1} + K \cdot LB_{Level\,2}.$$

If a solution to the VRPTW-SD-SV with value $UB_{Level\,1}$ is known and $LLB \geq UB_{Level\,1}$, the current branch-and-bound node of the first-level master cannot lead to an improving solution and may therefore be discarded. In particular for infeasible branches, e.g., when the first-level master has branched on the smaller number of vehicles, we observed that a quickly computed root or Lagrangean lower bound in the second-level master often leads to such an early termination.

ng-Path Relaxation. We now discuss the special case of using an *ng*-path relaxation in the second-level subproblem. Recall that in the second-level master, only elementary paths can be feasible integer solutions. Any non-elementary path is excluded due to constraints (3b), (3c), (3i) in the vertex-time formulation and

constraints (5b), (5c), (5i) in the arc-time formulation respectively. In particular, only elementary paths are passed to the first-level master on the regular way (solid arrow in Figure 1). However, using the shortcuts gives us the possibility to pass non-elementary paths to the first-level master.

Passing non-elementary paths is attractive because it can further accelerate the overall process. On the downside, the linear relaxation bound of the first-level master may deteriorate. There exists a clear tradeoff. If one strives for a best possible linear-relaxation bound of the first-level master, one must pass only elementary paths via the shortcuts. But if one allows non-elementary paths on the shortcuts, the first-level master may compute a linear-relaxation bound value that is between the value of the elementary and the *ng*-path relaxation. This computed bound can be better than the one of the *ng*-path relaxation, because the first-level subproblem is not able to solve the *ng*-path relaxation exactly (formulations (3) and (5) allow only elementary routes as exact solutions). Even worse, the value of the computed bound depends on the course of the branch-and-price(-and-cut) algorithm and is, in this sense, not unique. Nevertheless, the computed bound is certainly valid. This phenomenon was studied before by Desaulniers *et al.* (2016) and is known as *selective pricing*.

4.5. Optimal Schedule and Distribution Plan of a Fixed Path

For each path $p \in \Omega$, we can compute an optimal schedule T and an optimal distribution plan Q (optimal with regard to the current reduced cost, disregarding any synchronization constraints). For a given path $p \in \Omega$, this is the problem

$$\min_{r=(p,T,Q)\in\Omega} \bar{c}_r = \min_{r=(p,T,Q)\in\Omega} \sum_{(i,j)\in A(p)} c_{ij} - \mu - \sum_{i\in N} a_{ir}\pi_i - \sum_{(i,j)\in A} b_{ijr}\eta_{ij} + \sum_{(i,j)\in P} (\sigma_{ij}T_j - \sigma_{ij}T_i) - \sum_{i\in N} e_i Q_i.$$

Note that, for a single path, the computation of the schedule and the distribution plan can be done independently of one another.

To determine an optimal schedule, we use the forward label extension from the labeling algorithm for the ESPP with linear vertex cost presented by Tilk *et al.* (2017b), using a linear vertex cost of $\tilde{c}_i = \sum_{j\in N:(j,i)\in P} \sigma_{ji} - \sum_{j\in N:(i,j)\in P} \sigma_{ij}$. The computational effort is quadratic in the path length in the worst case. (Alternatively, the linear-time algorithm of Dumas *et al.* (1990) could be used, but the effect is minor as paths are relatively short).

We compute an optimal distribution plan Q as follows. The customers visited on p are ordered by decreasing profits, yielding a sequence $(j_1, \ldots, j_m)$. Then, for $i = 1, \ldots, m$, the delivered amount is computed iteratively as

$$Q_{j_i} := \min\left\{ \max\left\{ C - \sum_{l=1}^{i-1} Q_{j_l} - \sum_{l=i+1}^{m} q_{j_l}^{min}, q_{j_i}^{min} \right\}, q_{j_i}^{max} \right\}.$$

This procedure yields an extreme distribution plan in the sense that any path p contains at most one customer i who receives a delivery above the minimum quantity (q_i^{min}) and below the maximum (q_i^{max}). As mentioned at the end of Section 3, the route variables λ_r of the first-level master program are continuous, so they can be convex combined to yield all feasible combinations of schedules and distribution plans for a path. A similar idea was exploited in the labeling algorithm presented by Desaulniers (2010). Finally, note that in a more general context beyond VRPTW-SD-SV, when resources still depend on one another in a complex but linear way, the resource optimization problem for a fixed path can be solved as a linear program.

5. Computational Results

In this section, we report our computational results on the comparison of different nested branch-and-price-and-cut algorithms for the VRPTW-SD-SV. All results were obtained using a standard PC with an Intel(R) Core(TM) i7-6900k processor clocked at 3.2 GHz, 64 GB RAM, and Windows 10. The algorithms were implemented in C++ and compiled into 64-bit single-thread code with MS Visual Studio 2013. The callable library of CPLEX 12.6.2 was used for solving the RMPs in the first and second-level.

Because the VRPTW-SD-SV is studied in this paper for the first time, we created a new set of 120 benchmark instances. The set is divided into twelve instance classes with ten instances each that differ in the number of customers and temporal synchronization constraints as well as in the vehicle capacities and time window widths. More precisely, locations are placed in a 100×100 area, travel cost and travel time are based on rounded Euclidean distances, and the planning horizon is set to 1000 time units. We distinguish between small and large vehicle capacity (150 and 300) and time window width (25 to 125 and 50 to 250).

All parameters were drawn at random. Synchronization constraints were chosen such that they are not trivially fulfilled by the time windows of the involved vertices. The instance files can be downloaded from `http://logistik.bwl.uni-mainz.de/benchmarks.php`.

In a first test, we compared our algorithm when either the vertex-time or the arc-time formulation is used in the second-level branch-and-price with and without the use of shortcuts (SCs). In all cases, the second-level RMPs were always warm-started with the columns generated in previous iterations. We used a hard CPU time limit of 7200 seconds for this test. Table 1 summarizes the results. The table contains the number of instances solved to optimality (*#opt*) as well as the minimum, average, and maximum solution time in seconds (*Time*) for each instance class and both the vertex-time and the arc-time formulation, with and without SCs. The table entries show that, whether or not SCs were used, the vertex-time formulation clearly outperformed the arc-time formulation: over 60% more instances were solved to optimality, and the computation time was significantly lower. In addition, the improvement when using SCs was enormous: both the vertex-time and the arc-time formulation could solve around 60% more instances to optimality, and the average computation time decreased by more than 60% and 25% respectively.

Table 1: Comparison of both second-level master formulations with and without shortcuts

				Vertex-time formulation (3)				Arc-time formulation (5)			
	Instance class				Time [seconds]				Time [seconds]		
Setting	n	TWs, Cap	$\|P\|$	#opt	min	avg	max	#opt	min	avg	max
Without	20	small	5	10	9.7	37.9	161.9	10	99.8	277.3	554.5
Shortcuts	20	small	10	10	6.1	46.3	263.1	10	32.7	431.6	1996.4
	20	large	5	9	6.5	2074.4	7200.0	6	48.3	4220.3	7200.0
	20	large	10	8	9.4	2526.6	7200.0	5	152.5	4594.1	7200.0
	30	small	10	7	1286.8	3970.9	7200.0	0	7200.0	7200.0	7200.0
	30	small	20	9	297.0	1994.8	7200.0	1	6349.6	7115.0	7200.0
	30	large	10	2	2069.2	6446.6	7200.0	0	7200.0	7200.0	7200.0
	30	large	20	4	189.6	5566.9	7200.0	1	3350.3	6815.0	7200.0
	40	small	15	0	7200.0	7200.0	7200.0	0	7200.0	7200.0	7200.0
	40	small	30	4	80.4	5719.2	7200.0	2	330.1	5897.1	7200.0
	40	large	15	0	7200.0	7200.0	7200.0	0	7200.0	7200.0	7200.0
	40	large	30	1	867.1	6566.7	7200.0	0	7200.0	7200.0	7200.0
	Total			64		4112.5		35		5445.9	
With	20	small	5	10	2.1	7.2	17.9	10	5.1	47.0	83.6
Shortcuts	20	small	10	10	1.1	8.5	35.8	10	3.8	77.5	395.9
	20	large	5	10	0.6	62.0	362.3	10	1.4	979.3	4625.3
	20	large	10	10	3.6	414.4	3730.9	9	26.2	1745.4	7200.0
	30	small	10	10	15.2	316.4	1374.7	2	1038.8	6160.5	7200.0
	30	small	20	10	9.4	128.4	397.3	5	196.5	5017.1	7200.0
	30	large	10	6	52.2	3583.5	7200.0	1	2931.9	6773.2	7200.0
	30	large	20	8	4.3	1654.8	7200.0	7	73.3	3096.7	7200.0
	40	small	15	7	118.1	2911.2	7200.0	0	7200.0	7200.0	7200.0
	40	small	30	8	2.0	1693.4	7200.0	3	31.8	5578.2	7200.0
	40	large	15	5	30.0	4233.3	7200.0	1	5309.3	7010.9	7200.0
	40	large	30	8	6.1	2833.9	7200.0	5	92.3	4015.1	7200.0
	Total			102		1487.2		63		3975.1	

As can also be seen from the Table 1, out of the twelve instance classes, those with large vehicle capacities and wide time windows required the longest computation time. A larger number of customers also increases the solution time significantly, whereas a larger number of synchronization constraints leads to a decrease in the solution time. The latter can be attributed to the preprocessing, more synchronization constraints lead to tighter time windows and tighter time windows significantly decrease the computation time needed.

To better understand why the vertex-time formulation is superior, we ran a second test, investigating only the second-level branch-and-price. We ran our algorithm with SCs, and whenever the second-level master program was solved to optimality, i.e., the SCs did not find a negative reduced cost route, we solved this instance of the first-level pricing problem with both the vertex-time and the arc-time formulation. In this

way, both versions were solved with identical reduced cost, and the comparison was not impacted by an otherwise different trajectory of the dual prices. Moreover, pricing problems tend to become more difficult towards the end of the column-generation process; hence, the instances we used should be more of a stress test for the formulations. We used a hard CPU time limit of 7200 seconds for this test.

Table 2 reports results for the solution of the linear relaxation of the second-level master program. In the table, instances are grouped according to the number of customers. For each instance group, the table contains the minimum (*min*), geometric mean (*gm*), and maximum (*max*) of the ratio of the computation times (*Time ratio*), and the ratio of the linear relaxation bounds (*Bound ratio*). (The arithmetic mean makes no sense for expressing the average of factors, i.e., of non-normalized ratios of two values.) In addition, the column *#inst* displays the number of instances that were considered in the respective comparisons. For the computation of the time, we took into account only those 105 instances of the second-level master program for which the solution of the linear relaxation took more than 0.1 seconds in both formulations. In this way, the impact of inaccuracies in time measurement was reduced. For the linear relaxation ratio computation, we used all instances that were solved in the time limit with both formulations.

The table shows that the arc-time formulation is slightly stronger on average: there is an average gap of 2% between the values of the linear relaxations of the arc-time and the vertex-time formulation. (Note that the bound ratios in the table are positive, as the optimal values of the linear relaxations are always non-positive.) On the other hand, solving the linear relaxation of the vertex-time formulation is significantly faster: The table shows an average speed-up factor of 13 and the maximum factor is actually more than 185. Note that in very rare cases the linear relaxation of the arc-time formulation was solved faster (three instances in total), but the detailed results showed that computation times were less than 0.3 seconds for these instances. Moreover, the values of the linear relaxations were identical in both formulations for 373 out of 413 instances. This can again be attributed to the preprocessing; tighter time windows lead to less flexibility in the schedule of a route and this, in turn, often leads to the same linear relaxation values in both formulations. Indeed, when we ran this test without preprocessing, the linear relaxation bounds differed for 40% of the instances.

Table 2: Comparison of the solution of the linear relaxation of the second-level master

	Time ratio ($t_{LP}^{arc}/t_{LP}^{vertex}$)				Bound ratio (LB^{arc}/LB^{vertex})			
n	#inst	min	gm	max	#inst	min	gm	max
20	11	4.29	15.37	86.85	217	0.71	0.99	1.00
30	57	0.50	20.01	185.75	135	0.41	0.98	1.00
40	37	0.70	6.72	47.53	61	0.50	0.95	1.00
Total	105		13.25		413		0.98	

Table 3 reports the results of solving the second-level master to integer optimality with both formulations. For each instance class, the table contains the number of considered instances, the minimum, average, and maximum computation time in seconds as well as the minimum, average, and maximum number of branch-and-bound nodes needed. All in all, the results confirm the superiority of the vertex-time formulation, which on average required less than a ninth of the computation time compared to the arc-time formulation, although nearly 20% more branch-and-bound nodes were explored on average.

Table 3: Comparison of the integer solution of the second-level master

		Vertex-time formulation (3)						Arc-time formulation (5)					
		Time [seconds]			#B&B nodes			Time [seconds]			#B&B nodes		
n	#inst	min	avg	max	min	avg	max	min	avg	max	min	avg	max
20	217	0.1	6.5	205.4	1	93	531	0.1	140.1	4200.1	1	86	483
30	135	0.1	18.4	446.7	1	110	609	0.1	912.2	4337.3	1	104	249
40	61	0.1	285.6	4817.9	1	149	1101	0.1	622.0	5833.1	1	73	237
Total	413		51.6			107			463.7			90	

6. Conclusions

In this paper, we have considered vehicle routing problems with multiple resource interdependencies and have discussed the difficulties arising in branch-and-price algorithms when tradeoffs between the consumption of different resources exist. To overcome the issues caused by resource interdependencies, we have proposed a nested, two-level decomposition approach where both the original problem and the pricing problem for the original problem are solved by branch-and-price-and-cut. To demonstrate the approach on a prototypical example, we have examined the VRPTW-SD-SV, which extends the well-known VRPTW by specifying minimal and maximal delivery quantities for each customer and a customer-dependent profit that is paid for each demand unit delivered. The problem also requires temporal synchronization between some pairs of customers by indicating minimal offsets between the respective service start times.

We have experimented with two formulations for the second-level master program. These differ in the way the temporal aspect of the problem is handled: the first has time variables defined on the vertices of the underlying network, the second uses time variables defined on the arcs. In addition, we have considered the use of cooperation schemes to speed up the solution process by avoiding to solve the second-level master to optimality in each first-level iteration. Instead, we pass paths computed in the first- or second-level subproblem directly to the first-level master if there exist a compatible schedule and distribution plan such that the resulting route has negative reduced cost. To this end, we apply efficient procedures that allow us to quickly compute an optimal schedule and distribution plan for a given path with the current dual prices from the first-level master.

Our computational experiments, which are performed on a new set of benchmark instances, have shown that the approach using time variables on vertices is clearly superior: average computation times are much lower, and far more instances are solved to optimality within a given time limit. This is true even though the linear relaxation of the arc-time formulation is stronger and therefore has smaller branch-and-bound trees in the second-level master. However, this advantage is overcompensated by the shorter computation times needed to solve each branch-and-bound node of the vertex-time formulation. Experiments with the cooperation schemes show that these provide significant speedups for both formulations, almost doubling the number of instances solved to optimality.

In future work, we intend to adapt the presented decomposition approach to VRPs with more complicated resource interdependencies. We also plan to compare the approach with alternative methods, in particular, complete or partial discretization of resource consumptions. Discretization would allow the use of one-level branch-and-price with classical labeling algorithms for the solution of the subproblem, but requires ways to curtail the growth of the resulting subproblem networks.

Given the difficulty of their exact (as well as heuristic) solution on the one hand and their practical relevance on the other, VRPs with multiple resource interdependencies are well worthy of further study. Research in the field is growing, and we conjecture that the topic will remain on the research agenda of the vehicle routing community for some time.

Acknowledgement

This research was supported by the Deutsche Forschungsgemeinschaft (DFG) under grants IR 122/6-1 and IR 122/9-1, respectively. This support is gratefully acknowledged.

References

Achterberg, T. (2007). *Constraint Integer Programming.* Ph.D. thesis, Faculty II – Mathematics and Natural Sciences, Technische Universität Berlin, Berlin, Germany.

Akça, Z. (2010). *Integrated Location, Routing and Scheduling Problems: Models and Algorithms.* Ph.D. thesis, Lehigh University, Bethlehem, USA.

Archetti, C. and Speranza, M. (2008). The split delivery vehicle routing problem: A survey. In B. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 103–122. Springer, New York.

Archetti, C., Bouchard, M., and Desaulniers, G. (2011). Enhanced branch and price and cut for vehicle routing with split deliveries and time windows. *Transportation Science*, **45**(3), 285–298.

Baldacci, R., Mingozzi, A., and Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, **59**(5), 1269–1283.

Boland, N., Hewitt, M., Vu, D., and Savelsbergh, M. (2017). Solving the traveling salesman problem with time windows through dynamically generated time-expanded networks. In D. Salvagnin and M. Lombardi, editors, *Integration of AI and OR Techniques in Constraint Programming. CPAIOR 2017.*, volume 10335 of *Lecture Notes in Computer Science*, pages 254–262. Springer, Cham.

Braekers, K., Ramaekers, K., and Van Nieuwenhuyse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, **99**, 300–313.

Caprara, A., Furini, F., Malaguti, E., and Traversi, E. (2016). Solving the temporal knapsack problem via recursive Dantzig-Wolfe decomposition. *Information Processing Letters*, **116**(5), 379–386.

Christiansen, M. and Nygreen, B. (1998). A method for solving ship routing problems with inventory constraints. *Annals of Operations Research*, **81**(0), 357–378.

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

Desaulniers, G. (2010). Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations Research*, **58**(1), 179–192.

Desaulniers, G., Desrosiers, J., Ioachim, I., Solomon, M., Soumis, F., and Villeneuve, D. (1998). 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. Kluwer, Boston.

Desaulniers, G., Desrosiers, J., and Solomon, M., editors (2005). *Column Generation*. Springer, New York.

Desaulniers, G., Madsen, O., and Ropke, S. (2014). The vehicle routing problem with time windows. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, MOS-SIAM Series on Optimization, pages 119–159. Society for Industrial and Applied Mathematics, Philadelphia.

Desaulniers, G., Errico, F., Irnich, S., and Schneider, M. (2016). Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, **64**(6), 1388–1405.

Desrosiers, J., Dumas, Y., Solomon, M., and Soumis, F. (1995). Time constrained routing and scheduling. In M. Ball, T. Magnanti, C. Monma, and G. Nemhauser, editors, *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, pages 35–139. Elsevier, Amsterdam.

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

Dohn, A., Kolind, E., and Clausen, J. (2009). The manpower allocation problem with time windows and job-teaming constraints: A branch-and-price approach. *Computers & Operations Research*, **36**(4), 1145–1157.

Dohn, A., Rasmussen, M., and Larsen, J. (2011). The vehicle routing problem with time windows and temporal dependencies. *Networks*, **58**, 273–289.

Drexl, M. (2007). *On Some Generalized Routing Problems*. Ph.D. thesis, Faculty of Business and Economics, RWTH Aachen University, Aachen, Germany.

Drexl, M. (2012). Synchronization in vehicle routing—A survey of VRPs with multiple synchronization constraints. *Transportation Science*, **46**(3), 297–316.

Dumas, Y., Soumis, F., and Desrosiers, J. (1990). Optimizing the schedule for a fixed vehicle path with convex inconvenience costs. *Transportation Science*, **24**(2), 145–152.

Elhedhli, S. and Goffin, J.-L. (2005). Efficient production-distribution system design. *Management Science*, **51**(7), 1151–1164.

Feillet, D. (2010). A tutorial on column generation and branch-and-price for vehicle routing problems. *4OR*, **8**(4), 407–424.

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

Feillet, D., Dejax, P., and Gendreau, M. (2005). Traveling salesman problems with profits. *Transportation Science*, **39**(5), 188–205.

Feillet, D., Gendreau, M., and Rousseau, L.-M. (2007). New refinements for the solution of vehicle routing problems with branch and price. *INFOR*, **45**(4), 239–256.

Fink, M., Desaulniers, G., Frey, M., Kiermaier, F., Kolisch, R., and Soumis, F. (2016). Column generation for vehicle routing problems with multiple synchronization constraints. Technical Report G-2016-63, Les Cahiers du GERAD.

Fischer, F. and Helmberg, C. (2014). Dynamic graph generation for the shortest path problem in time expanded networks. *Mathematical Programming*, **143**(1–2), 257–297.

Gamache, M., Soumis, F., and Marquis, G. (1999). A column generation approach for large-scale aircrew rostering problems. *Operations Research*, **47**(2), 247–263.

Golden, B., Raghavan, S., and Wasil, E., editors (2008). *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces Series*. Springer, Berlin.

Hennig, F., Nygreen, B., and Lübbecke, M. (2012). Nested column generation applied to the crude oil tanker routing and scheduling problem with split pickup and split delivery. *Naval Research Logistics*, **59**(3–4), 298–310.

Ioachim, I., Gélinas, S., Soumis, F., and Desrosiers, J. (1998). A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, **31**(3), 193–204.

Ioachim, I., Desrosiers, J., Soumis, F., and Bélanger, N. (1999). Fleet assignment and routing with schedule synchronization constraints. *European Journal of Operational Research*, **119**(1), 75–90.

Irnich, S. (2008). Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectrum*, **30**(1), 113–148.

Irnich, S. and Desaulniers, G. (2005). Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. Solomon, editors, *Column Generation*, pages 33–65. Springer, New York.

Irnich, S., Schneider, M., and Vigo, D. (2014). Four variants of the vehicle routing problem. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, MOS-SIAM Series on Optimization, pages 241–271. Society for Industrial and Applied Mathematics, Philadelphia.

Jans, R. (2010). Classification of Dantzig-Wolfe reformulations for binary mixed integer programming problems. *European Journal of Operational Research*, **204**(2), 251–254.

Jepsen, M., Petersen, B., Spoorendonk, S., and Pisinger, D. (2008). Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, **256**(2), 497–511.

Jepsen, M., Petersen, B., and Spoorendonk, S. (2014). A branch-and-cut algorithm for the capacitated profitable tour problem. *Discrete Optimization*, **14**, 78–96.

Karabuk, S. (2009). A nested decomposition approach for solving the paratransit vehicle scheduling problem. *Transportation Research Part B*, **43**(4), 448–465.

Kohl, N., Desrosiers, J., Madsen, O., Solomon, M., and Soumis, F. (1999). 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, **33**, 101–116.

Lahyani, R., Khemakhem, M., and Semet, F. (2015). Rich vehicle routing problems: From a taxonomy to a definition. *European Journal of Operational Research*, **241**(1), 1–14.

Liberatore, F., Righini, G., and Salani, M. (2011). A column generation algorithm for the vehicle routing problem with soft time windows. *4OR*, **9**(1), 49–82.

Lübbecke, M. and Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, **53**(6), 1007–1023.

Maffioli, F. and Sciomachen, A. (1997). A mixed-integer model for solving ordering problems with side constraints. *Annals of Operations Research*, **69**(0), 277–297.

Mankowska, D., Meisel, F., and Bierwirth, C. (2014). The home health care routing and scheduling problem with interdependent services. *Health Care Management Science*, **17**(1), 15–30.

Muter, I., Cordeau, J.-F., and Laporte, G. (2014). A branch-and-price algorithm for the multidepot vehicle routing problem with interdepot routes. *Transportation Science*, **48**(3), 425–441.

Pecin, D., Pessoa, A., Poggi, M., and Uchoa, E. (2017). Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation*, **9**(1), 61–100.

Righini, G. and Salani, M. (2006). Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, **3**, 255–273.

Rothenbächer, A.-K., Drexl, M., and Irnich, S. (2017). Branch-and-price-and-cut for the truck-and-trailer routing problem with time windows. *Transportation Science*. Forthcoming.

Song, S. (2009). A nested column generation algorithm to the meta slab allocation problem in the steel making industry. *International Journal of Production Research*, **47**(13), 3625–3638.

Spliet, R. and Gabor, A. (2014). The time window assignment vehicle routing problem. *Transportation Science*, **49**(4), 721–731.

Tagmouti, M., Gendreau, M., and Potvin, J.-Y. (2007). Arc routing problems with time-dependent service costs. *European Journal of Operational Research*, **181**(1), 30–39.

Tilk, C., Rothenbächer, A.-K., Gschwind, T., and Irnich, S. (2017a). 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.

Tilk, C., Bianchessi, N., Drexl, M., Irnich, S., and Meisel, F. (2017b). Branch-and-price-and-cut for the active-passive vehicle-routing problem. *Transportation Science*. doi: 10.1287/trsc.2016.0730.

Toth, P. and Vigo, D., editors (2014). *Vehicle Routing: Problems, Methods, and Applications*. Society for Industrial and Applied Mathematics, Philadelphia.

Uygur, A. (2011). *Acceleration and Stabilization Techniques for Column Generation Applied to Capacitated Resource Management Problems*. Ph.D. thesis, Lehigh University, Bethlehem, USA.

van Eijl, C. (1995). A polyhedral approach to the delivery man problem. Technical Report 95-19, Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands.

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

Visser, T. (2015). *Synchronization in Simultaneous Vehicle and Crew Routing and Scheduling Problems with Breaks*. Master's thesis, Utrecht University, Utrecht, The Netherlands.