Branch-Cut-and-Price for Scheduling Deliveries with Time Windows in a Direct Shipping Network

Timo Gschwind*[,a], Stefan Irnich[a], Christian Tilk[a], Simon Emde[b]

[a] *Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, D-55128 Mainz, Germany.*
[b] *Fachgebiet Management Science / Operations Research, Technische Universität Darmstadt, Hochschulstraße 1, D-64289 Darmstadt, Germany.*

Abstract

In a direct shipping (or point-to-point) network, individual deliveries are round trips from one supplier to one customer and back to either the same or another supplier, i.e., a truck can only visit one customer at a time before it has to return to a supplier. We consider the multiple sources, multiple sinks case, where a given set of direct deliveries from a set of suppliers to a set of customers must be scheduled such that the customer time windows are not violated, the truck fleet size is minimal, and the total weighted customer waiting time is as small as possible. Direct shipping policies are, for instance, commonly employed in just-in-time logistics (e.g., in the automotive industry) or in humanitarian logistics. We present an exact branch-cut-and-price algorithm for this problem, which is shown to perform well on instances from the literature and newly generated ones. We also investigate under what circumstances bundling suppliers in so-called supplier parks actually facilitates logistics operations under a direct shipping policy.

Key words: direct deliveries, branch-cut-and-price, weighted customer waiting times, just-in-time logistics

1. Introduction

Transportation networks for the delivery of physical goods are typically distinguished by their structure as milk runs, crossdocking, or direct shipping (Chopra and Meindl, 2015, Chpt. 14). Milk run networks consolidate multiple customers or suppliers on one tour, whereas in a crossdocking network, shipments are routed through a transshipment hub and consolidated there. Such network structures are useful to improve economies of transportation if a shipment is less than a truckload (Du *et al.*, 2007).

In direct shipping networks (also called point-to-point networks), which are the focus of this paper, goods are moved directly from source to sink, without any intermediary step. Such networks are widely used in many industries, e.g., the automotive (Boysen *et al.*, 2015). Direct shipments are also common in many retail supply chains, not necessarily to supply stores directly, but to ship goods from supplier or wholesaler distribution centers to regional dispatching points (also referred

*Corresponding author.
Email addresses: gschwind@uni-mainz.de (Timo Gschwind), irnich@uni-mainz.de (Stefan Irnich), tilk@uni-mainz.de (Christian Tilk), emde@bwl.tu-darmstadt.de (Simon Emde)

to as pool points), where they are then further distributed to local retail stores via milk run (Chen, 2008). Further applications are disaster relief, where humanitarian goods must be moved from national emergency stockpiles to local staging areas in case of disasters (Knott, 1987; Balcik *et al.*, 2008), or transporting biomass fuel (Rauch *et al.*, 2007).

Apart from being the fastest way to move goods from A to B, direct shipping can also be considered the easiest shipping policy to implement and coordinate. It has been shown that it is the most efficient strategy if the economic lot size of the customers is close to the vehicle capacity (Gallego and Simchi-Levi, 1990; Barnes-Schuster and Bassok, 1997). For these reasons, direct deliveries are commonly used for *just-in-time* (JIT), high-velocity, high-bulk, perishable, and specialty goods (Chen, 2008).

In this context, we tackle the *direct delivery scheduling problem in a network* (DDSP-N): given a set of *suppliers* (e.g., distribution centers or supplier plants) from which a set of *customers* (e.g., pool points or OEM plants) need to be served via direct delivery (that is, neither suppliers nor customers can be aggregated on milk runs), which truck should make which delivery at what time? Since deliveries are usually made by 3rd party logistics providers, trucks can switch between suppliers. Moreover, since direct deliveries are particularly common for JIT and high-velocity goods, deliveries have time windows which must not be violated. From the viewpoint of the logistics provider, it is desirable to minimize, on the one hand, the cost of the truck fleet. On the other hand, customer waiting times should not be excessive, as this may lead to significant contractual penalties in case of delays (Boysen *et al.*, 2015).

The main contributions of this paper are as follows. First, we extend the model by Emde and Zehtabian (2017) to account for shipping networks with more than one supplier. Second, we develop a powerful exact branch-cut-and-price algorithm to solve this problem, the first exact solution method (apart from a default solver) for the DDSP-N. Finally, through computational experiments, we derive some managerial insight into the usefulness of clustering suppliers (e.g., in so-called "supplier parks") under a direct shipping policy.

The remainder of this paper is structured as follows. In Section 2, we review the pertinent literature. In Section 3, we formally define the DDSP-N, review the mixed-integer programming model of Emde and Zehtabian (2017), and present a path-based formulation. Section 4 introduces our custom-tailored branch-cut-and-price schemes, which we test in Section 5. Finally, Section 6 ends the paper with conclusion and outlook.

2. Literature Review

Direct shipping networks, although widespread in practice, have so far almost exclusively received attention from a strategic or supply chain management viewpoint, with the goal of determining optimal long-term supply policies considering both transportation as well as inventory holding cost. Such studies were published, e.g., by Gallego and Simchi-Levi (1990); Barnes-Schuster and Bassok (1997); Chopra (2003). Meyer and Amberg (2017) review model-based approaches to strategically select delivery policies and frequencies in the automotive context. Boysen *et al.* (2015) survey part logistics in the automotive industry and discuss point-to-point networks in this context in Section 3.2. of their paper.

To the best of our knowledge, the only paper explicitly dealing with direct shipping from a scheduling perspective, where a set of given trips needs to be scheduled within a finite planning horizon, is (Emde and Zehtabian, 2017). The authors introduce the problem and develop several heuristics, restricting their efforts to the one-supplier multiple-receiver case. In this paper, we

extend their problem by considering shipping networks with more than one supplier, as they are often encountered in practice.

Distributing goods from one or more suppliers to multiple customers bears some resemblance to classic (multi-depot) vehicle routing problems (VRPs, Vigo and Toth, 2014). However, these models are not immediately applicable to DDSP-N because trucks must return to a supplier between customer visits for loading. Moreover, the objective of DDSP-N is atypical for VRP. We discuss parallels between these two problems further in Section 3.2.

Routing problems that consider customer waiting times in the form of minimizing the sum of arrival times (without considering release dates) are referred to as travelling repairman problems (TRP), also called deliveryman or minimum latency problems. Literature on TRP with multiple vehicles is very limited, however. (Fakcharoenphol *et al.*, 2007; Luo *et al.*, 2014; Nucamendi-Guillén *et al.*, 2016; Angel-Bello *et al.*, 2017) are among the few contributions, assuming a given fixed number of vehicles and disregarding time windows. Other routing problems with latency objectives, like the cumulative capacitated VRP (e.g., Ribeiro and Laporte, 2012) or the school bus routing problem (surveyed by Park and Kim, 2010) are also structurally different from the DDSP-N because they assume given customer demands and limited vehicle capacities, which are immaterial for point-to-point deliveries.

The DDSP-N also bears some resemblance to the multi-depot vehicle scheduling problem, where a set of timetabled trips must be assigned to a fleet of vehicles (e.g., Carpaneto *et al.*, 1989; Dell'Amico *et al.*, 1993; Ribeiro and Soumis, 1994). Direct deliveries are typically not exactly timetabled like a bus schedule, however, but merely have time windows which must not be violated.

Finally, scheduling direct deliveries is reminiscent of parallel machine scheduling if we interpret trucks as machines, trips as jobs, and transfer times between suppliers as setup times. Machine scheduling with setup times and related problems have been surveyed by Allahverdi *et al.* (2008) and Pinedo (2016). However, minimizing the number of machines (trucks) is an unusual objective in the machine scheduling context (Yu and Zhang, 2009). If the number of vehicles/machines were fixed, our problem would be structurally similar (albeit not identical) to scheduling a set of jobs with given time windows and processing times on a set of identical parallel machines to minimize the total weighted flow time, where switching jobs requires a sequence-dependent setup time. In machine scheduling notation as introduced by Graham *et al.* (1979), this corresponds to the tuple $[P|r_j, d_j, S_{ij}|\sum w_j F_j]$. To the best of our knowledge, this problem has not been solved before.

3. Definition and Formulations of the DDSP-N

The DDSP-N can be formally defined as follows: the physical network $D = (\mathcal{L}, \mathcal{A})$ has the vertex set $\mathcal{L}$ representing the relevant physical locations, i.e., $\mathcal{L} = \{0, 0'\} \cup \mathcal{S} \cup \mathcal{N}$, where 0 and $0'$ are start and destination depots of the trucks, $\mathcal{S}$ is the set of suppliers (=terminals, distribution centers), and $\mathcal{N}$ the set of customers (=OEMs, pool points).

Moreover, we are given the set I of trips that must be performed. Each trip $i \in I$ is defined by the following additional attributes:

- s_i: the supplier $s_i \in \mathcal{S}$ from where goods are shipped;
- n_i: the customer $n_i \in \mathcal{N}$ who receives these goods;
- r_i: the ready times, i.e., the earliest time when a truck can depart from supplier s_i with the shipment associated with the trip;
- d_i: the due date, i.e., the latest time the service has to be completed at customer n_i;

p_i: the processing time of the trip, including the loading time of the truck at the supplier, the driving time to, and the unloading time at the customer, plus the time it takes to do any other activities necessary to ready the truck for its next trip (possibly refueling or breaks for the driver);

w_i: the weight of the trip to be used in the minimum wait (=earliness) objective (see below).

We assume that a truck can travel along the following connections: between

1. origin depot and all suppliers, i.e., $(0, s) \in \{0\} \times \mathcal{S} \subset \mathcal{A}$,
2. all customers and all suppliers, i.e., $(n, s) \in \mathcal{N} \times \mathcal{S} \subset \mathcal{A}$,
3. all trips with their associated supplier-customers pair, i.e., $\{(s_i, n_i) : i \in I\} \subset \mathcal{A}$, and
4. all customers and the destination depot, i.e., $(n, 0') \in \mathcal{N} \times \{0'\} \subset \mathcal{A}$.

Finally, we define the travel time $\tau(n, s)$ for the connections $(n, s) \in \mathcal{N} \times \mathcal{S}$ between customer and suppliers.

The task of the DDSP-N is to feasibly schedule all trips and to assign them to trucks such that (i) the number of employed trucks is minimal and (ii) the weighted earliness of all trips is minimal. Emde and Zehtabian (2017) define the DDSP-N with a hierarchical objective with priority on (i).

To precisely define the *weighted earliness*, we introduce schedule variables $T_i \in \mathbb{R}$ for all trips $i \in I$. The variable T_i describes the point in time when the trip execution is started (with loading at its supplier s_i). These schedule variables are bounded by $r_i \leq T_i \leq d_i - p_i$. If two trips $i, j \in I$ are assigned to the same truck, then either i precedes j leading to $T_i + p_i + \tau(n_i, s_j) \leq T_j$, or j precedes i implying $T_j + p_j + \tau(n_j, s_i) \leq T_i$. The weighted earliness objective is given by $\sum_{i \in I} w_i(T_i - r_i)$.

3.1. Compact Formulation of Emde and Zehtabian (2017)

The single-supplier variant of the DDSP-N was modeled by Emde and Zehtabian (2017). With slight modifications, this formulation also models the multiple-supplier case. The mixed-integer programming (MIP) formulation of Emde and Zehtabian uses, in addition to the schedule variables T_i for all $i \in I$, binary connection variables x_{ij} for all $i \in I \cup \{0\}$ and $j \in I \cup \{0'\}$. For two trips $i, j \in I$, the variable x_{ij} is one if i is performed directly before j by the same truck. The variables x_{0j} are one if $j \in I$ is the first trip performed by a truck. Similarly, $x_{i0'}$ equal to one means that $i \in I$ is the last trip performed by a truck. The model now reads as follows:

$$\min \sum_{j \in I} M \cdot x_{0j} + \sum_{i \in I} w_i \cdot (T_i - r_i) \tag{1a}$$

$$\text{s.t.} \sum_{j \in I \cup \{0'\}} x_{ij} = 1 \qquad \forall i \in I \tag{1b}$$

$$\sum_{i \in I \cup \{0\}} x_{ij} = 1 \qquad \forall j \in I \tag{1c}$$

$$(1 - x_{ij}) \cdot M + T_j \geq T_i + p_i + \tau(n_i, s_j) \qquad \forall (i, j) \in I \times I \tag{1d}$$

$$r_i \leq T_i \leq d_i - p_i \qquad \forall i \in I \tag{1e}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I \cup \{0\}, j \in I \cup \{0'\} \tag{1f}$$

The objective function (1a) gives priority to the minimization of the fleet (using a big-M constant). The secondary objective is the minimization of the total weighted earliness. Constraints (1b) and (1c) make sure that each trip has exactly one successor and predecessor, respectively, which can either be another trip or one of the depots 0 or $0'$. Inequalities (1d) make it impossible for a truck to

make two trips concurrently. Finally, (1e) ensure that trips are processed within their time windows and (1f) define the domain of the binary connection variables.

Obviously, constraints (1d) can be eliminated from formulation (1) for those pairs $(i, j) \in I \times I$ where i cannot precede j, i.e., $r_i + p_i + \tau(n_i, s_j) > d_j - p_j$. This strengthens the linear relaxation of formulation (1).

3.2. Path-based Formulation

We now model the movement of a truck as a route in a network. We define the underlying digraph $D = (V, A)$ with vertex set $V = I \cup \{0, 0'\}$ and arc set A. Each trip vertex $i \in I$ has an associated time window $[e_i, \ell_i] := [r_i, d_i - p_i]$ (see also (1e)) and service time p_i. For the origin and destination depot (0 and $0'$), we assume non-constraining time windows $[e_0, \ell_0] = [e_{0'}, \ell_{0'}]$ and define the weights $w_0 := w_{0'} := 0$.

The arc set A is a subset of $(\{0\} \times I) \cup (I \times I) \cup (I \times \{0'\})$, where we define travel times $t_{0j} := 0$ for $(0, j) \in \{0\} \times I$, $t_{ij} := p_i + \tau(n_i, s_j)$ for $(i, j) \in I \times I$, and $t_{i,0'} := 0$ for $(i, 0') \in I \times \{0'\}$. Those arcs $(i, j) \in A$ that fulfill $e_i + t_{ij} \leq \ell_j$ are feasible with respect to the travel and service times and time windows defined above, all other arcs can be eliminated. Figure 1 depicts the two modelling possibilities of an DDSP-N instance and two representations of a solution.

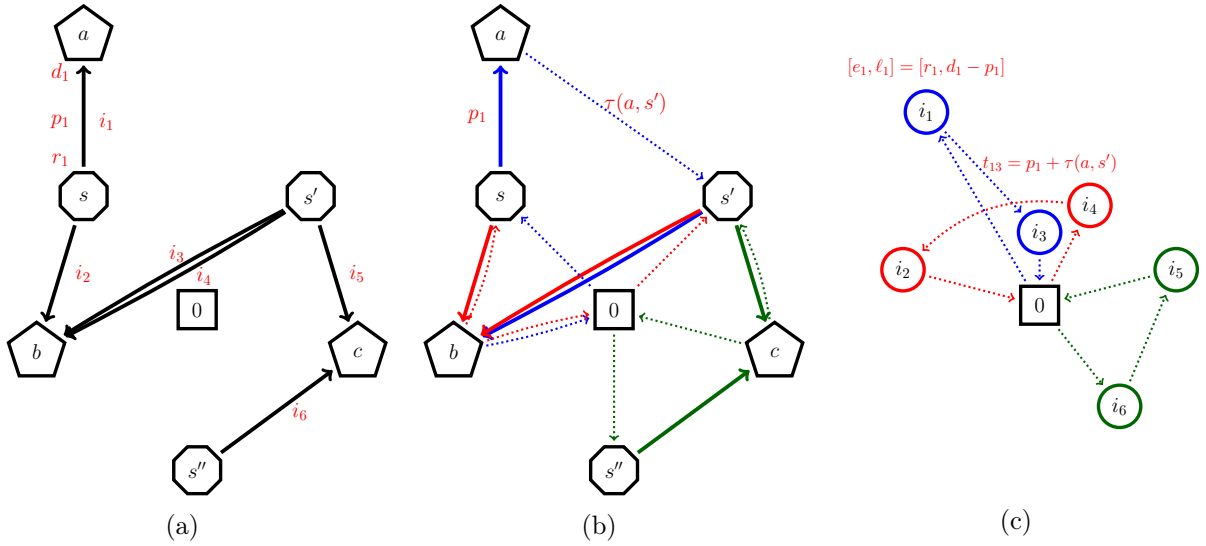


Figure 1: (a) DDSP-N instance with three suppliers $\{s, s', s'''\}$, three customers $\{a, b, c\}$, and six trips $I = \{i_1, i_2, \ldots, i_6\}$ with $i_1 = (s, a)$, $i_2 = (s, b)$, $i_3 = i_4 = (s', b)$, $i_5 = (s', c)$, and $i_6 = (s'', c)$; (b) Solution with three trucks; (c) Trip-based digraph (V, A) with the corresponding solution

A DDSP-N route P is an elementary 0-$0'$-path in D that is feasible with respect to the travel and service times and time windows defined above. Formally, for a route $P = (i_0, i_1, \ldots, i_p, i_{p+1})$ (recall that $i_0 = 0$ and $i_{p+1} = 0'$) there must exist a schedule $(T_0, T_1, \ldots, T_p, T_{p+1}) \in \mathbb{R}^{p+2}$ satisfying the following two constraints:

$$T_k \in [e_{i_k}, \ell_{i_k}] \qquad \forall k \in \{0, 1, 2, \ldots, p+1\} \tag{2a}$$

$$T_{k-1} + t_{i_{k-1}, i_k} \leq T_k \qquad \forall k \in \{1, 2, \ldots, p+1\}. \tag{2b}$$

These are classical time-window constraints known from the vehicle routing problem with time windows (VRPTW, Desaulniers *et al.*, 2014).

In the case of feasibility, using the *as-early-as-possible* schedule, i.e., $T_0 := e_{i_0}$ and $T_k := \max\{e_{i_k}, T_{k-1} + t_{i_{k-1},i_k}\}$ for $k = 1, 2, \ldots, p+1$, the accumulated weighted earliness of the route P is

$$w_P := \sum_{k=0}^{p+1} w_{i_k} \cdot (T_k - e_{i_k}) = \sum_{k=1}^{p} w_{i_k} \cdot (T_k - r_{i_k}).$$

Let Ω be the set of all feasible DDSP-N routes, and a_{iP} an indicator whether trip $i \in I$ is performed by route $P \in \Omega$. The set-partitioning formulation uses variables λ_P for $P \in \Omega$ to select the routes the solution comprises:

$$\min \sum_{P \in \Omega} (M + w_P)\lambda_P \tag{3a}$$

$$\text{s.t.} \sum_{P \in \Omega} a_{iP}\lambda_P = 1 \qquad \forall i \in I \tag{3b}$$

$$L \leq \sum_{P \in \Omega} \lambda_P \leq U \tag{3c}$$

$$\lambda_P \in \{0, 1\} \qquad \forall P \in \Omega \tag{3d}$$

Analogous to (1a), the objective (3a) is hierarchical minimizing the fleet size first and the total weighted earliness second. The partitioning constraints (3b) ensure that each trip is performed exactly once. The fleet size is bounded from below (above) by L (U) with the help of constraints (3c) (see Section 4.3), and (3d) are the binary constraints.

4. Branch-Cut-and-Price Algorithm

For solving the linear relaxation of the path-based formulation (3), we use a column-generation algorithm (Desaulniers *et al.*, 2005). Starting with a subset $\Omega' \subset \Omega$ of the feasible routes, the linear relaxation of formulation (3) defined over Ω' is denoted as the *restricted master program* (RMP). The column-generation algorithm alternates between the re-optimization of the RMP and the solution of the column-generation pricing problem that adds negative reduced-cost variables to the RMP, if one exists (see next section). If no reduced-cost variables exist, the current linear relaxation is solved to optimality. Branching is required to finally ensure integer solutions.

Section 4.1 presents the pricing problem of formulation (3). To strengthen the linear relaxation of formulation (3), Section 4.2 briefly describes known valid inequalities for the RMP and separation strategies for them. In Section 4.3, we present our DDSP-N-tailored branching schemes.

4.1. Pricing Problem

Let $\pi_i, i \in I$, be the dual prices of the partitioning constraints (3b) and let μ_L and μ_U be the dual prices of fleet size constraints (3c). The task of the pricing problem is to identify at least one feasible route $P \in \Omega$ with negative reduced cost

$$\tilde{c}_P = M + w_P - \mu_L - \mu_U - \sum_{i \in I} a_{iP}\,\pi_i$$

or guarantee that no such route exists. This problem is a variant of the elementary shortest path problem with resource constraints (ESPPRC) which can be solved by means of a dynamic-programming labeling algorithm (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 all necessary information on the resource consumption up to the endpoint of a partial path. Labels are propagated along the network arcs by resource extension functions (REFs, Desaulniers *et al.*, 1998; Irnich, 2008). Dominance procedures are invoked to identify and discard those partial paths and their labels that cannot lead to an optimal SPPRC solution. The main difference to classical ESPPRCs (as pricing problems of VRP variants) is that there are no routing costs in the DDSP-N. Instead, the pricing problem has to deal with the accumulated weighted earliness cost. The associated reduced cost of an arc $(i,j) \in A$ can be defined as

$$\tilde{c}_{ij} := -\frac{1}{2}(\pi_i + \pi_j) + \begin{cases} M, & \text{if } i = 0 \\ 0, & \text{otherwise} \end{cases}, \tag{4}$$

where we define $\pi_0 := \pi_{0'} := \mu_L + \mu_U$. With this definition, $\tilde{c}_P = w_P + \sum_{(i,j)\in A(P)} \tilde{c}_{ij}$ holds for all routes $P \in \Omega$.

A label in the DDSP-N is associated with a partial path ending at vertex $i \in V$ and has the attributes $(C_i, T_i, (S_i^v))$, where C_i is the accumulated reduced cost (including the weighted earliness), T_i is the time at vertex i (according to the as-early-as-possible schedule), and (S_i^v) is the counter how often a trip $v \in I$ has already been performed by the partial path. The initial label at vertex 0 is $(C_0, T_0, (S_0^v)) = (0, e_0, \mathbf{0})$. An arbitrary label $(C_i, T_i, (S_i^v))$ is extend along an arc $(i,j) \in A$ creating a new label $(C_j, T_j, (S_j^v))$ using the following REFs:

$$C_j = REF^{rdc}(C_i, T_i, (S_i^v)) = C_i + \tilde{c}_{ij} + w_j \cdot (T_i + t_{ij} - e_j)^+ \tag{5a}$$
$$T_j = REF^{time}(C_i, T_i, (S_i^v)) = \max\{e_j, T_i + t_{ij}\} \tag{5b}$$
$$S_j^v = REF^{visit,v}(C_i, T_i, (S_i^v)) = S_i^v + \delta_{jv}, \qquad \forall v \in I \tag{5c}$$

where δ_{jv} is the Kronecker delta equal to one if $j = v$, and 0 otherwise. The new label $(C_j, T_j, (S_j^v))$ is feasible if all of the following constraints are fulfilled:

$$T_j \le \ell_j \tag{6a}$$
$$S_j^v \le 1 \qquad \forall v \in I \tag{6b}$$

Note that for any feasible route $P \in \Omega$, the propagation of the initial label along the route's arcs according to (5) yields a final label $(C_{0'}, T_{0'}, (S_{0'}^v))$ at vertex $0'$ with $C_{0'} = \tilde{c}_P$.

Since the REFs (5) are non-decreasing and attributes are bounded only from above by (6), labels can be compared with the standard $\le$-operator to impose the following valid dominance rule: Let $(C_i, T_i, (S_i^v))$ and $(C_i', T_i', (S_i'^v))$ be two different labels associated with the same vertex $i \in I$. Then, $(C_i, T_i, (S_i^v))$ dominates $(C_i', T_i', (S_i'^v))$ if $C_i \le C_i'$, $T_i \le T_i'$, and $S_i^v \le S_i'^v$ for all $v \in I$ hold. All dominated labels can be discarded as long as, for each of them, at least one dominating label is kept.

Despite the general success of bidirectional labeling (see, e.g., Righini and Salani, 2006; Tilk *et al.*, 2017), we do not employ it here. The reason is that backward labeling is non-trivial for the DDSP-N due to the computation of the weighted earliness w_P. The simple case, as done in the VRPTW, is a backward labeling that propagates an *as-late-as-possible* schedule. This schedule

however overestimates the true weighted earliness making the definition of an effective dominance rule impossible. An exact computation of the weighted earliness in backward labeling can be established using a time-cost tradeoff function such as the piecewise linear tradeoff functions used in (Ioachim *et al.*, 1998; Tilk *et al.*, 2018). However, the backward case becomes much more difficult compared to the forward case so that we expect a bidirectional labeling to finally not pay off.

4.2. Cutting

Different classes of valid inequalities to strengthen the linear relaxation were successfully applied in branch-cut-and-price algorithms for many VRP variants. We use two known classes of valid inequalities for the VRPTW to strengthen the linear relaxation of formulation (3). First, we use the *k-path inequalities* introduced by Kohl *et al.* (1999) for the VRPTW with $k = 2$. Let $W \subset I$ be a subset of customers that cannot be visited by a single vehicle due to time window 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 $\sum_{P\in\Omega} \sum_{(i,j)\in\delta^-(W)} b_{ijP}\lambda_P \geq 2$, where b_{ijP} indicates whether route P uses arc (i, j). Note that the 2-path inequalities are robust cuts, meaning that their incorporation does not require structural adjustments in the pricing problem algorithm. Only the dual price of each 2-path inequality for a subset W needs to be subtracted from the reduced cost of the arcs in $\delta^-(W)$. We use the heuristic proposed by Kohl *et al.* (1999) to generate candidate sets W. Each candidate set is then tested if it has to be served by at least two vehicles (using a standard dynamic-programming based approach to solve the respective traveling salesman problem with time windows over $W \cup \{0, 0'\}$), and if so, the corresponding inequality is added.

The second class of valid inequalities that we use are *subset-row inequalities* (Jepsen *et al.*, 2008) defined on sets $U \subset I$ of cardinality three. They are given by $\sum_{P\in\Omega} \lfloor \frac{h_P}{2} \rfloor \lambda_P \leq 1$, where h_P is the number of times route P visits a customer in U. Violated subset-row inequalities can be separated by straightforward enumeration. The addition of subset-row inequalities to the RMP, however, requires some adjustments in the pricing problem algorithm. The value of the dual price of a subset-row inequality defined on a vertex set U has to be subtracted from the reduced cost of a label for every second visit to vertices in U. Therefore, an additional resource is needed for each inequality counting the number of times that a vertex in U is visited. For details on the adjustments see (Jepsen *et al.*, 2008).

4.3. Branching

Branching is required to finally ensure integer solutions of formulation (3). Let $\bar{\lambda}_P$ be the values of the corresponding decision variables λ_P, $P \in \Omega$, and let $\bar{x}_{ij} := \sum_{P\in\Omega} b_{ijP}\bar{\lambda}_P$, $(i, j) \in A$. We use two different branching schemes that are computationally evaluated in Section 5.2. In the first scheme, we apply a standard two-stage hierarchical branching as it is employed in most branch-cut-and-price approaches for VRP variants. In the first stage, we branch on the number of vehicles, if fractional, by setting the bounds in constraint (3c) to $L := \lceil \sum_{P\in\Omega} \bar{\lambda}_P \rceil$ in one branch and $U := \lfloor \sum_{P\in\Omega} \bar{\lambda}_P \rfloor$ in the other. If the number of vehicles is integer, the branching in the second stage is employed. There, we branch on arcs by choosing an arc $(i, j) = \arg\max_{(i,j)\in A}\{(1-|\bar{x}_{ij}-0.5|)w_j\}$. The zero-branch is implemented by eliminating (i, j) from the network D, while for the one-branch all ingoing arcs of j and all outgoing arcs of i are eliminated except for the selected arc (i, j).

In the second scheme, we apply a three-stage hierarchical branching by using branching on time windows of vertices after branching on the number of vehicles and before branching on arcs. Branching on time windows in column-generation approaches for time constrained routing problems was introduced in (Gélinas *et al.*, 1995). The idea is that there may be two or more routes visiting

the same vertex at different points in time in a fractional solution of the RMP while in an integer solution, each vertex is visited at a unique point in time. Such fractional solutions can be cut off by creating two branches that split the time window of a specific vertex in two disjunct parts. Branching on time windows is implemented as follows: We select a vertex i for which the difference between the latest and the earliest visit time at this vertex in the current solution is maximal, i.e., $i = \arg\max\{\max_{P\in\Omega:a_iP=1}\{T_i^P\} - \min_{P\in\Omega:a_iP=1}\{T_i^P\}\}$, where T_i^P denotes the visit time at vertex i on route $P \in \Omega$. The time window is then split into $[e_i, \bar{t}_i - 1]$ and $[\bar{t}_i, \ell_i]$, where $\bar{t}_i = \lceil \sum_{P\in\Omega} T_i^P \bar{\lambda}_P \rceil$ is the average visit time at vertex i in the fractional solution. Note that this branching rule does not ensure integrality of the route variables, since there can also be two or more routes visiting a vertex at the same time if the solution is fractional. Therefore, we have to apply branching on arcs in a third stage to complete the branching scheme.

To accelerate the overall solution process, we apply *strong branching* at the second and third stages. The decision on which arc or time window branching is performed is then taken according to the product rule (Achterberg, 2007, p. 62). A best-bound-first strategy is applied as node-selection rule, because our primary goal is to improve the dual bound.

5. Computational Results

In this section, we report computational results for our branch-cut-and-price algorithm. All results were obtained using a standard PC with an Intel(R) Core(TM) i7-5930k processor clocked at 3.5 GHz, 64 GB RAM, and Windows 7 Enterprise. 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 RMP. We use a hard CPU time limit of 3600 seconds for all computations. To speed up the solution of the pricing problem, we use the *ng-path relaxation* (Baldacci *et al.*, 2011) as it is common for VRPs. Moreover, to speed up the column-generation process, we use arc-reduced networks as heuristic pricers (Irnich and Desaulniers, 2005, p. 57). Cuts are only separated at the root node. Subset-row cuts are separated up to a maximum of 100 cuts and the number of cuts involving a specific customer is restricted to 5. Moreover, 2-path cuts are separated for sets of up to 15 customer and up to a maximum of 100 cuts.

In Section 5.1, we introduce the instances we used to test our branch-cut-and-price algorithm. Section 5.2 reports results for the different branching strategies and a comparison with (Emde and Zehtabian, 2017) for the single supplier variant is given. Finally, Section 5.3 provides managerial insight into the usefulness of clustering suppliers into supplier parks.

5.1. Instances

We tested our algorithm on the single supplier instances from (Emde and Zehtabian, 2017). A single supplier instance is defined by a set of trips I, each having a certain release and due date as well as a processing time and a weight in the objective. Emde and Zehtabian (2017) generated two classes with 10 instances each, in which the number of trips is 25 and 125, respectively. The planning horizon is one day and a time unit is ten minutes. All other parameters are drawn at random from given distributions.

To deal with the multi supplier case, we created additional classes of instances in a similar way: Each trip is associated with a random supplier and travel times between customers and suppliers based on Euclidean distances are given. Therefore, each physical location is defined by a x- and y-coordinate taken uniformly random from $[1, 50]$ and the processing times p_i are defined as travel distance plus a service time randomly taken from $[1, 10]$. Time windows are randomly created as

follows: r_i is taken randomly from $[1, 144]$ and d_i is randomly taken from $[r_i + p_i + 3, r_i + p_i + 24]$. Moreover, we assume that each client orders multiple trips during the planning horizon such that there exist five to ten trips to each customer location, which is realistic for JIT distribution systems (Chuah and Yingling, 2005). In total, we created 24 new classes consisting of five instances each. The classes differ in the number of trips (50,100,150, or 200), the number of suppliers (2,4,6), and the length of the planning horizon (1 or 2 days). For the 2-day planning horizon, we also double the time-window width of trips, i.e., r_i is taken random from $[1, 288]$ and d_i is randomly taken from $[r_i + p_i + 6, r_i + p_i + 48]$.

5.2. Results

In a first test, we compare the two-stage and the three-stage branching schemes with and without strong branching (SB). Table 1 summarizes the results for the different branching strategies. Therein, instances are grouped by the number $|S|$ of suppliers and the planning horizon (PH) resulting in 6 groups with 20 instances each. The table contains for each instance group and each of the four branching strategies the number of solved instances (#Opt), the average solution time in seconds (T), and the average number of nodes in the branch-and-bound tree (#BB).

Comparing the branching schemes, the three-stage branching scheme is clearly superior to the two-stage branching. Without strong branching, the branch-cut-and-price algorithm with the three-stage branching scheme is able to solve 28 instances more and the solution time decreases by around 40% on average. Note that strong branching only improves the two-stage branching scheme while it slightly worsens the performance of the three-stage branching scheme. Compared to the two-stage branching scheme with strong branching, the branch-cut-and-price algorithm with the three-stage branching scheme is able to solve only 14 more instances and the average solution time decreases by 21%. This behavior can be attributed to the number of solved nodes in the branch-and-bound tree that decreases by nearly 95% in the two-stage scheme and only by around 85% in the three-stage scheme when strong branching is applied. The latter percentage does not suffice to compensate the additional computation time induced by the evaluation of branching candidates in the strong branching procedure.

In all settings, there is no significant difference in the number of solved instances as well as in the average solution time when the number of suppliers varies. On the other hand, the number of solved instances significantly decreases and the solution times significantly increase in all settings for the instance groups with a two-day planning horizon.

Table 1: Comparison of different Branching Strategies

Instances		two-stage			SB two-stage			three-stage			SB three-stage		
|S|	PH	#Opt	T	#BB	#Opt	T	#BB	#Opt	T	#BB	#Opt	T	#BB
2	1	11	1639.3	6738	11	1641.5	355	16	946.3	3013	14	1178.1	444
4	1	11	1722.4	6415	13	1452.1	297	17	894.6	2431	17	1059.1	345
6	1	11	1721.3	6358	14	1243.3	238	16	787.3	1886	16	906.6	282
2	2	9	2100.5	2086	10	1957.0	116	13	1390.5	676	13	1601.7	129
4	2	9	2153.1	1687	11	1984.0	67	11	1691.7	649	11	1765.8	113
6	2	7	2634.3	3386	11	2096.9	136	13	1517.2	773	13	1679.6	145
Total		58	1995.2	4445	70	1729.2	202	86	1204.6	1571	84	1365.1	243

In Table 2, we give more detailed results on our best algorithm (three-stage branching scheme without strong branching). In addition, we now group the instances by the number of customers and

the length of the planning horizon, since the number of suppliers does not impact the computation time significantly. The table contains for each instance group the number of instances (#), the number of solved instances, and the minimum, average, and maximum computation time needed to solve an instance to proven optimality. Moreover, the table presents, for the unsolved instances, two types of gaps between the overall lower bound (at termination) and the best known solution: (i) the number of times that the number of vehicles in the lower bound differs from the best known solution by exactly one (#Veh+1) and (ii) the minimum, average, and maximum gap in weighted earliness (WE [%]). The best known solutions are taken over all test and pretests we have performed. Note that the difference in the number of vehicles is at most one in all unsolved instances and that we compute the weighted earliness gap only over all unsolved instances and only over those in which the number of vehicles in the lower bound is equal to the number of vehicles in the best known solution. The table shows that the number of customers significantly impacts the solution time. Regarding the gaps, the number of vehicles in the lower bound differs only five times from the best known solution and the gap in weighted earliness is around 5% on average.

Table 2: Detailed Results of our best Strategy

Instances				T				WE [%]		
|C|	PH	#	#Opt	min	avg	max	#Veh+1	min	avg	max
50	1	15	15	0.1	0.5	2.2	–	–	–	–
100	1	15	15	0.7	14.8	105.8	–	–	–	–
150	1	15	13	20.5	829.2	3600.0	0	0.03	0.14	0.24
200	1	15	6	177.4	2659.8	3600.0	2	0.03	1.39	5.71
50	2	15	15	0.3	1.9	5.8	–	–	–	–
100	2	15	14	26.6	499.7	3600.0	0	0.05	0.05	0.05
150	2	15	7	134.5	2243.0	3600.0	1	0.11	6.24	17.51
200	2	15	1	416.4	3388.0	3600.0	2	0.04	8.33	33.36
Total		120	86	0.1	1204.6	3600.0	5	0.03	5.30	33.36

Next, we test our algorithm on the single supplier instances of Emde and Zehtabian (2017). Table 3 summarizes the results. The table contains for both instance classes the number of solved instances and the average solution times of our branch-cut-and-price algorithm as well as of the solution of the compact model in CPLEX carried out by Emde and Zehtabian (2017). Moreover, we compare the best heuristic solutions found in Emde and Zehtabian (2017) regarding the difference in the number of vehicles and the gap in terms of weighted earliness compared to the optimal solution. Summarizing, our branch-cut-and-price algorithm is able to solve all single supplier instances to optimality with an average computation time of 0.1 seconds for the 25 trip instances and 104.2 seconds for the 125 trip instances, while CPLEX solves all 25 trip instances in an average of 6.0 seconds but cannot solve any of the 125 trip instances in 1800 seconds of computation time. Moreover, the detailed results show that the computation times of our branch-cut-and-price algorithm never exceed 485 seconds on the single supplier instances. Table 3 also reveals that the heuristic of Emde and Zehtabian (2017) is able to meet the minimum number of vehicles for all 25 trip instances and finds 9(1) solutions with 1(2) more vehicles than the minimum number of vehicles for the large 125 trip instances. The average gap in weighted earliness over the instances where they are able to find the optimal number of vehicles is 4.1%.

Table 3: Comparison with Emde and Zehtabian (2017) on Single Supplier Instances

		BCP		CPLEX EZ		Heuristic EZ		
|C|	#inst	#Opt	T	#Opt	T	#Opt	#Veh (+2/1)	WE [%]
25	10	10	0.1	10	6.0	3	0/0	4.1
125	10	20	104.2	0	1800.0	0	1/9	–
Total	20	40	52.1	10	903.0	3	1/9	4.1

5.3. Effect of Supplier Centralization

So far, we have assumed that suppliers are randomly dispersed on the map. However, in reality, this is not necessarily the case. Especially in JIT industries, where short delivery cycles are key, suppliers often form clusters close to their customers, that is, so-called supplier parks (e.g., Larsson, 2002; Pfohl and Gareis, 2005). Such supplier parks are supposed to facilitate JIT logistics, but for the companies involved relocating may be expensive and make it impossible to benefit from wage level and taxation differences across country or regional borders. To investigate whether pooling suppliers in regional clusters is actually beneficial when employing direct deliveries, we conduct the following series of experiments.

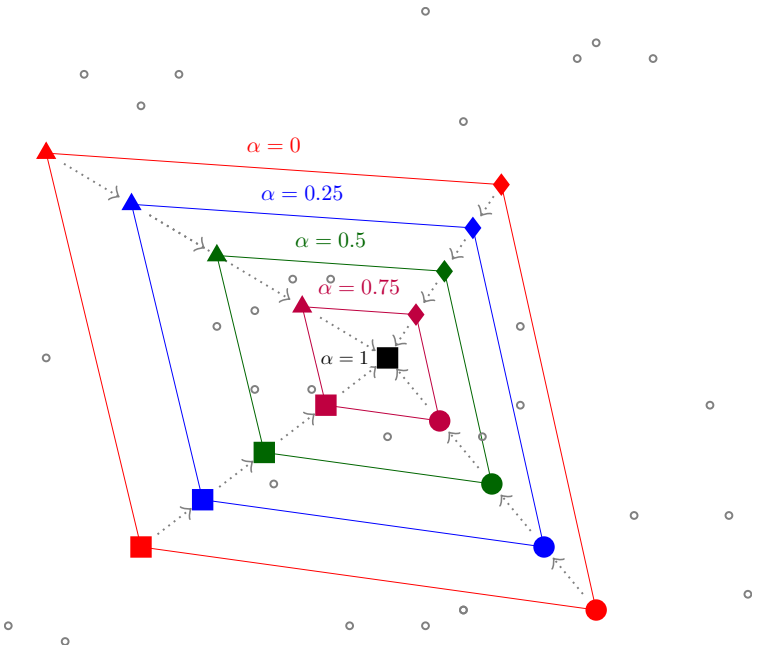


Figure 2: Clustering of suppliers with values $\alpha \in \{0, 0.25, 0.5, 0.75, 1.0\}$ for a base instance with four suppliers

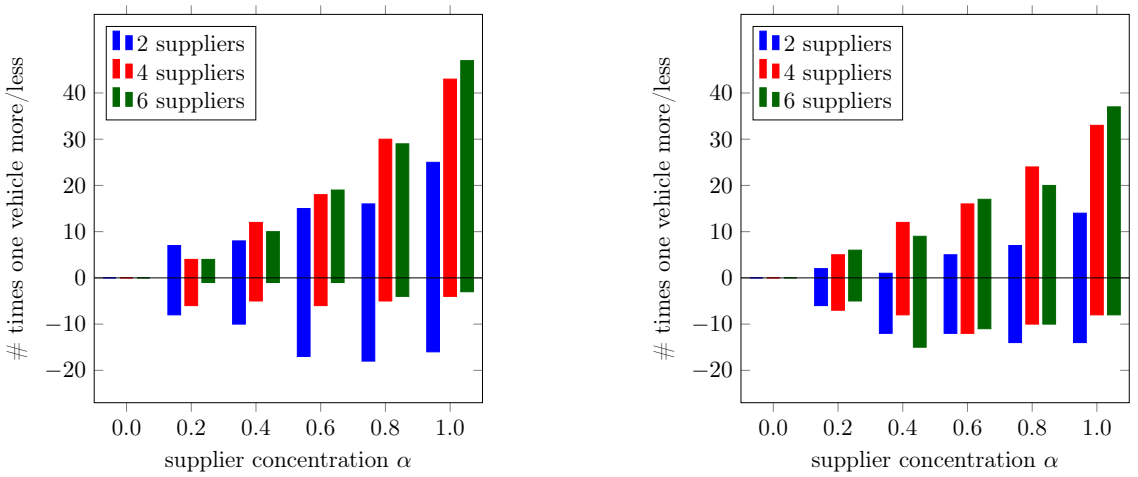
We created 3600 additional instances with 100 trips each that can be divided into two classes of 1800 instances each.

First, we generate 100 *base instances* for each class and each $|S| \in \{2, 4, 6\}$ in the same fashion as before except that the suppliers are placed more centrally in both classes, i.e., placed randomly at coordinates $(x_{S_i}, y_{S_i}) \in [15, 35] \times [15, 35]$, while customers coordinates (x_{C_i}, y_{C_i}) are taken randomly from $[1, 50] \times [1, 50]$, i.e., the suppliers' locations are somewhat optimized with respect to

their customers. Additionally, we distinguish the `Random` instance class, where trips are assigned randomly to a supplier, and the `Optimized` class, where trips are assigned to the nearest supplier (regarding Euclidean distances) with a higher probability of $0.5 + \frac{1}{|S|}$.

For all created base instances and all values $\alpha \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$, we build additional instances to model the clustering of the suppliers. These instances differ only in the supplier coordinates: For all $i \in S$, we compute new coordinates $(x_{S_i}, y_{S_i}) := (\lceil (1-\alpha)x_{S_i} + \alpha x_{CM} \rceil, \lceil (1-\alpha)y_{S_i} + \alpha y_{CM} \rceil)$, where CM is the center of mass of all customer locations, i.e., $CM := (x_{CM}, y_{CM}) = (\lfloor \sum_{i \in I} x_{C_i}/|I| \rfloor, \lfloor \sum_{i \in I} y_{C_i}/|I| \rfloor)$. As a result, $\alpha = 0$ reproduces the base instance, $0 < \alpha < 1$ increases the concentration of suppliers, and $\alpha = 1$ has all suppliers at the same location. The procedure of concentrating suppliers for a base instance with four suppliers is depicted in Figure 2.

All of the 3600 instances were solved to optimality in less than 50 seconds on average. Figure 3 shows how often all deliveries could be made with one vehicle less than in the baseline scenario ($\alpha = 0$), i.e., where all suppliers are located at their original (random) sites, and how often an extra vehicle was necessary to visit all customers compared to that scenario. Figure 4 visualizes the same information with regard to the weighted earliness objective.

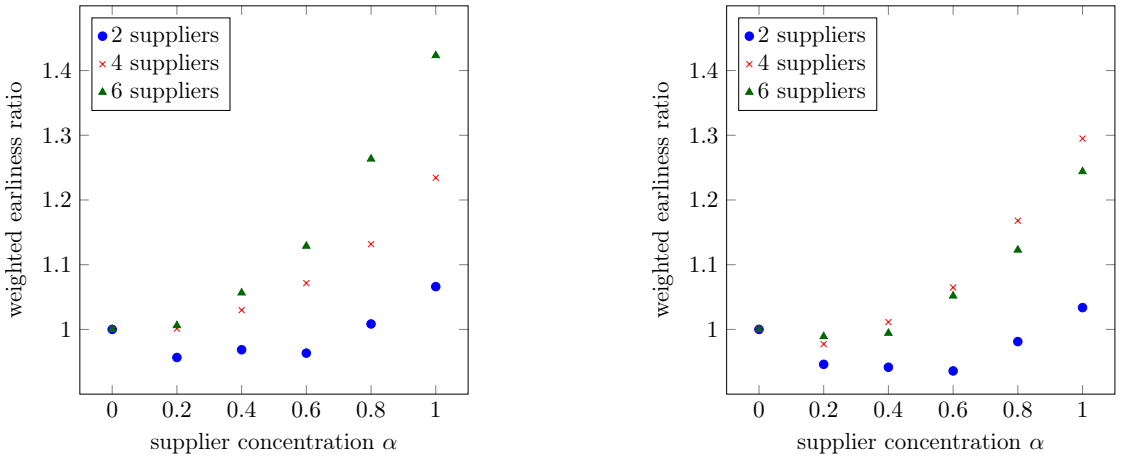


(a) Optimized Supplier Assignments

(b) Random Supplier Assignments

Figure 3: Difference in Number of Vehicles

While there is some expected random fluctuation, surprisingly, our tests reveal that supplier concentration negatively affects both fleet size and weighted earliness on average. The more centralized the suppliers are, the more the logistics performance suffers. The performance loss is particularly strong in case of optimized customer-supplier assignments (class `Optimized`), because the advantage of matching customers with nearby suppliers is lost if all suppliers are in the same spot anyway. However, even in the `Random` class, more trucks are required and weighted earliness rises when suppliers are grouped together. At first glance, this seems counterintuitive, since the average distance between suppliers and customers drops when siting suppliers at the center of mass of their customers. However, the return trips become longer in many cases. Since trucks do not need to return to the same supplier from whence they set off, after visiting a far-distant customer, a truck may be able to return to a nearby supplier for its next trip as long as suppliers are widely dispersed. If suppliers are clustered in the same location, trucks must always make a full round trip back to the same area from which they came. This leads to the somewhat surprising conclusion



(a) Optimized Supplier Assignments

(b) Random Supplier Assignments

Figure 4: Difference in Weighted Earliness

that supplier parks can actually be detrimental to JIT logistics performance when direct shipping policies are used.

6. Conclusion and Outlook

In this paper, we introduced the direct delivery scheduling problem in networks (DDSP-N), which is concerned with assigning and timetabling a given set of trips to vehicles such that the vehicle fleet is as small as possible and the customer waiting times are minimal. We adapted the compact model of Emde and Zehtabian (2017), presented a path-based formulation, and solved the latter via branch-cut-and-price. Our algorithm solved all instances from the literature in less than one minute on average. On newly generated problems, branch-cut-and-price solved to optimality most DDSP-N instances with up to 100 customers in acceptable time. For larger data sets, it found tight bounds within 1 hour of CPU time.

Our computational tests also revealed that, somewhat counterintuitively, the common just-in-time practice of gathering multiple suppliers in so-called supplier parks may not actually facilitate logistics operations in a direct shipping network. Of course, real-world just-in-time shipping does not only consist of direct deliveries; however, they do often make up a sizeable share of total deliveries. For instance, Klug (2010) reports that about 30% of parts are delivered directly via truck to the BMW plant in Leipzig (Germany). Given such figures, it is worth investigating whether supplier centralization is necessarily as helpful as is widely believed.

As point-to-point shipping is often used for high-velocity goods, where even slight delays can be problematic, future research should study the robustness of direct delivery networks by incorporating uncertain travel times into a stochastic model. Our model considers this only implicitly by minimizing waiting times. Moreover, industrial distribution networks often contain a mix of direct deliveries and other shipping strategies. Optimizing delivery policy and route choices in a mixed network is a worthwhile topic for future work.

References

Achterberg, T. (2007). *Constraint Integer Programming.* Ph.D. thesis, Technische Universität Berlin, Fakultät II – Mathematik und Naturwissenschaften, Berlin, Germany.

Allahverdi, A., Ng, C., Cheng, T. E., and Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, **187**(3), 985–1032.

Angel-Bello, F., Cardona-Valdes, Y., and Alvarez, A. (2017). Mixed integer formulations for the multiple minimum latency problem. *Operational Research*. DOI: 10.1007/s12351-017-0299-4.

Balcik, B., Beamon, B. M., and Smilowitz, K. (2008). Last mile distribution in humanitarian relief. *Journal of Intelligent Transportation Systems*, **12**(2), 51–63.

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.

Barnes-Schuster, D. and Bassok, Y. (1997). Direct shipping and the dynamic single-depot/multi-retailer inventory system. *European Journal of Operational Research*, **101**(3), 509–518.

Boysen, N., Emde, S., Hoeck, M., and Kauderer, M. (2015). Part logistics in the automotive industry: Decision problems, literature review and research agenda. *European Journal of Operational Research*, **242**(1), 107–120.

Carpaneto, G., Dell'Amico, M., Fischetti, M., and Toth, P. (1989). A branch and bound algorithm for the multiple depot vehicle scheduling problem. *Networks*, **19**(5), 531–548.

Chen, L. (2008). *Product & Customer Profiling for Direct Store Delivery (DSD).* Master's thesis, Massachusetts Institute of Technology, Cambridge, MA.

Chopra, S. (2003). Designing the distribution network in a supply chain. *Transportation Research Part E: Logistics and Transportation Review*, **39**(2), 123–140.

Chopra, S. and Meindl, P. (2015). *Supply Chain Management: Strategy, Planning, and Operation.* Prentice Hall, Upper Saddle River, NJ, 6th edition.

Chuah, K. H. and Yingling, J. C. (2005). Routing for a just-in-time supply pickup and delivery system. *Transportation Science*, **39**(3), 328–339.

Dell'Amico, M., Fischetti, M., and Toth, P. (1993). Heuristic algorithms for the multiple depot vehicle scheduling problem. *Management Science*, **39**(1), 115–125.

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. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, chapter 3, pages 57–93. Kluwer Academic Publisher, Boston, Dordrecht, London.

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

Desaulniers, G., Madsen, O., and Ropke, S. (2014). The vehicle routing problem with time windows. In D. Vigo and P. Toth, editors, *Vehicle Routing*, chapter 5, pages 119–159. Society for Industrial and Applied Mathematics, Philadelphia, PA.

Du, T., Wang, F., and Lu, P.-Y. (2007). A real-time vehicle-dispatching system for consolidating milk runs. *Transportation Research Part E: Logistics and Transportation Review*, **43**(5), 565–577.

Emde, S. and Zehtabian, S. (2017). Scheduling direct deliveries with time windows to minimize truck fleet size and customer waiting times. Working paper, Technische Universität Darmstadt.

Fakcharoenphol, J., Harrelson, C., and Rao, S. (2007). The k-traveling repairmen problem. *ACM Transactions on Algorithms (TALG)*, **3**(4), 40.

Gallego, G. and Simchi-Levi, D. (1990). On the effectiveness of direct shipping strategy for the one-warehouse multi-retailer r-systems. *Management Science*, **36**(2), 240–243.

Gélinas, S., Desrochers, M., Desrosiers, J., and Solomon, M. (1995). A new branching strategy for time constrained routing problems with applications to backhauling. *Annals of Operations Research*, **61**, 91–109.

Graham, R. L., Lawler, E. L., Lenstra, J. K., and Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. In *Annals of Discrete Mathematics*, volume 5, pages 287–326. Elsevier, Amsterdam.

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

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*, chapter 2, pages 33–65. Springer, Berlin.

Jepsen, M., Petersen, B., Spoorendonk, S., and Pisinger, D. (2008). Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, **56**(2), 497–511.

Klug, F. (2010). *Logistikmanagement in der Automobilindustrie: Grundlagen der Logistik im Automobilbau*. Springer.
Knott, R. (1987). The logistics of bulk relief supplies. *Disasters*, **11**(2), 113–115.
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**(1), 101–116.
Larsson, A. (2002). The development and regional significance of the automotive industry: supplier parks in western europe. *International journal of urban and regional research*, **26**(4), 767–784.
Luo, Z., Qin, H., and Lim, A. (2014). Branch-and-price-and-cut for the multiple traveling repairman problem with distance constraints. *European Journal of Operational Research*, **234**(1), 49–60.
Meyer, A. and Amberg, B. (2017). Transport concept selection considering supplier milk runs–an integrated model and a case study from the automotive industry. *Transportation Research Part E: Logistics and Transportation Review*.
Nucamendi-Guillén, S., Martínez-Salazar, I., Angel-Bello, F., and Moreno-Vega, J. M. (2016). A mixed integer formulation and an efficient metaheuristic procedure for the k-travelling repairmen problem. *Journal of the Operational Research Society*, **67**(8), 1121–1134.
Park, J. and Kim, B.-I. (2010). The school bus routing problem: A review. *European Journal of Operational Research*, **202**(2), 311–319.
Pfohl, H.-C. and Gareis, K. (2005). Supplier parks in the German automotive industry: A critical comparison with similar concepts. *International Journal of Physical Distribution & Logistics Management*, **35**(5), 302–317.
Pinedo, M. L. (2016). *Scheduling: Theory, algorithms, and systems*. Springer, Berlin.
Rauch, P., Gronalt, M., and Häuslmayer, H. (2007). Überregionales Logistik- und Versorgungsnetzwerk für Holz-Biomasse. *Schriftenreihe Berichte aus Energie- und Umweltforschung*, **51**.
Ribeiro, C. C. and Soumis, F. (1994). A column generation approach to the multiple-depot vehicle scheduling problem. *Operations Research*, **42**(1), 41–52.
Ribeiro, G. M. and Laporte, G. (2012). An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, **39**(3), 728–735.
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**(3), 255–273.
Tilk, C., Rothenbächer, A.-K., Gschwind, T., and Irnich, S. (2017). 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. (2018). Branch-and-price-and-cut for the active-passive vehicle-routing problem. *Transportation Science*, **52**, 300–319.
Vigo, D. and Toth, P., editors (2014). *Vehicle Routing*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia, PA.
Yu, G. and Zhang, G. (2009). Scheduling with a minimum number of machines. *Operations Research Letters*, **37**(2), 97–101.