Branch-and-Price-and-Cut for the Truck-and-Trailer Routing Problem with Time Windows

Ann-Kathrin Rothenbächer,

Michael Drexl and Stefan Irnich

September 15, 2016

Discussion paper number 1617

Contact Details:

Ann-Kathrin Rothenbächer
Chair of Logistics Management
Gutenberg School of Management and Economics
Johannes Gutenberg Universität Mainz
Jakob-Welder-Weg 9
55128 Mainz, Germany

anrothen@uni-mainz.de

Michael Drexl
Chair of Logistics Management
Gutenberg School of Management and Economics
Johannes Gutenberg Universität Mainz
Jakob-Welder-Weg 9
55128 Mainz, Germany

drexl@uni-mainz.de

Stefan Irnich
Chair of Logistics Management
Gutenberg School of Management and Economics
Johannes Gutenberg Universität Mainz
Jakob-Welder-Weg 9
55128 Mainz, Germany

irnich@uni-mainz.de

Branch-and-Price-and-Cut for the Truck-and-Trailer Routing Problem with Time Windows

Ann-Kathrin Rothenbächer[*,a], Michael Drexl[a,b], Stefan Irnich[a]

[a] *Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University, Mainz D-55099, Germany.*
[b] *Faculty of Applied Natural Sciences and Industrial Engineering, Deggendorf Institute of Technology, Deggendorf D-94469, Germany.*

Abstract

In this paper, we present a new branch-and-price-and-cut algorithm to solve the truck-and-trailer routing problem with time windows (TTRPTW) and two real-world extensions. In all TTRPTW variants, the fleet consists of one or more trucks that may attach a trailer. Some customers are not accessible with a truck-and-trailer combination, but can however be serviced by one if the trailer is previously detached and parked at a suitable location. In the first extension, the planning horizon comprises two days and customers may be visited either on both days or only once, in which case twice the daily supply must be collected. The second extension incorporates load transfer times depending on the quantity moved from a truck to its trailer. The exact branch-and-price-and-cut algorithm for the standard variant and the two new extensions is based on a set-partitioning formulation in which columns are routes describing the movement of a truck and its associated trailer. Linear relaxations of this formulation are solved by column generation where new routes are generated with a dynamic programming labeling algorithm. The effectiveness of this pricing procedure can be attributed to the adaptation of techniques such as bidirectional labeling, the *ng*-neighbourhood, and heuristic pricing using dynamically reduced networks and relaxed dominance. The cutting component of the branch-and-price-and-cut adds violated subset-row inequalities to strengthen the linear relaxation. Computational studies show that our algorithm outperforms existing approaches on TTRP and TTRPTW benchmark instances used in the literature.

Key words: Vehicle Routing, Truck-and-Trailer Routing, Branch-and-Price-and-Cut

1. Introduction

We consider variants of the truck-and-trailer routing problem with time windows (TTRPTW) motivated by an application in raw milk collection in Bavaria. Vehicles of a dairy need to pick up milk from farms. Because of usually large distances between dairy and farms and high milk production rates, trucks with attached trailers are used to increase the total vehicle capacity. However, some farms are not accessible by truck-and-trailer combinations (henceforth referred to as *complete vehicles*) because of small yards or restricted access roads. These customers can though be serviced by a complete vehicle if the trailer is previously detached and parked at a suitable location. Before going back to the depot, the truck must return to the parking location of its trailer and recouple it.

Accessibility imposes that customers can be partitioned into *truck customers* that are only accessible by trucks without attached trailer, and *trailer customers* that can be serviced by either a complete vehicle or a single truck. At trailer customers and at a set of optional *transshipment locations*, a trailer can be decoupled from its truck and parked. The truck can then visit truck and/or trailer customers without the trailer. This

*Corresponding author.
Email address: `anrothen@uni-mainz.de` (Ann-Kathrin Rothenbächer)

is referred to as a *subtour*. A truck can terminate a subtour, i.e., return to its parked trailer, at any time. It can then transfer load to the trailer. After a subtour, the truck can start another one or re-couple the trailer and pull it away. We assume that collected supply is directly loaded into the trailer if the trailer is attached and has residual capacity. At truck customers and whenever supply exceeds the trailer's residual capacity, supply goes directly into the truck.

Given a vehicle fleet and a set of customers, the task is to find a cost-minimal set of routes to collect the whole supply of all customers while respecting vehicle capacities, time window constraints, and accessibility restrictions. The vehicle fleet is heterogeneous, containing several types of trucks and trailers with different capacities. Trucks can either perform a route alone, or they can pull a single, uniquely assigned trailer, which can be detached temporarily. Optimization includes deciding whether to use a given truck as a single vehicle or as the motorized part of a complete vehicle.

An example of the TTRP (for simplicity without time windows and load amounts) with a possible solution is depicted in Figure 1. The fleet consists of two trucks and two trailers that can be combined as indicated in the figure. All vehicles are based at a central depot where all routes start and end. There are four truck and four trailer customers to serve. Additionally, two transshipment locations are offered whose visit is optional. The depicted solution comprises two routes: the orange tour is performed by truck 1, and the light blue/purple route, which contains two subtours, is executed by a complete vehicle consisting of truck 2 and trailer 2. Trailer 1 and transshipment point number 10 are not used.

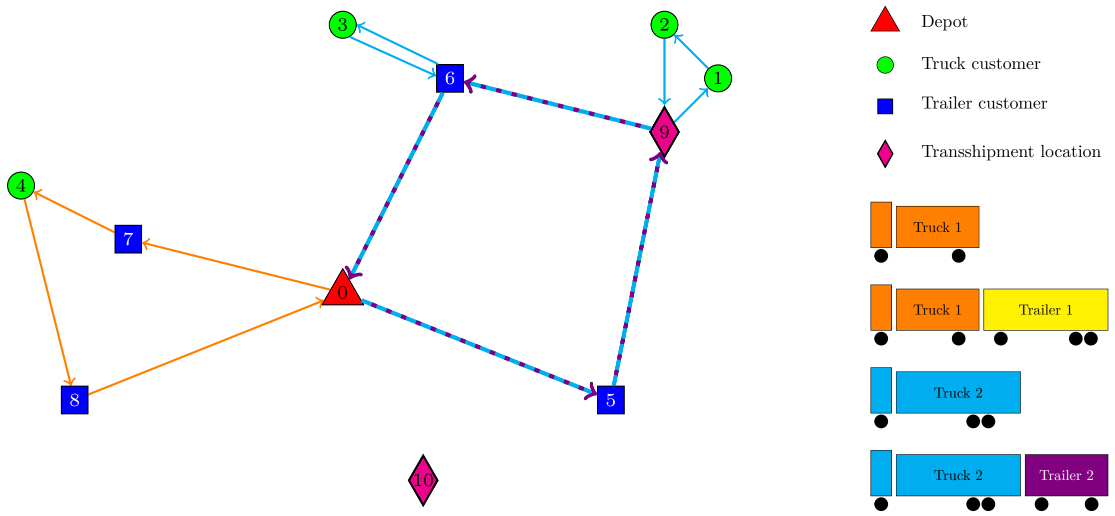


Figure 1: Example of the TTRP

The two challenging extensions that we address are of high practical relevance and, to the best of our knowledge, they have not yet been discussed in the literature. First, an important trend in the dairy business is *two-day collection*: many modern farms dispose of sufficient storage capacity to keep their milk for at least one day before it must be collected. Such customers can then be visited either daily or only every other day. The decisions to take in the resulting two-day planning problem include the determination of those customers that are serviced only once by collecting their entire two-day supply, and, if so, on which day the visit occurs (day 1 or day 2). The problem is perfectly symmetric with respect to the two days. We exploit this symmetry by applying a tailored column-generation stabilization method, which allows to halve the number of pricing problems to be solved, at least before day-specific branching is performed.

Second, an aspect up to now neglected in the TTRP literature is that in practice, the time for transferring load from truck to trailer often depends on the transferred quantity. Notably, this is the case in milk collection. The literature either assumes an immediate transfer or a fixed positive time independent of the quantity. Consequently, our second extension is the introduction of a quantity-dependent load transfer time, which imposes a tradeoff between the duration of a transfer operation and the resulting free capacity in the truck. Such a tradeoff heavily complicates the labeling algorithm.

Overall, our contribution is the presentation of effective branch-and-price-and-cut algorithms for the TTRPTW and the two extensions. For the classical TTRPTW, our new algorithm outperforms alternative exact approaches from the literature. The success of the new approach can be attributed to a new bidirectional labeling algorithm for the pricing subproblem and the use of the subset-row cuts to strengthen the linear relaxation of the master problem (see Jepsen *et al.*, 2008).

The remainder of this paper is organized as follows. A review of the relevant literature is provided in the next section. In Section 3, our branch-and-price-and-cut algorithm is explained with emphasis on the solution of the pricing subproblem and branching strategy. Section 4 establishes the extension of the planning horizon from one to two days. Section 5 presents the necessary adaptations when a positive load transfer time is assumed. Computational results are discussed in Section 6, and conclusions are drawn in Section 7.

2. Literature Review

Although the consideration of trailers in the literature on vehicle routing problems (VRPs) dates back several decades, the TTRP has attracted increased attention of the research community only in the last few years. This is surprising, given the large number of practical applications described in the existing publications: in addition to the already mentioned raw milk collection, TTRP applications include the distribution of goods to grocery stores (Semet and Taillard, 1993), of finished dairy products to customers (Gerdessen, 1996), of compound animal food to farmers (Gerdessen, 1996), postal mail delivery (Bodin and Levy, 2000), the movement of empty and loaded containers for a logistics company (Tan *et al.*, 2006), and fuel oil delivery to private households (Drexl, 2011).

The term "truck-and-trailer routing problem" was coined by Chao (2002). Since then, numerous advanced metaheuristics for different TTRP variants were presented by, e.g., Scheuerer (2006), Villegas *et al.* (2010, 2011, 2013), Caramia and Guerriero (2010a,b), Lin *et al.* (2011), Derigs *et al.* (2013), Pasha *et al.* (2014), and Batsyn and Ponomarenko (2014). All of these papers deal with deterministic problem settings. By contrast, Torres *et al.* (2015) consider a TTRP with fuzzy constraints, and Mirmohammadsadeghi and Ahmed (2015) handle stochastic demands. Only Lin *et al.* (2011) and Derigs *et al.* (2013) take into account time windows. The paper by Cuda *et al.* (2015) contains a detailed overview of the TTRP literature, discussing modeling aspects and details of solution approaches.

Exact procedures for TTRPs are still scarce. We are aware of only three works on this topic: Belenguer *et al.* (2016) propose a branch-and-cut algorithm for a single-vehicle variant of the TTRP without time windows. On a test bed with 32 random Euclidean instances, the largest instance solved to optimality has 100 customers and 10 transshipment locations. To the best of our knowledge, the only published exact algorithms tackling a multi-vehicle TTRP are the branch-and-price algorithms by Drexl (2011) and Parragh and Cordeau (2015). A comparison of these two papers with the present work, concerning the problem aspects considered and the solution methods used, is shown in Table 1.

3. Branch-and-Price-and-Cut

We start with a partial formalization of the problem and present the path-based formulation of the TTRPTW in Section 3.1. The path-based formulation is solved with a branch-and-price-and-cut algorithm. The column-generation subproblem and its resolution by a dynamic-programming labeling algorithm are discussed in detail in Section 3.2. Section 3.3 describes the branching strategy, Section 3.4 the integration of subset-row cuts, and Section 3.5 acceleration techniques.

3.1. Path-Based Formulation

Let N be the set of customers, L be the set of truck types, and H be the set of trailer types. The set of vehicle classes K contains both single trucks and certain combinations of a truck and a trailer and is defined as a subset of $L \times (H \cup \{0\})$. Elements $k = (l, 0)$ represent the single trucks of type $l \in L$. TTRPTW

	Our paper	Parragh and Cordeau (2015)	Drexl (2011)
Time windows	✓	✓	✓
Heterogeneous fleet	✓		✓
Dedicated transshipment locations	✓		✓
Two-day planning horizon	✓		
Quantity-dependent load transfer times	✓		
Heuristic and exact labeling algorithm	✓	✓	✓
Bidirectional labeling	✓		
Metaheuristic for pricing		✓	
ng-route relaxation	✓	✓	
Valid inequalities	✓		
Stabilization	✓	✓	

Table 1: Comparison of branch-and-price algorithms for different TTRP variants

instances may assume a limited fleet so that the number of available vehicles of type $f \in L \cup H$ is constrained by n_f. For convenience, we define the vehicle indicator b_f^k to be 1 if vehicle class $k \in K$ uses vehicle type $f \in L \cup H$, and 0 otherwise.

The path-based formulation is a set-partitioning model with one variable for each possible TTRPTW route describing the movement of a truck and, if present, its associated trailer. For a vehicle of class k, the set of all valid routes is R^k. Each route $r \in R^k$ has a corresponding decision variable λ_r^k, which equals 1 if the route is chosen, and 0 otherwise. Depending on the type of TTRPTW instance, route costs c_r can include variable routing costs, depending on traveled distance and on whether the truck is moving alone or towing the trailer, and transshipment costs. Transshipment costs occur whenever trailers are coupled or decoupled, or load is transferred from truck to trailer. We define a_{rn} as the number of times route r visits customer n.

The path-based model is:

$$\min \quad \sum_{k \in K} \sum_{r \in R^k} c_r \lambda_r^k \tag{1a}$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{r \in R^k} a_{rn} \lambda_r^k = 1 \qquad (\pi_n) \qquad \forall\, n \in N \tag{1b}$$

$$\sum_{k \in K} \sum_{r \in R^k} b_f^k \lambda_r^k \le n_f \qquad (\mu_f) \qquad \forall\, f \in L \cup H \tag{1c}$$

$$\lambda_r^k \in \{0,1\} \qquad \forall\, k \in K, r \in R^k \tag{1d}$$

The objective function (1a) aims at minimizing the total transportation costs. Then, the partitioning constraints (1b) ensure that every customer is visited exactly once. Constraints (1c) guarantee that the fleet size is not exceeded by the selected routes. Finally, the variable domains are given by (1d).

3.2. Column Generation

Model (1) cannot be solved directly due to the huge number of route variables. Instead we can solve it with a branch-and-price algorithm which uses a column-generation subproblem, the pricing problem, to dynamically generate route variables as they are needed (Lübbecke and Desrosiers, 2005). Similar to many other vehicle-routing problems, the pricing problems of the TTRPTW are instances of a shortest-path problem with resource constraints (SPPRC) on graphs with negative cost cycles (Irnich and Desaulniers, 2005). For each vehicle class $k \in K$, there is a specific type of SPPRC that has to be solved. Here, the goal is to find at least one route with negative reduced costs or to prove that no such route exists.

We start with the description of the underlying pricing network in Section 3.2.1. The resources and their resource extension functions are introduced in Section 3.2.2, while the TTRPTW-specific dominance is explained in Section 3.2.3. Details concerning bidirectional labeling are discussed in Section 3.2.4.

3.2.1. Network

The vehicle-class specific pricing network $D^k = (V, A^k)$ for vehicle class $k \in K$ is constructed as in (Drexl, 2011) so that each location is associated with one or several vertices. The four types of relevant locations are the depot, truck customers, trailer customers, and dedicated transshipment locations. The depot is modeled by a start vertex d^+ and a destination vertex d^-. Let $N^L \subset N$ be the set of truck customers and let $N^H \subset N$ be the set of trailer customers. Each truck customer is represented by a single vertex $n \in N^L$, while trailer customers are represented by four vertices, i.e., n for the service, $d(n)$ for detaching the trailer, $\tau(n)$ for load transfer, and $c(n)$ for coupling the trailer at the customer's location. Similarly, let T be the set of optional transshipment locations. There are three vertices for each transshipment location $t \in T$, namely, $d(t)$ for detaching the trailer, $\tau(t)$ for load transfer, and $c(t)$ for coupling the trailer at the transshipment location. Since both trailer customers and transshipment locations offer the parking and load transfer possibility, we define the set of *parking locations* $P = N^H \cup T$. Summarizing, the vertices V comprise the depots $\{d^+, d^-\}$, vertices $N^L \cup N^H$ for service at customers, decoupling vertices $D = \{d(p) : p \in P\}$, transfer vertices $\mathcal{T} = \{\tau(p) : p \in P\}$, and coupling vertices $C = \{c(p) : p \in P\}$. Table 2(a) lists the vertices for the network D^k.

Location	Associated vertices	Index set
Depot	d^+, d^-	
Truck customers	n	$n \in N^L$
Trailer customers	$n, d(n), \tau(n), c(n)$	$n \in N^H$
Transfer locations	$d(t), \tau(t), c(t)$	$t \in T$

(a) Vertices $i \in V$

	Arcs	to $j \in$					
	(i, j)	N^L	N^H	D	$\mathcal{T}$	C	d^-
	d^+	ST	✓	CV			
	N^L	✓	✓		CV	CV	ST
from	N^H	✓	✓	CV	CV	CV	✓
$i \in$	D	CV	CV				
	$\mathcal{T}$	CV	CV				
	C		CV	CV			CV

(b) Arcs $(i, j) \in A^k$ depend on the vehicle class k (CV=complete vehicle only, ST=single truck only, and ✓=both)

Table 2: Network $D^k = (V, A^k)$.

The type of arcs present in the pricing network depends on the vehicle class k. For single trucks $k = (l, 0)$, the arc set A^k contains only connections between the customer vertices and from/to the depot. Table 2(b) depicts these arcs with the symbols ST (single truck only) and ✓. For complete vehicles $k = (l, h)$ with $h \neq 0$, there exist many more possible connections. They are marked in Table 2(b) with the symbols CV (complete vehicle only) and ✓.

Decoupling vertices can be reached only with the trailer attached; transfer and coupling vertices can be reached only if the trailer is waiting at the corresponding location. Decoupling and transfer vertices can be left only by the truck; coupling vertices can be left only with the trailer attached. In other words, a trailer parked at a parking location implicitly 'moves' from the corresponding decoupling vertex to the transfer vertex (may be skipped) and from there to the coupling vertex. After parking a trailer at a location and before attaching it again, the truck may perform an arbitrary number of subtours and visits of the corresponding transfer vertex. Visiting a transfer or a coupling vertex includes the possibility of load transfer at zero cost and time. By contrast, when reaching a decoupling vertex, a transfer is not needed, because, as mentioned in the Introduction, the supply collected since the last visit to a transshipment location is loaded in the trailer as far as possible. This logic is guaranteed by the network structure and the feasibility checks of the resource extension functions (REFs) described in the following.

3.2.2. Labeling Algorithm

We start with the description of the cost of a TTRPTW route for a fixed vehicle $k = (l, h) \in K$ and introduce all relevant parameters needed to describe the resources and their consumption. The costs of an arc $(i, j) \in A^k$ consist of driving costs and handling costs occurring when a certain operation (service, decoupling, transfer, or coupling) is performed at a location. Driving costs can depend on the specific vehicle class k and on whether the trailer is towed. We incorporate the handling costs associated with the tail vertex i into the costs of (i, j). Handling costs can be location-specific, e.g., service costs can depend on the supply that a customer provides. In any case, the cost of arc $(i,j) \in A^k$ for $k=(l,h)$ can be expressed as

$$c^k_{ij}(\delta) = c^l_{ij} + \delta \cdot c^h_{ij},$$

where c^l_{ij} are truck-dependent and c^h_{ij} are trailer-dependent costs, and $\delta \in \{0,1\}$ indicates whether trailer h is actually towed from i to j. Similarly, the time t_{ij} consumed by traveling along arc (i,j) comprises the driving time and the handling time at the vertex i. We assume that t_{ij} does not depend on the vehicle k (but this could easily be modeled, too). While the service time may depend on the customer's supply, the coupling, the transfer, and the decoupling operations each consume a fixed time independent of the location. In Section 5 we will relax the constant-time assumption for load transfer.

For any feasible route r in $D^k = (V, A^k)$ from the start d^+ to the destination d^-, the sequence of vertices uniquely determines when and where the trailer is attached to the truck. Hence, we can define an indicator $\delta^H_{r,ij} \in \{0,1\}$ being equal to 1 if the trailer h is attached to the truck l when traveling along arc $(i,j) \in A^k$. Moreover, let the set of customer vertices and arcs covered by r be defined as $N(r)$ and $A(r)$ respectively. Let (π_n) be the dual prices associated with the partitioning constraints (1b) and (μ_f) be the dual prices associated with the fleet-size constraints (1c). Now, the reduced costs of a route r performed by a vehicle $k=(l,h)$ can be expressed as

$$\tilde{c}^k_r = \sum_{(i,j)\in A(r)} c^k_{ij}(\delta^H_{r,ij}) - \sum_{n\in N(r)} \pi_n - \sum_{f\in F} b^k_f \mu_f = \sum_{(i,j)\in A(r)} \tilde{c}^k_{ij}(\delta^H_{r,ij}),$$

where we define

$$\tilde{c}^k_{ij}(\delta) = c^k_{ij}(\delta) - \begin{cases} \pi_n, & j \in N \\ 0, & \text{otherwise} \end{cases} - \begin{cases} \mu_l + \mu_h, & i = d^+ \\ 0, & \text{otherwise} \end{cases}.$$

A TTRPTW route is time-feasible, if service at a visited customer n starts within its service time window $[e_n, \ell_n]$. The time window $[e_{d^+}, \ell_{d^+}] = [e_{d^-}, \ell_{d^-}]$ models the planning horizon. It is assumed that all other vertices $v \in D \cup \mathcal{T} \cup C$ have a non-restrictive time window. This means that trailer customers may have a restricted service time window, but that their location can be used for parking and transshipment at any time.

Each customer $n \in N$ has a positive supply q_n. The capacity of a vehicle $k=(l,h)$ is given by its truck and its trailer capacity Q^k_L and Q^k_H respectively. Single trucks, i.e., vehicles with $h=0$, have $Q^k_H = 0$. A TTRPTW route is load-feasible, if at no point neither truck nor trailer carry more collected supply than their given capacity.

Our labeling approach for the the classical TTRPTW uses six types of resources to ensure feasibility of the route of a complete vehicle. A partial path r from d^+ to a vertex $i \in V$ is represented by a label $E_i = (E^{cost}_i, E^{time}_i, E^{loadL}_i, E^{loadH}_i, E^{posH}_i, (E^{cust_n}_i)_{n\in N})$, where the label components are:

E^{cost}_i:	reduced cost of path r;
E^{time}_i:	earliest service/operation start time at vertex i;
E^{loadL}_i:	collected supply loaded into the truck at vertex i;
E^{loadH}_i:	collected supply loaded into the trailer at vertex i;
E^{posH}_i:	current position $p \in P$ of the trailer if detached from the truck, otherwise $\perp$;
$E^{cust_n}_i$:	number of times that customer $n \in N$ is visited along path r. Also set to 1 if customer n is not visited but is unreachable from r. A customer n is unreachable if $E^{loadL}_i + E^{loadH}_i + q_n > Q^k_L + Q^k_H$ or $E^{time}_i + t_{in} > \ell_n$ in which case it cannot be part of any feasible extension of path r.

In the initial label at vertex d^+, all components are set to 0 except $E^{time}_i = e_{d^+}$ and $E^{posH}_i = \perp$. The extension of a label E_i along an arc $(i,j) \in A^k$ is performed using the following REFs:

$$E^{cost}_j = E^{cost}_i + \tilde{c}^k_{ij}(\delta_{E^{posH}_i,\perp}) \tag{2a}$$

$$E^{time}_j = \max\{e_j, E^{time}_i + t_{ij}\} \tag{2b}$$

$$E_j^{loadL} = \begin{cases} E_i^{loadL} + q_j, & j \in N \text{ and } E_i^{posH} \neq \perp \\ E_i^{loadL} + \min\{E_i^{loadH} + q_j - Q_H^k, 0\}, & j \in N^H \text{ and } E_i^{posH} = \perp \\ E_i^{loadL}, & j \in D \cup \{d^-\} \\ E_i^{loadL} - \min\{E_i^{loadL}, Q_H^k - E_i^{loadH}\}, & j \in \mathcal{T} \cup C \end{cases} \tag{2c}$$

$$E_j^{loadH} = \begin{cases} E_i^{loadH}, & (j \in N \text{ and } E_i^{posH} \neq \perp) \text{ or } j \in D \cup \{d^-\} \\ \min\{E_i^{loadH} + q_j, Q_H^k\}, & j \in N^H \text{ and } E_i^{posH} = \perp \\ E_i^{loadH} + \min\{E_i^{loadL}, Q_H^k - E_i^{loadH}\}, & j \in \mathcal{T} \cup C \end{cases} \tag{2d}$$

$$E_j^{posH} = \begin{cases} E_i^{posH}, & j \in N \cup \mathcal{T} \\ p, & j \in D \text{ where } j = d(p), p \in P \\ \perp, & j \in C \cup \{d^-\} \end{cases} \tag{2e}$$

$$E_j^{cust_n} = \begin{cases} E_i^{cust_n} + 1, & j \in N \text{ and } j = n \\ \max\{E_i^{cust_n}, U_n(E_j)\}, & \text{otherwise.} \end{cases} \tag{2f}$$

We briefly explain the non-trivial parts of the REFs. In (2a), the Kronecker delta $\delta_{E_i^{posH},\perp}$ indicates whether the trailer is currently attached, i.e., whether $E_i^{posH} = \perp$. In (2c) and (2d), we distribute the collected supply among the truck and the trailer. As the truck can reach customers that the trailer cannot, the best strategy is to keep the truck as free as possible. Thus, supply is always directly loaded into the trailer whenever it is attached using the residual capacity of the trailer. Furthermore, at each coupling and transfer point, as much load as possible is transferred from the truck to the trailer. As mentioned, no load transfer takes place at decoupling points. In (2e), the decoupling operation is modeled in the second case in which we record the physical location where the trailer is parked. This is either a trailer customer or a transshipment location. Conversely, the recoupling operation is modeled in the third case of (2e). Finally, in (2f), the visited and unreachable customers are updated, where $U_n(E_j) \in \{0, 1\}$ indicates whether customer n has become unreachable either due to the time E_j^{time} or the total load $E_j^{loadL} + E_j^{loadH}$.

The label E_j resulting from the extension along arc $(i, j) \in A^k$ is feasible if $E_j^{time} \leq \ell_j$, $E_j^{loadL} \leq Q_L^k$, $E_j^{loadH} \leq Q_H^k$, $E_j^{cust_n} \leq 1$ for all $n \in N$, and the trailer position allows the visit of vertex j. The latter is true if $j \in N^H$ or one of the following conditions is fulfilled:

$$\begin{aligned} E_i^{posH} &\neq \perp \quad \text{for} \quad j \in N^L \\ E_i^{posH} &= \perp \quad \text{for} \quad j \in D \\ E_i^{posH} &= p \quad \text{for} \quad j \in \mathcal{T} \cup C \text{ with } j = t(p) \text{ or } j = c(p) \end{aligned}$$

The last condition means that transfer and coupling vertices can only be reached if the trailer is not attached and waiting at this location.

For the routes of single trucks $k = (l, 0)$, the resources E_i^{loadH} and E_i^{posH} are not needed. The REFs for cost, time, truck load, and unreachable customers are defined as in the standard vehicle-routing problem with time windows (VRPTW).

Note that we consider locations of trailer customers as transshipment points without additional constraints. Any number of trailers may be parked there. Transfer is possible within and outside the customer's service time window. Moreover, parking the trailer and serving the customer are independent so that this customer may be served by another vehicle or by the vehicle under consideration but at another stop earlier or later in the route. Parragh and Cordeau (2015), however, require that locations of trailer customers are only used as parking places if their supply is collected during the stop (called *strict parking rule* in the following). For this setting, we introduce an additional binary label component $E_i^{collPosH}$ indicating whether the trailer is currently parked (in this case, it is at the parking location E_i^{posH}) and the supply of that customer still needs to be collected. Therefore, the attribute is set to 1 whenever a decoupling vertex $d(n)$ of a customer n is reached from a vertex $v \neq n$. Its value is only changed to 0 if the supply vertex stored in E_i^{posH} is visited.

Initially, $E_{d+}^{collPosH}$ is set to 0 and the REF for any arc $(i,j) \in A^k$ is

$$E_j^{collPosH} = \begin{cases} 1, & j = d(n) \in D, n \in N, \text{ and } i \neq n \\ 0, & j \in N \text{ and } j = E_i^{posH} \\ E_i^{collPosH}, & \text{otherwise} \end{cases}.$$

For testing the feasibility of the resulting label E_j after the extension along arc (i,j), we must check the additional condition

$$E_i^{collPosH} = 0 \quad \text{or} \quad j \in N^H \text{ with } c(j) = i$$

whenever the partial path is extended from a coupling vertex $i \in C$.

3.2.3. Dominance

Labeling algorithms use dominance to discard labels that cannot lead to an improved d^+-d^--path w.r.t. another label (or set of labels). A label E_j dominates another label E'_j associated to the same vertex j if all the following conditions are fulfilled:

$$E_j^{res} \leq E'^{res}_j \qquad res \in \{cost, time, loadL, cust_n (n \in N)\} \tag{4a}$$

$$E_j^{loadL} + E_j^{loadH} \leq E'^{loadL}_j + E'^{loadH}_j \tag{4b}$$

$$E_j^{posH} = E'^{posH}_j \tag{4c}$$

and

$$E_j^{collPosH} = E'^{collPosH}_j \tag{4d}$$

The last condition (4d) is only relevant in the case of the strict parking rule of Parragh and Cordeau (2015).

The above dominance can be strengthened using three additional arguments:

Different Trailer Positions $E_j^{posH} \neq E'^{posH}_j$. Condition (4c) can only be violated at a customer vertex j. Then, there is no clear preference between having the trailer attached or parked, because in the first case the truck does not need to return to the parking place, whereas in the second case truck customers can be approached immediately. Moreover, two labels with different current parking positions of the trailers (with $E_j^{posH}, E'^{posH}_j \neq \perp$) are not directly comparable, as no completion of one path is a feasible completion of the other path.

However, label E_j may still dominate E'_j if the first vehicle can be transferred into the same trailer status as the second without violating the dominance constraints (4). To check this, we hypothetically move the first vehicle in up to three steps: (1) If $E_j^{posH} \neq \perp$, the first truck must pick up its trailer at location E_j^{posH}. (2) If $E'^{posH}_j \neq \perp$, the first truck must park its trailer at location E'^{posH}_j. (3) The first truck must move back to the current customer j. This creates a detour imposing higher values in the cost and time resources of E_j, but if the resulting label still fulfills condition (4), label E'_j can be discarded. The detour back to vertex j conflicts with the elementarity constraints. However, if the triangle inequality holds, the above check is a sufficient condition to show that all extensions of E'_j are dominated.

Higher Truck Load $E_j^{loadL} \geq E'^{loadL}_j$. Fulfilling condition (4b) and at the same time $E_j^{loadL} > E'^{loadL}_j$ is only possible during a subtour, i.e., for $E_j^{posH} \neq \perp$. Again, label E_j may dominate E'_j fulfilling all dominance conditions except $E_j^{loadL} \leq E'^{loadL}_j$ if the first vehicle could perform a detour to its trailer position in order to transfer load from truck to trailer. If the resulting label has smaller or equal cost and time resources, it dominates the second label.

Different Values indicating Collection at the Trailer Position $E_j^{collPosH} < E_j'^{collPosH}$. A strengthened dominance rule can be formulated for the case that $E_j^{collPosH} = 0$ and $E_j'^{collPosH} = 1$ hold. Both vehicles have to return to the location of customer $n = E_j^{posH}$, but the first cannot collect the reward π_n anymore. The second, however, needs to serve the customer n, finally imposing higher costs and additional load and service time. It is also less flexible on the current subtour due to the customer's service time window $[e_n, \ell_n]$. The modified dominance rule for $E_j^{collPosH} = 0$ and $E_j'^{collPosH} = 1$ is:

$$E_j^{res} \leq E_j'^{res} \qquad res \in \{time, loadL\} \tag{5a}$$

$$E_j^{cost} \leq E_j'^{cost} - \pi_{E_j^{posH}} \tag{5b}$$

$$E_j^{cust_n} \leq E_j'^{cust_n} \qquad n \in N, n \neq E_j^{posH} \tag{5c}$$

$$E_j^{loadL} + E_j^{loadH} \leq E_j'^{loadL} + E_j'^{loadH} + q_{E_j^{posH}} \tag{5d}$$

$$E_j^{posH} = E_j'^{posH} \tag{5e}$$

3.2.4. Bidirectional Labeling

When solving hard SPPRC instances, the number of labels often increases strongly with the length of the generated partial paths. Bidirectional labeling has been successfully applied to mitigate this type of combinatorial explosion. Forward and backward labels are only extended up to a so-called half-way point that splits the domain of a monotone resource into two intervals, one for the forward and one for the backward labeling. For the TTRPTW, we use the time resource as the monotone resource and define the half-way point as the middle $t^{hwp} = (e_{d^+} + \ell_{d^-})/2$ of the planning horizon. After labeling in both directions has terminated, compatible forward and backward partial paths are merged into complete d^+-d^--paths. For details we refer to (Righini and Salani, 2006).

As the forward labeling for the TTRPTW is already non-trivial to describe, we would like to avoid giving a long-winded and complicated description of the backward case. Therefore, we rely on a formal argument and design the backward labeling algorithm for the TTRPTW completely symmetric to the forward labeling algorithm. To this end, we define a reversed SPPRC instance in which the underlying network $D^k = (V, A^k)$ is replaced by the inverse network (V, B^k) with $(i, j) \in B^k$ if and only if $(j, i) \in A$, all time windows $[e_i, \ell_i]$ replaced by $[-\ell_i, -e_i]$, and all other coefficient are kept. Moreover, start and destination vertices d^+ and d^- as well as the role of coupling and decoupling vertices are swapped. We claim that for every feasible route of the given SPPRC instance, the reversed route is feasible in the reversed SPPRC instance, and vice versa.

Note first that the statement is certainly true for single trucks $k = (l, 0)$ and routes of complete vehicles $k = (l, h)$ that collect less supply than the truck's capacity. In these cases, the reversed SPPRC is equivalent to what has been suggested for the VRPTW in (Righini and Salani, 2006) and several subsequent publications. However, if more supply is collected, our loading policy that transfers as much as possible from the truck to the trailer is not directly interchangable between forward and backward solutions because feasible amounts for one direction may be infeasible for the opposite direction. To prove our claim, we argue similar to Desaulniers *et al.* (2014) and consider a feasible forward route $r = (v_0, v_1, v_2, \ldots, v_m)$ for which the selected supply exceeds the truck's capacity Q_L^k. One can obviously define transfer quantities in such a way that the truck arrives fully loaded at the destination. Moreover, let the transfer quantities at the vertices be $(t_0, t_1, t_2, \ldots, t_m)$ (with $t_i = 0$ for all non-parking vertices i) and the supply loaded into the truck at the vertices be $(q_0^L, q_1^L, q_2^L, \ldots, q_m^L)$ (with $q_i^L = 0$ for all parking vertices i). Then, the feasibility of the route implies

$$Q_L^k \geq \sum_{j=0}^{i} q_j^L - \sum_{j=0}^{i-1} t_j \geq \sum_{j=0}^{i} q_j^L - \sum_{j=0}^{i} t_j \geq 0,$$

for all $i \in \{0, 1, \ldots, m\}$, meaning that before (left term) and after (right term) the visit at vertex v_i the current load in the truck is non-negative and does not exceed the truck capacity. The fact that the truck arrives at d^- fully loaded implies $Q_L^k = \sum_{j=0}^{m} q_j^L - \sum_{j=0}^{m} t_j$. By subtracting this term from the above inequalities and multiplying with -1, we get

$$0 \leq \sum_{j=i+1}^{m} q_j^L - \sum_{j=i}^{m} t_j \leq \sum_{j=i+1}^{m} q_j^L - \sum_{j=i+1}^{m} t_j \leq Q_L^k,$$

showing that the route with same the transfer quantities $(t_0, t_1, t_2, \dots, t_m)$ considered in the reversed direction is also feasible.

In order to simplify the merge procedure, we restrict ourselves to customer vertices as merge points and mimic a kind of merge over arcs. First, the restriction to customer vertices is feasible because any useful route contains at least one customer vertex. The only necessary modification is on the forward half-way test: it requires that forward labels E_i are extended along arcs $(i,j) \in A^k$ up to the first customer vertex $j \in N$ that fulfills $E_j^{time} > t^{hwp}$. Second, we merge forward labels E_i and backward labels E'_i associated with the same vertex $i \in N$. However, it is more convenient to consider the predecessor label $E'_j = pred(E'_i)$ instead of the backward label E'_i itself (this trick was first described by Tilk *et al.*, 2016). Now, the concatenation of the forward partial path given by E_i and the backward partial path given by E'_i with $E'_j = pred(E'_i)$ is a feasible d^+-d^--path if and only if

$$E_i^{time} + t_{ij} \leq -E'^{\,time}_j \tag{6a}$$

$$E_i^{loadL} + E'^{\,loadL}_j \leq Q_L^k \tag{6b}$$

$$E_i^{loadL} + E_i^{loadH} + E'^{\,loadL}_j + E'^{\,loadH}_j \leq Q_L^k + Q_H^k \tag{6c}$$

$$E_i^{posH} = E'^{posH}_i \tag{6d}$$

$$E_i^{cust_n} + E'^{\,cust_n}_j \leq 1 \qquad \forall\, n \in N \tag{6e}$$

$$E_i^{collPosH} \neq E'^{\,collPosH}_j. \tag{6f}$$

The resulting route has reduced costs $E_i^{cost} + \tilde{c}_{ij}(\delta_{E_i^{posH},\perp}) + E'^{\,cost}_j$. Note that condition (6a) uses the time resource $E'^{\,time}_j$ defined in the reversed SPPRC, which is negative when used in the original context. In (6d), both trailer-position related resources refer to the merge vertex i. Moreover, condition (6e) is only valid if in at least one direction the customer-service related resources $E_i^{cust_n}$ or $E'^{\,cust_n}_j$ respectively describe exactly the subset of customers that are served. If unreachable customers are included in both directions, condition (6e) is too strict and thus incorrect. Finally, condition (6f) is not applicable if the extended dominance related to the resource *collPosH* is used in both forward and backward direction.

3.3. Branching

Let $\tilde{\lambda}_r^k$ be the values of variables λ_r^k in a solution of the linear relaxation to model (1). Moreover, we define the flow value $\tilde{x}_{ij}^k = \sum_{r \in R} a_{r,ij} \tilde{\lambda}_r^k$ for all $k \in K$ and $(i,j) \in A^k$, where $a_{r,ij}$ is the number of times that route r traverses arc (i,j). Let the aggregated flow value of arc (i,j) be $\tilde{x}_{ij} = \sum_{k \in K} \tilde{x}_{ij}^k$. Note that for an arc (i,j) with at least one customer vertex as an endpoint the values $\tilde{x}_{ij}$ and $\tilde{x}_{ij}^k$ are binary in a feasible integer solution.

As long as there are fractional values $\tilde{\lambda}_r^k$, we apply the following four-level branching scheme. First, we branch on the overall number of vehicles. Second, we branch on the aggregated flow value for an arc (i,j). Third, we branch on the vehicle-specific flow value for an arc (i,j). In the two latter cases, we always select arcs with at least one customer vertex as an endpoint first, since the branching decision can be implemented by network modifications. If all arcs with fractional value have no customer vertices as endpoints, we branch on the integer flow value by adding an inequality to the master program.

Note that branching on arcs disables the extended dominance rules including fictional detours. The existence of forbidden arcs would require a check of all arcs in the detour and even all arcs from the last visited parking location vertex in this detour to all possible successors.

There can still exist fractional solutions with integer aggregated and vehicle-specific flows after applying the above branching rules. An example is depicted in Figure 2. Four routes performed with the same

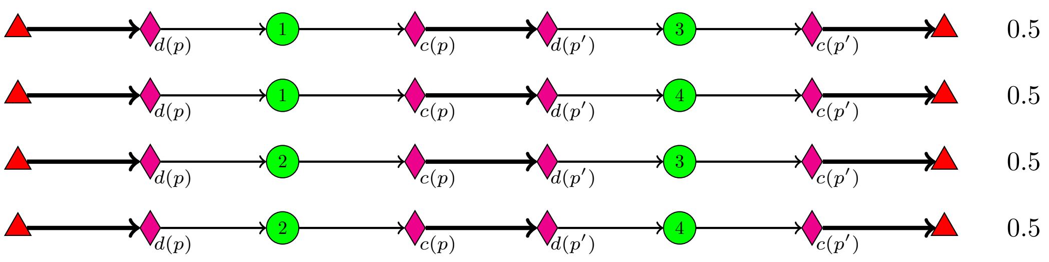


Figure 2: Example of four fractional paths of same vehicle class despite integral arc flows

vehicle class $k \in K$ each with a value $\tilde{\lambda}_r^k$ of 0.5 impose integer flows on the depicted arcs. The arcs $(d^+, d(p))$, $(c(p), d(p'))$, and $(c(p'), d^-)$ have a flow of 2, while all others have a flow of 1.

The rule for the fourth level is based on the following property: If for each customer the predecessor and successor customer (if any) are uniquely determined, a basic solution to the linear relaxation of (1) is integer. This is true because the predecessors and successors unambiguously define sets of customers served by the same vehicle. The branching decisions on the third level have already determined the vehicle class $k \in K$ that visits the customer set. All routes with a vehicle of class k serving the same set of customers have the same coefficients in the constraints of (1). Therefore, in a basic solution to the linear relaxation of (1) only one of the corresponding route variables can be positive so that one with the lowest cost is selected exactly once.

The fourth level of branching decisions finally guarantees integer path-variables in (1) without explicitly branching on the associated variables. If two customer vertices i and j are served consecutively, i before j, the two are either connected directly via arc (i, j) (handled by the rules above) or with a subpath Q of length 2 or 3, i.e., $Q = (i, v_1, j)$ or $Q = (i, v_1, v_2, j)$ where $v_1, v_2 \in C \cup \mathcal{T} \cup D$. We branch on such subpaths connecting two customers i and j. More precisely, to exclude a subpath Q in the network D^k this subpath is declared forbidden, and several such decisions are managed effectively by the network modification procedure of Villeneuve and Desaulniers (2005). To enforce a subpath Q in the network D^k, another type of network modification can be performed. One must create copies of the intermediate vertices v_1 and v_2 (if present), remove all arcs from the forward/backward star of i/j, and connect i with j via the copies. Such branching on sequences of customers has been applied in different contexts but with similar network modification procedures in (Desaulniers, 2010) and (Bode and Irnich, 2012).

3.4. Cuts

To strengthen the linear relaxation, we incorporate subset-row (SR) inequalities into model (1) (see Jepsen *et al.*, 2008). A valid SR inequality is defined on a subset of tasks. For the standard TTRPTW, tasks are the visits to the customers. For the TTRPTW with two-day planning horizon (see the later Section 4) tasks are the visits to the customers at a particular day, i.e., pairs of customer and day. We restrict ourselves to SR inequalities defined on three tasks as proposed by Jepsen *et al.* (2008) because they can be separated by straightforward enumeration. Given a set S_f of three tasks, the SR inequality $SR(S_f)$ is given by

$$\sum_{k \in K} \sum_{r \in R^k} \left\lfloor \frac{g_{fr}}{2} \right\rfloor \lambda_r^k \leq 1 \qquad (\sigma_f),$$

where g_{fr} is the number of times route r serves tasks in S_f.

The incorporation of SR inequalities into the master problem complicates the TTRPTW pricing problems: Let $\sigma_f \leq 0$ be the dual price of the SR inequality $SR(S_f)$. Then, σ_f has to be subtracted from the reduced costs of a route for every second service to tasks in S_f. In order to keep track of such services, one additional binary resource E^{SR_f} for each inequality $SR(S_f)$ must be defined. It indicates the parity of the number of times tasks in S_f are served.

Jepsen *et al.* (2008) suggested an improved dominance, in which otherwise incomparable labels can be considered. For two labels E_j and E'_j fulfilling the dominance conditions (4), the reduced-cost comparison in (4a) has to be replaced by the following generally tighter condition

$$E_j^{cost} - \sum_{f: E_j^{SR_f} > E_j'^{SR_f}} \sigma(S_f) \leq E_j'^{cost}.$$

In the merge step of the bidirectional labeling algorithm, the concatenation of the forward partial path given by E_i and the backward partial path given by E_i' with $E_j' = pred(E_i')$ has reduced costs of

$$E_i^{cost} + \tilde{c}_{ij}(\delta_{E_i^{posH},\perp}) + E_j'^{cost} - \sum_{f: E_i^{SR_f} + E_j'^{SR_f} = 2} \sigma(S_f).$$

3.5. Acceleration Techniques

In this section, we sketch some further techniques that we use to accelerate our algorithm. First, we start the algorithm with a valid lower bound for the number of vehicles necessary to serve all customers with respect to their supply. This bound is calculated by solving a bin-packing problem with a bin size equal to the maximal total vehicle capacity Q^{max} among all vehicle classes. If the run time exceeds one minute, the trivial lower bound $\lceil \sum_{n \in N} q_n / Q^{max} \rceil$ is used. Second, we relax the elementary requirement and use the ng-neighborhood relaxation presented by Baldacci *et al.* (2011) in order to accelerate the solution of the pricing problem. Third, in the first pricing iterations, we solve the pricing problem only for single-truck vehicle classes as long as new columns are generated. This significantly reduces the number of time-consuming pricings of routes of complete vehicles. Fourth, the pricing problems for complete vehicles are solved heuristically until no new columns are found. To this end, the dominance criterion is relaxed by ignoring conditions (4a) for $res = cust_n$, (4c), and (4d). Moreover, we use reduced networks by restricting, for each vertex, the number of outgoing arcs to both customer vertices and vertices of parking locations, where those with lower reduced costs are preferred. Fifth and finally, we apply the MIP solver of CPLEX for a maximum of one minute using the columns generated up to this point in order to increase the chance of finding good upper bounds. The MIP solver is called after solving the branch-and-bound root node and when reaching the time limit.

4. Two-Day Planning Horizon

Caused by the nowadays widespread use of larger storage tanks with improved cooling technology, many modern milk-producing farms no longer need to be visited every day. Instead, a visit every second day suffices, in which case the accumulated supply of two days must be collected. We model the case of a two-day planning horizon with two types of customers as follows. The first subset of customers (N^{every}) still needs a visit every day. The other customers (N^{option}) can be visited either every day for collection of their 'normal', daily supply, or every second day for collection of twice that amount.

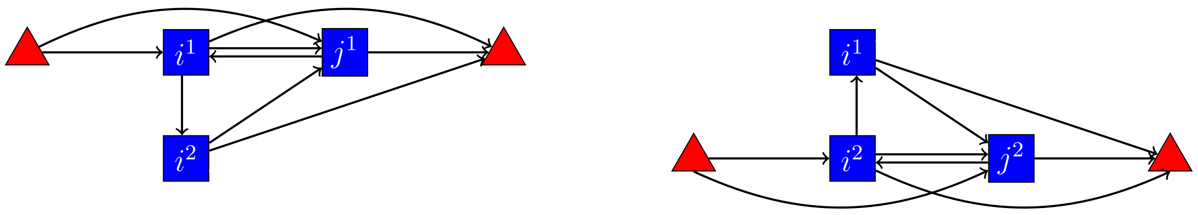
We solve the two-day problem by creating two tasks n^1, n^2 for each customer $n \in N$, representing the collection of the supply of the first and the second day respectively. The partitioning condition in the master problem (1b) is replaced by

$$\sum_{k \in K} \sum_{r \in R^k} a_{rn^\theta} \lambda_r^k = 1 \qquad (\pi_{n^\theta}) \qquad \forall\, n \in N, \theta \in \{1, 2\}, \tag{7}$$

where a_{rn^θ} indicates that route r covers task n^θ.

The distinction between the two types of customers is realized in the pricing networks. For each vehicle class $k \in K$ and each day $\theta \in \{1, 2\}$, there is one network for routes operated on that day. The network is structured as the network D^k described in Section 3.2.1 and contains depot, decoupling, transfer, and coupling vertices. In addition, each customer vertex is replaced by one or two task vertices. Precisely, customers $n \in N^{\text{every}}$ who need to be visited every day have one vertex n^θ for the task of day θ in the network, while customers $n \in N^{\text{option}}$ with service option have two vertices n^1 and n^2 for both tasks, see

Figure 3. Vertices for tasks of the same day are connected as in the standard TTRPTW network D^k. Moreover, each task vertex representing a customer $n \in N^{\text{option}}$ is connected to its corresponding task vertex of the other day. In addition, each such vertex can be left towards all vertices in the network of day θ that can be reached from customer n.



(a) Network for collection on the first day $\theta = 1$ (b) Network for collection on the second day $\theta = 2$

Figure 3: Example of the two pricing networks for a customer $i \in N^{\text{option}}$ (collection every other day allowed) and a customer $j \in N^{\text{every}}$ (must be visited every day)

Symmetry Considerations. A drawback of the two-day modeling and solution approach presented above is that the number of partitioning constraints (7) of the path-based formulation and the number of column-generation subproblems that need to be solved is doubled. We now show that the perfect symmetry with respect to the two planning days can be exploited so that the computational effort is approximately halved again.

We claim that for solving the linear relaxation of the two-day horizon master program, i.e., (1) with (1b) replaced by (7), nothing is lost with respect to the strength of the dual bound when partitioning constraints are aggregated. More precisely, every two constraints (7) for $\theta \in \{1, 2\}$ of a customer $n \in N$ can be replaced by aggregated constraints of the form

$$\sum_{k \in K} \sum_{r \in R^k} (a_{rn^1} + a_{rn^2}) \lambda_r^k = 2 \qquad \forall\, n \in N. \tag{8}$$

The proof relies on the concept of deep dual-optimal inequalities (DDOIs, Ben Amor *et al.*, 2006), originally introduced in order to stabilize column-generation algorithms: Pairwise aggregation of two constraints as suggested by (8) is equivalent to imposing constraints

$$\pi_{n^1} = \pi_{n^2} \qquad \forall\, n \in N,$$

to the dual formulation as shown by Gschwind and Irnich (2016, Proposition 7). This is again equivalent to the introduction of additional unconstrained variables η_n (=additional columns), one per customer, to the primal formulation. The new variable η_n has coefficient $+1$ in the partitioning constraint of the day-one-task and -1 in the constraint of the day-two-task so that the resulting partitioning constraints become

$$\sum_{k \in K} \sum_{r \in R^k} a_{rn^1} \lambda_r^k + \eta_n = 1 \qquad \forall\, n \in N \tag{9a}$$

$$\sum_{k \in K} \sum_{r \in R^k} a_{rn^2} \lambda_r^k - \eta_n = 1 \qquad \forall\, n \in N. \tag{9b}$$

Thus, the *extended primal formulation* has additional variables η_n and the task-partitioning constraints (7) are replaced by (9). Any feasible solution to the linear relaxation of the extended primal formulation can be transformed into a feasible solution to the linear relaxation of the original formulation having constraints (7) with identical costs. The procedure works as follows: Let $\tilde{\lambda}_r^k$ be a feasible solution to the extended primal formulation. Regardless of the values of η_n, define for each route $r \in R^k$ operated on day θ by vehicle $k \in K$ the corresponding route r' operated on the other day. Further define $\widehat{\lambda}_r^k = (\tilde{\lambda}_r^k + \tilde{\lambda}_{r'}^k)/2$ for all routes $r \in R^k$

and vehicles $k \in K$, which yields the equivalent feasible solution for formulation (1) with task-partitioning constraints (7). Now, Proposition 1 in (Gschwind and Irnich, 2016) ensures that under these conditions, the dual bound of the extended primal formulation and the original formulation coincide.

In summary, the use of the aggregated partitioning constraints (8) accompishes that the number of constraints is the same as in the one-day problem. Moreover, it induces identical subproblems for both days $\theta = 1$ and $\theta = 2$, so that only one pricing subproblem, for example for day $\theta = 1$, needs to be solved. Admittedly, this subproblem is larger than the one-day subproblem. However, a by-product of aggregation is that the dual variables are stabilized, leading to a generally faster termination of the column-generation procedure.

Branching and cutting can destroy the perfect symmetry. In order to maintain symmetry as long as possible, we always add symmetric pairs of SR inequalities. Hence, if $SR(S_f)$ is added for tasks S_f, the inequality $SR(S_{f'})$ for the other-day tasks $S_{f'}$ is also added. Both $SR(S_f)$ and $SR(S_{f'})$ are then aggregated, leading to a single inequality $\sum_{k\in K}\sum_{r\in R^k}(\lfloor g_{fr}/2\rfloor + \lfloor g_{f'r}/2\rfloor)\lambda_r^k \leq 2$.

While branching on the number of vehicles preserves symmetry, branching on individual arcs of the two day-specific networks (see Figure 3) requires that the column-generation master program be disaggregated. We do so by replacing the aggregated constraints (8) with the disaggregated ones (7), and by adding the other-day route variables $\lambda_{r'}^k$ for all active route variables λ_r^k.

Overnight Return of Trailers. As pointed out by Tricoire (2016), a two-day planning horizon allows further savings when the requirement that all vehicles return to the depot at the end of each day is relaxed and it is allowed to park an empty trailer overnight at a transshipment location. It depends on the application whether this can be done in practice. The consideration of the overnight parking option requires considerable modifications to our model and the solution approach described in this section. Moreover, we expect the additional savings to be much smaller than those obtained by switching from a one-day to a two-day horizon, so we leave this as a topic for further research.

5. Quantity-Dependent Transfer Time

Up to now, the TTRP literature either assumes an immediate transfer of load from truck to trailer or a fixed positive time independent of the quantity. In reality, however, the time for transferring load from truck to trailer increases with the transferred quantity. In this section, we discuss the non-trivial modifications that are needed to exactly model quantity-dependent load transfer times in the column-generation subproblem. The modified SPPRC in this case must consider the tradeoff between the visit time at parking locations and the free capacity in the truck. A similar issue arises in the context of ship routing and scheduling when port visits are time-constrained and the port stay times are influenced by the amount of cargo loaded or unloaded. Brønmo *et al.* (2007) and Hennig *et al.* (2012) study such problems and respectively apply column generation and branch-and-price. Brønmo *et al.* solve an LP to determine an optimal schedule and load quantities for given port visit sequences. Hennig *et al.* use nested branch-and-price: they solve the subproblem of determining a time- and load-feasible ship route (port visit sequence) by decomposing it into a master problem that computes schedules and load patterns and a subproblem for computing ship routes (a shortest path problem with time windows solved by dynamic programming). However, as far as we know, we herewith present the first labeling algorithm for this type of tradeoff, i.e., the first that handles the determination of route, schedule, and load transfer quantity simultaneously.

In the following, let $\rho > 0$ be the transfer rate for the quantity transferrable from the truck to the trailer during one time unit. Since usually the technical equipment of the vehicles determines the transfer rate, it can be assumed independent of the transfer location. Note that with quantity-dependent transfer times, the strategy to always transfer as much as possible from the truck to the trailer is no longer optimal. If more load than necessary is transferred, the additional delay may cause a time window of a customer visited later to be missed. Therefore, the load-time tradeoff needs to be taken into account explicitly.

Figure 4 depicts the route $(d^+, d(p), 1, \tau(p), 2, c(p), d^-)$ vising customers 1 and 2 and the decoupling vertex $d(p)$, the transfer vertex $\tau(p)$, and the coupling vertex $c(p)$ of a parking location p, where load can be transferred. Below each transshipment and customer vertex, a diagram of truck load vs. time is presented.

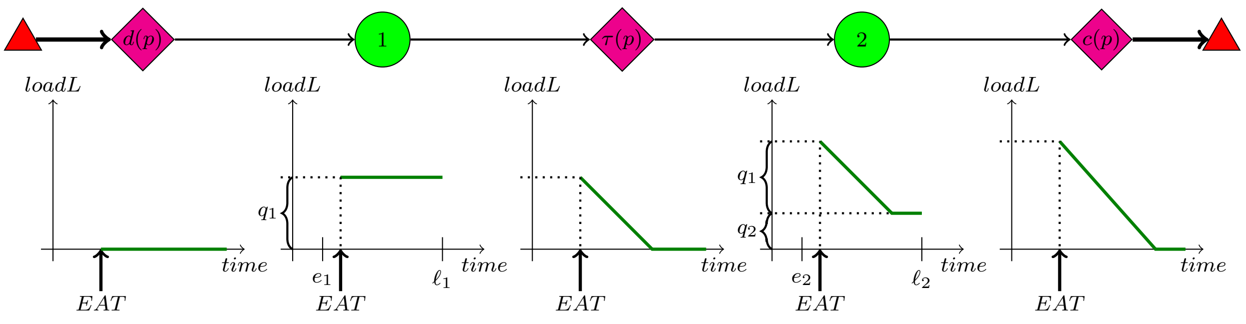
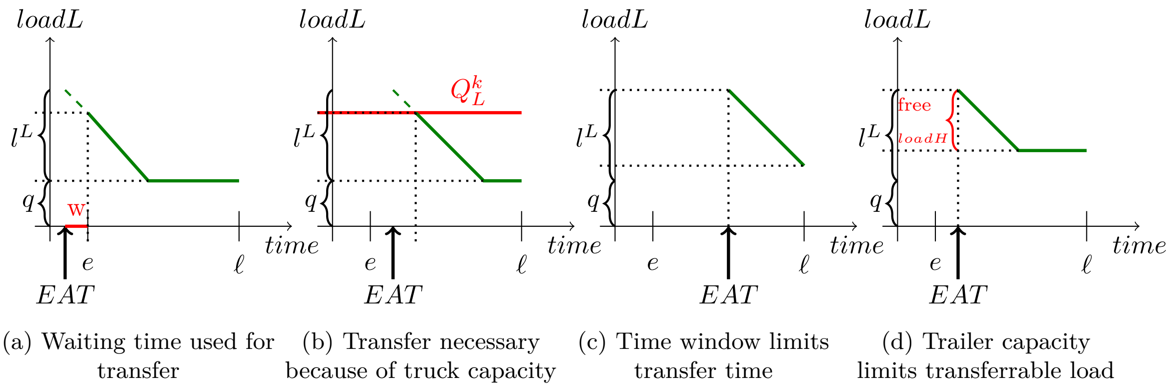


Figure 4: Example for tradeoff propagation with quantity-dependent transfer times

The time axis is marked with EAT representing the *earliest arrival time* at the vertex (this time can be before the service time window opens). Moreover, at customers 1 and 2, the time windows $[e_1, \ell_1]$ and $[e_2, \ell_2]$ are shown. We discuss the five tradeoff curves (depicted in green) for the vertices $d(p), 1, \tau(p), 2$, and $c(p)$. Each tradeoff curve displays to which extent a later start of service (starting not earlier than E_i^{time}) allows to reduce the load inside the truck (resource E_i^{loadL}). When the decoupling vertex $d(p)$ is reached, no load has been collected yet. Thus, no transfer possibility arises at the first transshipment point, reflected by the 0-function. During the first subtour to customer 1, load has to be collected to the truck so that still no tradeoff is visible. When the truck meets its trailer again at $\tau(p)$, the option is to either continue the tour immediately (with q_1 loaded in the truck) or to spend time to transfer load from truck to trailer. The non-zero slope of the tradeoff curve is $-\rho$. In our example, we assume that the truck as well as the trailer each have sufficient capacity to accommodate the complete supply of both customers, $q_1 + q_2$. Hence, the tradeoff curves in the diagrams below $\tau(p)$ and below customer 2 indicate that the supply of customer 1 can be partly or completely transferred to the trailer at $\tau(p)$. Likewise, the curve in the diagram below $c(p)$ shows that the entire customer supply can either remain in the truck or be transferred partly or completely.



(a) Waiting time used for transfer (b) Transfer necessary because of truck capacity (c) Time window limits transfer time (d) Trailer capacity limits transferrable load

Figure 5: Special cases for arrival at customer concerning the tradeoff

In general, a non-zero time-load tradeoff can be consumed or reduced. Figure 5 illustrates four cases when this occurs. All variants depict a possible situation for the visit of a customer with service time window $[e, \ell]$, in which l^L refers to the load collected to the truck before and q is the supply of the currently visited customer. First, if a customer is reached before the time window opens, all unavoidable waiting time is used for transfer, see Figure 5(a). Second, if the available capacity of the truck does not suffice to collect the complete supply, the surplus must be transferred lowering the disposable amount of load, see Figure 5(b). Third, if a customer is reached within its service time window then the remaining time limits the transfer time, see Figure 5(c). Fourth and finally, the available capacity of the trailer limits the transferrable amount, see Figure 5(d).

5.1. Time-Load Tradeoff, Resources, and their Propagation

The general form of the time-load tradeoff curve is shown in Figure 6. It can be defined by three parameters: the earliest possible starting time of the service at the vertex (E_i^{time}), the highest possible truck load at this time ($E_i^{\overline{loadL}}$), and the minimal achievable truck load ($E_i^{\underline{loadL}}$). The latter two values define two new resources and replace the resource E_i^{loadL} of the case with constant transfer times handled in Section 3.2.2. Accordingly, new resource extension functions for the resources E_i^{time}, $E_i^{\overline{loadL}}$, $E_i^{\underline{loadL}}$, and E_i^{loadH} must be defined whereas the propagation and feasibility checks of all other resources (E_i^{cost}, E_i^{posH}, $E_i^{cust_n}$, and $E_i^{collPosH}$) are not affected.

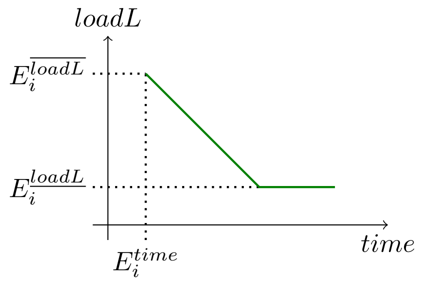


Figure 6: Shape of the tradeoff curve

Let $(i,j) \in A^k$ be any arc in the subnetwork of a complete vehicle $k \in K$. We define some auxiliary values useful for defining the new REFs for propagating the new resources along (i,j). First, the load that can be transferred in the waiting time ($LTWT$) depends on the unavoidable waiting time, which is the time difference between the earliest arrival time and the start of the time window, and it is also limited by the maximum potentially transferable load at previous transshipment locations. Hence, this value can be defined as

$$R_{ij}^{LTWT}(E_i) = \min\{\rho \cdot \max\{e_j - (E_i^{time} + t_{ij}), 0\}, E_i^{\overline{loadL}} - E_i^{\underline{loadL}}\}.$$

Second, in order to serve the following customer j, there may be some load that must be transferred to the trailer before ($NLTB$). This amount is

$$R_{ij}^{NLTB}(E_i) = \max\{E_i^{\overline{loadL}} + q_j - Q_L^k, 0\}.$$

Third, the minimal load that has to remain in the truck ($MinLL$) is restricted by the trailer's capacity and the time window of the current vertex i, i.e.,

$$R_{ij}^{MinLL}(E_i) = \max\{0, E_i^{\overline{loadL}} - (Q_H^k - E_i^{loadH}), E_i^{\overline{loadL}} - \rho \cdot (\ell_i - E_i^{time})\}.$$

The REFs for extending a label E_i along an arc (i,j) primarily depend on the type of the head vertex j. Therefore, we present the REFs by distinguishing the following three cases:
(1) the arrival at a customer without the trailer,
(2) the arrival at a customer with the trailer attached, and
(3) the arrival at a coupling, transfer, decoupling, or the depot vertex d^-.
For convenience, we also present all REFs for the inclusion of quantity-dependent transfer times in the standard form, i.e., grouped by resources, in the Appendix, Section I.

Arrival at Customer without Trailer. All load has to be collected to the truck, but sometimes the truck load can ($LTWT$) or must ($NLTB$) be reduced.

$$E_j^{time} = \max\{e_j,\ E_i^{time} + t_{ij} + R_{ij}^{NLTB}(E_i)/\rho\}$$
$$E_j^{\overline{loadL}} = E_i^{\overline{loadL}} + q_j - \max\{R_{ij}^{NLTB}(E_i),\ R_{ij}^{LTWT}(E_i)\}$$

$$
\begin{aligned}
E_j^{loadH} &= E_i^{loadH} + \max\{R_{ij}^{NLTB}(E_i),\ R_{ij}^{LTWT}(E_i)\} \\
E_j^{\underline{loadL}} &= \max\{E_i^{\underline{loadL}} + q_j, R_{ij}^{MinLL}(E_i)\}
\end{aligned}
$$

The label E_j resulting from the extension along arc $(i,j) \in A^k$ is feasible if $E_j^{time} \leq \ell_j$, $E_j^{loadH} \leq Q_H^k$, and $R_{ij}^{NLTB} \leq E_i^{\overline{loadL}} - E_i^{\underline{loadL}}$ holds in addition to the conditions for the remaining resources (see Section 3.2.2).

Arrival at Customer with Trailer attached. The supply is collected as much as possible directly to the trailer to keep the truck flexible.

$$
\begin{aligned}
E_j^{time} &= \max\{e_j,\ E_i^{time} + t_{ij}\} \\
E_j^{\overline{loadL}} &= E_i^{\overline{loadL}} - \min\{R_{ij}^{LTWT}(E_i), Q_H^k - E_i^{loadH} - q_j\} \\
E_j^{loadH} &= E_i^{loadH} + q_j + \min\{R_{ij}^{LTWT}(E_i), Q_H^k - E_i^{loadH} - q_j\} \\
E_j^{\underline{loadL}} &= \max\{E_i^{\underline{loadL}} + \max\{E_i^{loadH} + q_j - Q_H^k, 0\}, R_{ij}^{MinLL}(E_i)\}
\end{aligned}
$$

Similarly to the first case, the label E_j is feasible w.r.t. these resources if $E_j^{time} \leq \ell_j$ and the minimal truck load does not exceed the truck capacity, i.e., $E_j^{\underline{loadL}} \leq Q_L^k$.

Arrival at Coupling, Transfer, Decoupling, or Depot. No change in the loads is realized, only the potential transfer is limited by the free capacity in the trailer.

$$
\begin{aligned}
E_j^{time} &= E_i^{time} + t_{ij} \\
E_j^{\overline{loadL}} &= E_i^{\overline{loadL}} \\
E_j^{loadH} &= E_i^{loadH} \\
E_j^{\underline{loadL}} &= R_{ij}^{MinLL}(E_i)
\end{aligned}
$$

The only feasibility condition for these resources is $E_j^{time} \leq \ell_j$ when arriving at the depot $j = d^-$.

5.2. Dominance

The original dominance has to be extended to model the tradeoff correctly. A label E_j is better than another label E'_j belonging to the same vertex j, if the dominance criteria (4) concerning the unchanged resources are fulfilled and if furthermore the tradeoff curve of E_j lies completely left below the curve of E'_j. This means that, at every feasible service start time of label E'_j, the truck load of E_j is less than or equal to the truck load of E'_j. Because the slope of the decreasing part of all tradeoff curves is equal, this can be expressed by requiring a smaller or equal truck load of label E_j at the starting time of E'_j and a not greater minimal truck load in addition to the not greater time. So E_j dominates E'_j, if the following conditions hold:

$$
\begin{aligned}
&\text{(4a) with } res \in \{cost, time, cust_n (n \in N)\}, \text{(4c), (4d)} \\
E_j^{\overline{loadL}} + E_j^{loadH} &\leq E_j'^{\overline{loadL}} + E_j'^{loadH} \\
E_j^{\overline{loadL}} - \rho \cdot (E_j'^{time} - E_j^{time}) &\leq E_j'^{\overline{loadL}} \\
E_j^{\underline{loadL}} &\leq E_j'^{\underline{loadL}}
\end{aligned}
$$

5.3. Bidirectional Labeling

As described in Section 3.2.4, backward propagation can be done on a reversed network. The merging conditions need to be adapted slightly to consider the time-load tradeoff. The total vehicle capacity limits the sum of the maximal truck loads and the trailer loads. The combined truck load has to respect the capacity but can use the transfer potentials from both partial paths. The time difference between the forward and the backward label can be used for transfer, but is restricted by the sum of the transfer potentials $(\overline{loadL} - \underline{loadL})$. Thus, a forward label E_i can be concatenated with a backward label E'_i whose predecessor is given by $E'_j = pred(E'_i)$, if and only if the following conditions are respected:

$$\text{(6a)},\ \text{(6d)–(6f)}$$

$$E_i^{\overline{loadL}} + E_i^{loadH} + E_j'^{\overline{loadL}} + E_j'^{loadH} \leq Q_L^k + Q_H^k$$

$$E_i^{\overline{loadL}} + E_j'^{\overline{loadL}} - \min\{\rho \cdot (E_j'^{time} - t_{ij} - E_i^{time}), E_i^{\overline{loadL}} - E_i^{\underline{loadL}} + E_j'^{\overline{loadL}} - E_j'^{\underline{loadL}}\} \leq Q_L^k$$

6. Computational Study

The algorithms were implemented in C++ and use the LP and MIP solver of CPLEX 12.6.0. All computations were performed in single thread mode on a PC with an Intel Core i7-4790K CPU clocked at 3.60 GHz, with 8 GB RAM, running Windows 7 Enterprise. Apart from the single-thread mode, default settings were used for the CPLEX MIP solver. All tests were run with a CPU time limit of one hour.

6.1. Instances

To evaluate the performance of our algorithm, we used two different test sets.

The first instance set was developed by Drexl (2011). The instances are structured so as to resemble the situation of raw milk collection in Bavaria, Germany. Each instance has the same number n of truck customers, trailer customers, and dedicated transshipment locations, with $n \in \{6, \dots, 10, 25\}$. In the following, these instances are denoted by "TTRP_n". Remark that the number of vertices equals $2 + 8n$, as described in Section 3.2.1. For each value of n, there are 30 different instances. The vehicle fleet is heterogeneous, with two different truck types that can both be coupled with a suitable trailer, leading to four vehicle classes altogether. Costs are incurred for the covered distance, with additional costs for a coupled trailer. The customers have no time windows, but a maximal tour duration is given.

The second instance set was created by Parragh and Cordeau (2015) from the Solomon instances (Solomon, 1987) by identifying truck customers as described by Lin *et al.* (2011): customers were sorted by increasing distance from their nearest neighbor, and from each instance, three new instances were derived by taking the first 25 %, 50 %, and 75 % of customers as truck customers and the remaining ones as trailer customers. These instances are henceforth referred to by "OriginalName_n_p", with n indicating the number of customers (25, 50, 100) and p standing for the rounded percentage of truck customers (25, 50, 75). The number of vertices equals $2 + pn + 4(1 - p)n$. Besides, a trailer capacity was added, so that there are two vehicle classes, one representing a complete vehicle and one a truck without a trailer. As known, the Solomon instances have time windows, and costs and times are proportional to the Euclidean distance between locations, independent of whether or not a trailer is attached. Note that we rounded up the times to one decimal place. In these instances, there are no dedicated transshipment locations. As mentioned, Parragh and Cordeau (2015) require that trailer customer locations be used as parking places only if their supply is collected during the stop. Thus, we also respect this strict parking rule for these instances.

6.2. Results

Preliminary tests showed that the influence of a reduced network, a maximal number of columns to add per pricing iteration, and the size of the *ng*-neighborhood (*ng*-size) can be significant. We found that the best results concerning the thinned-out network were obtained by keeping, for each customer, only the arcs leading to the 7 most promising neighbors (those with lowest reduced costs) and to the 5 nearest transshipment locations. Also the depot was connected only to these 12 best neighbors. The maximal number of columns to add per pricing iteration was chosen dependent on the instance size: it was best to add at most six times the number of vertices. The best *ng*-size for the TTRP-instances was 6, whereas the modified Solomon instances could be solved better with an *ng*-size of 8. We limited the overall number of subset-row cuts to 120 and generated a maximum of 10 cuts simultaneously. We required a minimum violation of 0.1 and stopped the separation after the third level of the branch-and-bound tree. We report all computation times in seconds. Moreover, we define the relative optimality gap between the best found upper bound ub and the best found lower bound lb as $(ub - lb)/ub \cdot 100\,\%$.

In the following, we present aggregated results for the two instance sets. More detailed results, including the individual objective function values, can be found in the Appendix, Section II. Note that, unfortunately, the extended dominance rules described in Section 3.2.3 did not consistently reduce solution times; for several instances, the stronger dominance was offset by the increased computing time for the dominance check. For this reason, all presented results were obtained without the extended dominance tests.

The aggregated results for the TTRP instances are depicted in Table 3. Only ten of the TTRP instances with n between 6 and 10 could not be solved to optimality within one hour. Compared to the results of Drexl (2011), 36 new optimal solutions were found. The high number of instances solved in the extended root node by incorporating the subset-row cuts demonstrates the effectiveness of these inequalities. The highest remaining optimality gap overall was less than 5 %, and the largest instance that could be solved to optimality had 76 locations, i.e., 202 vertices.

Instance set	# instances solved to optimality			∅ CPU time	∅ gap
	Total	Root w/o cuts	Root with cuts		
TTRP_6	30/30	6	29	1.5	0.00
TTRP_7	30/30	6	28	36.7	0.00
TTRP_8	29/30	4	27	182.4	0.02
TTRP_9	27/30	3	25	392.5	0.05
TTRP_10	24/30	2	20	842.4	0.20
TTRP_25	1/30	0	1	3559.5	1.51
Total	141/180	21	130	835.8	0.29

Table 3: Aggregated results for TTRP instances

The aggregated results for the Solomon-based instances with 25 customers are presented in Table 4. All in all, only 2 of these 168 Solomon-based instances could not be solved to optimality within one hour of CPU time. As expected, a higher percentage of truck customers simplifies the problem, because fewer parking possibilities exist. Also for the Solomon-based instances, the subset-row inequalities had a large influence on the results. In the Appendix, Section II, in Tables 8 and 9, we also show our results for the 50- and 100-customer Solomon-based instances presented by Parragh and Cordeau (2015). Compared to the results of Parragh and Cordeau (2015), an optimal solution was found for 35 additional instances.

Running the Solomon-based instances for a two-day planning horizon yielded the results shown in Table 5. We assumed that every second customer was allowed to be visited every other day. A comparison of the effectiveness of the root stabilization described in Section 4 leads to a surprising result. Despite a much smaller root solution time when incorporating the stabilization methods, the total solution time was not reduced consistently, and even some significantly larger optimality gaps occurred. The reason is that many optimal solutions can be found already in the extended root node (with the SR cuts). In the case of root stabilization, the former integral solution is split evenly between both days, so that a potential integrality is mostly destroyed. Overall, nearly 70 % of the two-day instances could be solved to optimality within one hour of CPU time, although the number of tasks is twice as high as for the one-day planning horizon. Moreover, it is remarkable that more than 14 % of the costs can be saved by serving a part of the customers only every other day.

Results concerning the quantity-dependent transfer times were obtained with the Solomon-based instances, as the ones of Drexl (2011) have no time windows and are thus not influenced by non-constant transfer times. We assumed a transfer rate of one unit per second, i.e., $\rho = 1$. The computational results are summarized in Table 6. In the instance sets R1, R2, and RC2, all optimal solutions matched those for constant transfer times. This is simply because in both versions, all routes of optimal solutions are performed by single trucks. In the C1, C2, and RC1 instance groups, however, several optimal solutions differed from those for constant transfer times and had higher costs. Nevertheless, it is clear that the im-

Instance set	# instances solved to optimality			∅ CPU time	∅ gap
	Total	Root w/o cuts	Root with cuts		
C1_25_25	9/9	8	9	48.3	0
C1_25_50	9/9	3	9	43.2	0
C1_25_75	9/9	0	8	351.7	0
C2_25_25	7/8	0	8	846.9	0.10
C2_25_50	8/8	0	8	602.5	0
C2_25_75	8/8	0	8	335.6	0
R1_25_25	12/12	5	11	1.4	0
R1_25_50	12/12	5	12	0.5	0
R1_25_75	12/12	5	12	0.2	0
R2_25_25	10/11	1	10	531.3	0.00
R2_25_50	11/11	1	11	301.2	0
R2_25_75	11/11	1	11	52.4	0
RC1_25_25	8/8	7	8	3.1	0
RC1_25_50	8/8	7	8	1.4	0
RC1_25_75	8/8	7	8	0.5	0
RC2_25_25	8/8	8	8	339.3	0
RC2_25_50	8/8	8	8	227.9	0
RC2_25_75	8/8	8	8	37.0	0
Total	166/168	74	164	195.8	0.01

Table 4: Aggregated results for Solomon-based instances

pact of quantity-dependent transfer times is limited if the time windows are not tight and if the amount collected during all subtours barely exceeds the truck capacity. Finally, it is noteworthy that, despite the more complicated resource extension functions, the number of solved instances and the solution times are nearly as good as those for constant transfer times.

7. Conclusion

In this paper, we have studied the truck-and-trailer routing problem with time windows and two new extensions: the consideration of a two-day planning horizon where it is allowed to visit some customers only every other day, and the inclusion of a quantity-dependent time consumption for the transfer of load from a truck to its trailer. All variants were tackled by effective branch-and-price-and-cut algorithms, using subset-row inequalities to strengthen the lower bound of the linear relaxation. The two-day planning horizon leads to undesirable symmetries. To deal with these, we proposed a constraint-aggregation and stabilization procedure based on the concept of deep dual-optimal inequalities. Quantity-dependent load transfer times cause a tradeoff between the time needed for the transfer operation and the truck capacity gained. This requires additional resources and more complicated resource extension functions in the labeling algorithm for solving the pricing problems.

Computational experiments were performed with established TTRPTW benchmark instances. The results compare very favorably with those known from the literature; many instances could be solved to optimality for the first time.

Further research can be done regarding algorithmic as well as problem aspects. As for algorithmics, for the two-day planning horizon, an improved transition from the symmetric to the asymmetric treatment of the problem might reduce the run times significantly. To this end, it would be necessary to devise a more sophisticated procedure for finding suitable subsets of the first-day columns to copy to the second day.

	With stabilization				Without stabilization				
	# optimal		∅ CPU time		# optimal		∅ CPU time		
Instance set	Total	Root with cuts	Total	Root with cuts	Total	Root with cuts	Total	Root with cuts	∅ cost savings in %
C1_25	15/27	9	1695.6	137.5	15/27	14	1763.7	368.2	4.01
C2_25	10/24	7	2192.3	744.5	10/24	8	2282.7	1138.3	10.88
R1_25	29/36	2	1044.2	5.6	27/36	13	1175.6	19.3	17.46
R2_25	23/33	18	1537.9	912.9	21/33	15	1640.5	1047.0	22.13
RC1_25	20/24	18	609.2	8.9	21/24	21	480.3	31.0	14.56
RC2_25	18/24	18	916.1	776.1	18/24	18	917.0	849.6	14.76
Total	115/168	72	1329.4	421.1	112/168	89	1383.3	557.4	14.48

Table 5: Aggregated results for Solomon-based instances with two-day planning horizon

Instance set	# optimal	∅ CPU time	∅ gap	# changed solutions	∅ cost increase in %
C1_25	26/27	199.4	0.00	5	0.65
C2_25	23/24	749.3	0.07	1	0.04
R1_25	36/36	1.0	0	0	0
R2_25	32/33	297.6	0.01	0	0
RC1_25	24/24	6.8	0	22	6.35
RC2_25	24/24	249.1	0	0	0
Total	165/168	234.3	0.01	28	1.02

Table 6: Aggregated results for Solomon-based instances with quantity-dependent transfer times

From a modeling point of view, the next big step would be to abandon the fixed assignment of trailers to trucks. Especially if the service at customers takes a significant amount of time, it may be useful to leave a trailer at a customer for loading, continue the tour with the truck only, and recouple the trailer later with the same or a different compatible truck. This complicates the problem considerably, because a synchronization between all vehicles becomes necessary and the pricing problems for the different vehicles become interdependent.

Acknowledgement

The research of the second and the third author was supported by the Deutsche Forschungsgemeinschaft (DFG) under grants DR 963/2-1 and IR 122/6-1 respectively. This support is gratefully acknowledged.

References

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.

Batsyn, M. and Ponomarenko, A. (2014). Heuristic for a real-life truck and trailer routing problem. *Procedia Computer Science*, **31**, 778–792.

Belenguer, J., Benavent, E., Martínez, A., Prins, C., Prodhon, C., and Villegas, J. (2016). A branch-and-cut algorithm for the single truck and trailer routing problem with satellite depots. *Transportation Science*, **50**(2), 735–749.

Ben Amor, H., Desrosiers, J., and Valério de Carvalho, J. M. (2006). Dual-optimal inequalities for stabilized column generation. *Operations Research*, **54**(3), 454–463.

Bode, C. and Irnich, S. (2012). Cut-first branch-and-price-second for the capacitated arc-routing problem. *Operations Research*, **60**(5), 1167–1182.

Bodin, L. and Levy, L. (2000). Scheduling of local delivery carrier routes for the united states postal service. In M. Dror, editor, *Arc Routing: Theory, Solutions, and Applications*, pages 419–442. Kluwer, Boston.

Brønmo, G., Christiansen, M., and Nygreen, B. (2007). Ship routing and scheduling with flexible cargo sizes. *Journal of the Operational Research Society*, **58**(9), 1167–1177.

Caramia, M. and Guerriero, F. (2010a). A heuristic approach for the truck and trailer routing problem. *Journal of the Operational Research Society*, **61**(7), 1168–1180.

Caramia, M. and Guerriero, F. (2010b). A milk collection problem with incompatibility constraints. *Interfaces*, **40**(2), 130–143.

Chao, I. (2002). A tabu search method for the truck and trailer routing problem. *Computers & Operations Research*, **29**(1), 33–51.

Cuda, R., Guastaroba, G., and Speranza, M. (2015). A survey on two-echelon routing problems. *Computers & Operations Research*, **55**, 185–199.

Derigs, U., Pullmann, M., and Vogel, U. (2013). Truck and trailer routing—Problems, heuristics and computational experience. *Computers & Operations Research*, **40**(2), 536–546.

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., Errico, F., Irnich, S., and Schneider, M. (2014). Exact algorithms for electric vehicle-routing problems with time windows. Les Cahiers du GERAD G-2014-110, GERAD, Montréal, Canada.

Drexl, M. (2011). Branch-and-price and heuristic column generation for the generalized truck-and-trailer routing problem. *Journal of Quantitative Methods for Economics and Business Administration*, **12**, 5–38.

Gerdessen, J. (1996). Vehicle routing problem with trailers. *European Journal of Operational Research*, **93**(1), 135–147.

Gschwind, T. and Irnich, S. (2016). Dual inequalities for stabilized column generation revisited. *INFORMS Journal on Computing*, **28**(1), 175–194.

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.

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.

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.

Lin, S., Yu, V., and Lu, C. (2011). A simulated annealing heuristic for the truck and trailer routing problem with time windows. *Expert Systems with Applications*, **38**(12), 15244–15252.

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

Mirmohammadsadeghi, S. and Ahmed, S. (2015). Memetic heuristic approach for solving truck and trailer routing problems with stochastic demands and time windows. *Networks and Spatial Economics*, **15**(4), 1093–1115.

Parragh, S. and Cordeau, J.-F. (2015). Branch-and-price for the truck and trailer routing problem with time windows. Technical report, CIRRELT, Université de Montréal, Canada.

Pasha, U., Hoff, A., and Løkketangen, A. (2014). A hybrid approach for milk collection using trucks and trailers. *Annals of Management Science*.

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.

Scheuerer, S. (2006). A tabu search heuristic for the truck and trailer routing problem. *Computers & Operations Research*, **33**(4), 894–909.

Semet, F. and Taillard, E. (1993). Solving real-life vehicle routing problems efficiently using tabu search. *Annals of Operations Research*, **41**(4), 469–488.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, **35**(2), 254–265.

Tan, K., Chew, Y., and Lee, L. (2006). A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, **172**(3), 855–885.

Tilk, C., Bianchessi, N., Drexl, M., Irnich, S., and Meisel, F. (2016). Branch-and-price for the active-passive vehicle-routing problem. *Transportation Science*. Forthcoming.

Torres, I., Cruz, C., and Verdegay, J. L. (2015). Solving the truck and trailer routing problem with fuzzy constraints. *International Journal of Computational Intelligence Systems*, **8**(4), 713–724.

Tricoire, F. (2016). Private communication.

Villegas, J., Prins, C., Prodhon, C., Medaglia, A., and Velasco, N. (2010). GRASP/VND and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots. *Engineering Applications of Artificial Intelligence*, **23**(5), 780–794.

Villegas, J., Prins, C., Prodhon, C., Medaglia, A., and Velasco, N. (2011). A GRASP with evolutionary path relinking for the truck and trailer routing problem. *Computers & Operations Research*, **38**(9), 1319–1334.

Villegas, J., Prins, C., Prodhon, C., Medaglia, A., and Velasco, N. (2013). A matheuristic for the truck and trailer routing problem. *European Journal of Operational Research*, **230**(2), 231–244.

Villeneuve, D. and Desaulniers, G. (2005). The shortest path problem with forbidden paths. *European Journal of Operational Research*, **165**(1), 97–107.

Appendix

I. Resource Extension Functions for Quantity-Dependent Load Transfer Times

In this section, we present the complete set of REFs for quantity-dependent load transfer times (see Section 5) grouped by resource.

$$E_j^{cost} = E_i^{cost} + \tilde{c}_{ij}^{k}(\delta_{E_i^{posH},\perp})$$

$$E_j^{time} = \begin{cases} \max\{e_j,\ E_i^{time} + t_{ij} + R_{ij}^{NLTB}(E_i)/\rho\}, & j \in N \text{ and } E_i^{posH} \neq \perp \\ \max\{e_j,\ E_i^{time} + t_{ij}\}, & \text{otherwise.} \end{cases}$$

$$E_j^{\overline{loadL}} = \begin{cases} E_i^{\overline{loadL}} + q_j - \max\left\{R_{ij}^{NLTB}(E_i),\ R_{ij}^{LTWT}(E_i)\right\}, & j \in N \text{ and } E_i^{posH} \neq \perp \\ E_i^{\overline{loadL}} - \min\{R_{ij}^{LTWT}(E_i), Q_H^k - E_i^{loadH} - q_j\}, & j \in N^H \text{ and } E_i^{posH} = \perp \\ E_i^{\overline{loadL}}, & j \in D \cup \mathcal{T} \cup C \cup d^- \end{cases}$$

$$E_j^{\underline{loadL}} = \begin{cases} \max\{E_i^{\underline{loadL}} + q_j, R_{ij}^{MinLL}(E_i)\}, & j \in N \text{ and } E_i^{posH} \neq \perp \\ \max\{E_i^{\underline{loadL}} + \max\{E_i^{loadH} + q_j - Q_H^k, 0\}, R_{ij}^{MinLL}(E_i)\}, & j \in N^H \text{ and } E_i^{posH} = \perp \\ R_{ij}^{MinLL}(E_i), & j \in D \cup \mathcal{T} \cup C \cup d^- \end{cases}$$

$$E_j^{loadH} = \begin{cases} E_i^{loadH} + \max\{R_{ij}^{NLTB}(E_i),\ R_{ij}^{LTWT}(E_i)\}, & j \in N \text{ and } E_i^{posH} \neq \perp \\ E_i^{loadH} + q_j + \min\{R_{ij}^{LTWT}(E_i), Q_H^k - E_i^{loadH} - q_j\}, & j \in N^H \text{ and } E_i^{posH} = \perp \\ E_i^{loadH}, & j \in D \cup \mathcal{T} \cup C \cup d^- \end{cases}$$

$$E_j^{posH} = \begin{cases} E_i^{posH}, & j \in N \cup \mathcal{T} \\ p, & j \in D \text{ where } j = d(p), p \in P \\ \perp, & j \in C \cup \{d^-\} \end{cases}$$

$$E_j^{cust_n} = \begin{cases} E_i^{cust_n} + 1, & j \in N \text{ and } j = n \\ \max\{E_i^{cust_n}, U_n(E_j)\}, & \text{otherwise.} \end{cases}$$

II. Detailed Computational Results

Table 7 shows a comparison of the performance of the different labeling directions in the pricing problem. For the monodirectional forward, the monodirectional backward and the bidirectional labeling the number of instances solved to optimality, the average CPU time in seconds, the average relative gap between best upper and best lower bound, and the number of generated labels per instance group in 10^6 are given. As expected, the bidirectional labeling could reduce the runtime by about 40 % compared to the monodirectional labeling.

	Forward				Backward				Bidirectional			
Instance set	Opt.	Time	Gap	#Lab.	Opt.	Time	Gap	#Lab.	Opt.	Time	Gap	#Lab.
C1_25	26/27	301.7	0.03	11.70	26/27	462.2	0.05	16.23	27	147.8	0	8.10
C2_25	21/24	937.5	0.20	24.67	20/24	1134.3	0.27	29.87	23	595.0	0.05	20.05
R1_25	36/36	1.0	0	0.46	36/36	1.6	0	0.54	36	0.7	0	0.36
R2_25	32/33	446.9	0.08	16.39	32/33	408.0	0.00	15.58	32	295.0	0,00	11.98
RC1_25	24/24	2.5	0	1.06	24/24	8.6	0	1.67	24	1.7	0	0.88
RC2_25	24/24	292.6	0	6.18	24/24	137.4	0	5.60	24	201.4	0	5.08
Total	163/168	312.6	0.05	9.76	162/168	337.7	0.05	11.09	166/168	195.9	0.01	7.45

Table 7: Comparison of labeling strategies for Solomon-based instances with 25 customers

The following tables show detailed computational results for the Solomon-based instances as well as for the TTRP instances with varying number of customers and additional transshipment locations. All columns have the same headers and inform about the tested instance, the lower bound, the upper bound, the resulting optimality gap, the solution time in seconds (with T.L. indicating the time limit of 1 hour), the number of added subset-row inequalities, the number of branch-and-bound nodes and the number of generated routes. Instances that could be solved to optimality for the first time are printed in bold.

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
C101_50_25	396.02	396.02	0	3.1	0	1	2,198
C101_50_50	426.08	426.08	0	3.6	20	1	2,347
C101_50_75	453.39	453.39	0	4.4	60	1	2,053
C201_50_25	391.97	391.97	0	145.8	20	1	20,891
C201_50_50	403.69	406.30	0.24	T.L.	110	1	20,315
C201_50_75	403.39	406.42	0.48	T.L.	110	2	27,505
R101_50_25	1046.70	1046.70	0	0.1	0	1	595
R101_50_50	1046.70	1046.70	0	0.1	0	1	525
R101_50_75	1046.70	1046.70	0	0	0	1	626
R201_50_25	794.34	794.34	0	16.3	10	1	4,882
R201_50_50	794.34	794.34	0	1.7	0	1	5,778
R201_50_75	794.34	794.34	0	0.6	0	1	4,718
RC101_50_25	945.68	959.38	1.43	T.L.	120	82	104,339
RC101_50_50	959.92	959.92	0	22.7	120	5	4,936
RC101_50_75	976.20	976.20	0	6.2	110	1	1,843
RC201_50_25	686.31	686.31	0	17.7	0	1	5,628
RC201_50_50	686.31	686.31	0	3.1	0	1	4,816
RC201_50_75	686.31	686.31	0	0.7	0	1	4,657

Table 8: Detailed results for Solomon-based instances with 50 customers

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
C101_100_25	899.75	899.75	0	80.7	20	1	7,646
C101_100_50	986.41	1003.86	1.74	T.L.	120	1	91,990
C101_100_75	1042.04	1043.28	0.12	T.L.	120	1	75,380
C201_100_25	662.46	695.51	1.34	T.L.	20	1	65,535
C201_100_50	692.67	719.04	3.67	T.L.	60	1	66,124
C201_100_75	704.40	737.08	0.90	T.L.	120	2	41,711
R101_100_25	1644.64	1644.64	0	28.9	23	1	11,048
R101_100_50	1644.64	1644.64	0	2.7	13	1	4,278
R101_100_75	1644.64	1644.64	0	1.3	14	1	3,055
R201_100_25	1146.69	1161.48	0.16	T.L.	50	1	53,157
R201_100_50	1147.65	1247.81	0.01	T.L.	120	1	61,870
R201_100_75	1147.80	1147.80	0	687	120	1	54,211
RC101_100_25	1653.93	1653.93	0	1337.9	120	82	48,098
RC101_100_50	1706.56	1734.14	1.59	T.L.	120	5	141,283
RC101_100_75	1770.71	1788.68	1.00	T.L.	120	1	88,300
RC201_100_25	1258.73	1290.68	0.93	T.L.	30	1	44,097
RC201_100_50	1264.60	1265.56	0.08	T.L.	120	1	91,870
RC201_100_75	1265.56	1265.56	0	581.3	120	1	42,941

Table 9: Detailed results for Solomon-based instances with 100 customers

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
C101_25_25	205.21	205.21	0	0.1	0	1	714
C101_25_50	219.58	219.58	0	0.3	10	1	896
C101_25_75	235.18	235.18	0	0.4	50	1	815
C102_25_25	203.68	203.68	0	8.3	0	1	2,624
C102_25_50	218.34	218.34	0	3.2	10	1	1,755
C102_25_75	235.18	235.18	0	33.4	88	1	1,660
C103_25_25	203.68	203.68	0	28.8	0	1	2,477
C103_25_50	215.88	215.88	0	24.1	20	1	2,234
C103_25_75	235.18	235.18	0	120.8	77	1	2,506
C104_25_25	200.39	200.39	0	392.2	10	1	3,361
C104_25_50	212.89	212.89	0	356.5	20	1	4,026
C104_25_75	232.71	232.71	0	2,995.0	91	5	6,290
C105_25_25	204.92	204.92	0	0.2	0	1	977
C105_25_50	218.95	218.95	0	0.4	10	1	931
C105_25_75	235.18	235.18	0	0.9	70	1	795
C106_25_25	205.21	205.21	0	0.1	0	1	761
C106_25_50	219.58	219.58	0	0.4	10	1	1,042
C106_25_75	235.18	235.18	0	0.6	55	1	759
C107_25_25	204.92	204.92	0	0.5	0	1	684
C107_25_50	208.94	208.94	0	0.3	0	1	688
C107_25_75	227.08	227.08	0	0.6	40	1	738
C108_25_25	204.92	204.92	0	1.3	0	1	1,415
C108_25_50	208.94	208.94	0	0.7	0	1	925
C108_25_75	227.08	227.08	0	2.3	40	1	1,521
C109_25_25	204.31	204.31	0	3.6	0	1	1,624
C109_25_50	208.33	208.33	0	3.3	0	1	1,755
C109_25_75	227.08	227.08	0	11.2	50	1	2,133
C201_25_25	223.07	223.07	0	4.4	30	1	3,643
C201_25_50	223.07	223.07	0	1.5	30	1	2,423
C201_25_75	227.29	227.29	0	15.6	80	1	3,602
C202_25_25	220.52	220.52	0	84.1	30	1	6,076
C202_25_50	220.52	220.52	0	15.6	30	1	4,987
C202_25_75	220.52	220.52	0	4.2	30	1	4,344
C203_25_25	220.52	220.52	0	962.5	50	1	7,440
C203_25_50	220.52	220.52	0	109.7	40	1	7,096
C203_25_75	220.52	220.52	0	32.6	50	1	7,325
C204_25_25	217.38	217.38	0	1,347.2	10	1	11,579
C204_25_50	218.16	218.16	0	706.4	20	1	10,014
C204_25_75	220.32	220.32	0	645.6	60	1	9,130
C205_25_25	223.07	223.07	0	16.1	30	1	4,712
C205_25_50	223.07	223.07	0	5.5	30	1	5,329
C205_25_75	225.66	225.66	0	5.5	60	1	3,793
C206_25_25	223.07	223.07	0	26.2	30	1	5,437
C206_25_50	223.07	223.07	0	10.0	30	1	4,677
C206_25_75	225.66	225.66	0	10.0	60	1	4,895
C207_25_25	222.94	225.45	1.11	T.L.	50	1	9,860
C207_25_50	222.87	222.87	0	3599.9	50	1	7,797
C207_25_75	225.45	225.45	0	1,903.6	70	1	5,321
C208_25_25	222.90	222.90	0	683.4	40	1	7,162
C208_25_50	222.90	222.90	0	369.0	40	1	6,808
C208_25_75	225.49	225.49	0	67.3	70	1	5,046

Table 10: Detailed results for Solomon-based instances C

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
R101_25_25	618.33	618.33	0	0.0	0	1	134
R101_25_50	618.33	618.33	0	0.0	0	1	134
R101_25_75	618.33	618.33	0	0.0	0	1	129
R102_25_25	548.11	548.11	0	0.1	7	1	506
R102_25_50	548.11	548.11	0	0.1	7	1	506
R102_25_75	548.11	548.11	0	0.0	9	1	513
R103_25_25	455.70	455.70	0	0.2	0	1	733
R103_25_50	455.70	455.70	0	0.1	0	1	642
R103_25_75	455.70	455.70	0	0.0	0	1	541
R104_25_25	417.96	417.96	0	0.7	0	1	1,022
R104_25_50	417.96	417.96	0	0.2	0	1	809
R104_25_75	417.96	417.96	0	0.1	0	1	825
R105_25_25	531.54	531.54	0	0.0	0	1	247
R105_25_50	531.54	531.54	0	0.0	0	1	247
R105_25_75	531.54	531.54	0	0.0	0	1	237
R106_25_25	466.48	466.48	0	0.3	10	1	682
R106_25_50	466.48	466.48	0	0.1	10	1	595
R106_25_75	466.48	466.48	0	0.0	10	1	507
R107_25_25	429.20	429.20	0	1.5	20	1	780
R107_25_50	429.20	429.20	0	0.5	20	1	772
R107_25_75	429.20	429.20	0	0.2	10	1	676
R108_25_25	404.28	404.28	0	3.8	20	1	994
R108_25_50	404.28	404.28	0	2.0	30	1	1,126
R108_25_75	404.28	404.28	0	0.4	20	1	1,208
R109_25_25	442.63	442.63	0	0.0	0	1	490
R109_25_50	442.63	442.63	0	0.0	0	1	490
R109_25_75	442.63	442.63	0	0.0	0	1	451
R110_25_25	445.18	445.18	0	2.7	24	7	4,020
R110_25_50	445.18	445.18	0	0.3	20	1	509
R110_25_75	445.18	445.18	0	0.1	20	1	509
R111_25_25	429.70	429.70	0	0.6	10	1	1,051
R111_25_50	429.70	429.70	0	0.3	10	1	852
R111_25_75	429.70	429.70	0	0.1	10	1	744
R112_25_25	402.85	402.85	0	7.2	40	1	956
R112_25_50	402.85	402.85	0	2.8	40	1	932
R112_25_75	402.85	402.85	0	1.2	40	1	719

Table 11: Detailed results for Solomon-based instances R1

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
R201_25_25	464.38	464.38	0	0.3	9	1	968
R201_25_50	464.38	464.38	0	0.1	9	1	817
R201_25_75	464.38	464.38	0	0.1	9	1	817
R202_25_25	411.49	411.49	0	1.1	0	1	4,557
R202_25_50	411.49	411.49	0	0.7	0	1	4,339
R202_25_75	411.49	411.49	0	0.2	0	1	3,202
R203_25_25	392.33	392.33	0	27.6	10	1	6,646
R203_25_50	392.33	392.33	0	7.4	10	1	4,579
R203_25_75	392.33	392.33	0	2.7	10	1	3,878
R204_25_25	355.89	355.89	0	1,133.3	40	1	7,529
R204_25_50	355.89	355.89	0	1,348.9	80	1	8,153
R204_25_75	355.89	355.89	0	69.0	70	1	7,338
R205_25_25	394.06	394.06	0	2.2	10	1	3,454
R205_25_50	394.06	394.06	0	0.8	10	1	2,920
R205_25_75	394.06	394.06	0	0.5	10	1	2,813
R206_25_25	375.48	375.48	0	32.0	10	1	6,468
R206_25_50	375.48	375.48	0	6.3	10	1	7,001
R206_25_75	375.48	375.48	0	1.6	10	1	5,141
R207_25_25	362.64	362.64	0	259.4	10	1	10,796
R207_25_50	362.64	362.64	0	92.7	10	1	7,781
R207_25_75	362.64	362.64	0	32.8	10	1	7,724
R208_25_25	329.33	329.33	0	658.3	10	1	11,670
R208_25_50	329.33	329.33	0	230.4	10	1	12,023
R208_25_75	329.33	329.33	0	39.4	10	1	9,973
R209_25_25	371.56	371.56	0	67.1	60	1	6,042
R209_25_50	371.56	371.56	0	14.4	50	1	5,990
R209_25_75	371.56	371.56	0	4.8	50	1	4,458
R210_25_25	405.48	405.48	0	16.9	10	1	6,808
R210_25_50	405.48	405.48	0	5.9	10	1	5,540
R210_25_75	405.48	405.48	0	2.4	10	1	4,730
R211_25_25	351.75	351.91	0.04	T.L.	30	1	8,561
R211_25_50	351.91	351.91	0	1,605.6	50	1	8,862
R211_25_75	351.91	351.91	0	422.9	60	1	7,067

Table 12: Detailed results for Solomon-based instances R2

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
RC101_25_25	473.54	473.54	0	0.7	46	1	621
RC101_25_50	478.73	478.73	0	0.4	40	1	686
RC101_25_75	487.76	487.76	0	0.3	50	1	632
RC102_25_25	363.86	363.86	0	0.8	0	1	1,078
RC102_25_50	369.68	369.68	0	0.5	0	1	1,300
RC102_25_75	391.38	391.38	0	0.1	0	1	847
RC103_25_25	346.51	346.51	0	1.6	0	1	1,474
RC103_25_50	354.13	354.13	0	0.7	0	1	1,500
RC103_25_75	373.28	373.28	0	0.2	0	1	1,013
RC104_25_25	320.61	320.61	0	6.2	0	1	1,860
RC104_25_50	342.02	342.02	0	2.4	0	3	1,461
RC104_25_75	361.07	361.07	0	0.6	0	1	800
RC105_25_25	412.56	412.56	0	0.2	0	1	475
RC105_25_50	419.72	419.72	0	0.2	0	1	798
RC105_25_75	434.35	434.35	0	0.1	0	1	608
RC106_25_25	355.32	355.32	0	0.3	0	1	618
RC106_25_50	361.25	361.25	0	0.2	0	1	608
RC106_25_75	388.71	388.71	0	0.1	0	1	691
RC107_25_25	318.46	318.46	0	2.2	0	1	1,335
RC107_25_50	333.23	333.23	0	1.2	0	1	1,395
RC107_25_75	345.55	345.55	0	0.5	0	1	1,091
RC108_25_25	314.64	314.64	0	13.0	0	1	2,898
RC108_25_50	331.25	331.25	0	5.9	0	1	2,101
RC108_25_75	345.55	345.55	0	2.1	0	1	1,494
RC201_25_25	361.24	361.24	0	0.1	0	1	745
RC201_25_50	361.24	361.24	0	0.1	0	1	745
RC201_25_75	361.24	361.24	0	0.0	0	1	745
RC202_25_25	338.82	338.82	0	0.9	0	1	1,273
RC202_25_50	338.82	338.82	0	0.4	0	1	1,273
RC202_25_75	338.82	338.82	0	0.1	0	1	1,273
RC203_25_25	327.69	327.69	0	19.9	0	1	2,230
RC203_25_50	327.69	327.69	0	2.7	0	1	2,170
RC203_25_75	327.69	327.69	0	1.0	0	1	2,235
RC204_25_25	300.24	300.24	0	66.7	0	1	7,560
RC204_25_50	300.24	300.24	0	38.0	0	1	10,317
RC204_25_75	300.24	300.24	0	15.0	0	1	9,491
RC205_25_25	338.93	338.93	0	2.4	0	1	2,371
RC205_25_50	338.93	338.93	0	1.2	0	1	1,852
RC205_25_75	338.93	338.93	0	0.2	0	1	1,496
RC206_25_25	325.10	325.10	0	1.1	0	1	2,049
RC206_25_50	325.10	325.10	0	0.4	0	1	2,709
RC206_25_75	325.10	325.10	0	0.2	0	1	2,285
RC207_25_25	298.95	298.95	0	7.7	0	1	7,442
RC207_25_50	298.95	298.95	0	2.2	0	1	2,640
RC207_25_75	298.95	298.95	0	0.6	0	1	1,858
RC208_25_25	269.57	269.57	0	2,615.1	0	1	10,601
RC208_25_50	269.57	269.57	0	1,778.4	0	1	11,410
RC208_25_75	269.57	269.57	0	278.6	0	1	9,518

Table 13: Detailed results for Solomon-based instances RC

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
TTRP_6_0	163,874	163,874	0	1.1	16	1	1,004
TTRP_6_1	143,909	143,909	0	1.0	10	1	1,152
TTRP_6_2	145,645	145,645	0	0.4	0	1	999
TTRP_6_3	197,850	197,850	0	0.3	9	1	619
TTRP_6_4	178,383	178,383	0	0.2	10	1	568
TTRP_6_5	142,888	142,888	0	1.8	15	1	1,190
TTRP_6_6	145,040	145,040	0	1.5	10	1	1,235
TTRP_6_7	148,384	148,384	0	1.2	10	1	1,094
TTRP_6_8	173,353	173,353	0	1.4	20	1	644
TTRP_6_9	147,385	147,385	0	1.2	20	1	1,056
TTRP_6_10	157,903	157,903	0	0.5	3	1	857
TTRP_6_11	152,266	152,266	0	1.5	25	1	975
TTRP_6_12	163,427	163,427	0	3.4	30	1	1,016
TTRP_6_13	143,010	143,010	0	0.6	10	1	1,537
TTRP_6_14	164,496	164,496	0	1.7	10	1	949
TTRP_6_15	155,263	155,263	0	1.1	10	1	938
TTRP_6_16	165,577	165,577	0	0.7	0	1	837
TTRP_6_17	173,967	173,967	0	0.2	0	1	840
TTRP_6_18	149,120	149,120	0	2.2	19	1	870
TTRP_6_19	136,069	136,069	0	1.5	17	1	867
TTRP_6_20	113,710	113,710	0	1.2	0	1	1,655
TTRP_6_21	154,589	154,589	0	0.5	10	1	1,012
TTRP_6_22	157,212	157,212	0	1.0	10	1	867
TTRP_6_23	152,416	152,416	0	0.5	0	1	1,077
TTRP_6_24	157,880	157,880	0	2.6	10	1	1,212
TTRP_6_25	150,638	150,638	0	0.8	0	1	,903
TTRP_6_26	173,962	173,962	0	0.7	10	1	770
TTRP_6_27	171,155	171,155	0	0.7	9	1	973
TTRP_6_28	178,840	178,840	0	2.7	20	1	1,000
TTRP_6_29	165,292	165,292	0	9.6	28	3	1,607
TTRP_7_0	177,536	177,536	0	2.3	20	1	1,230
TTRP_7_1	167,016	167,016	0	5.3	20	1	1,241
TTRP_7_2	189,518	189,518	0	0.4	10	1	759
TTRP_7_3	198,714	198,714	0	0.6	20	1	1,209
TTRP_7_4	187,852	187,852	0	1.4	10	1	616
TTRP_7_5	178,057	178,057	0	1.1	19	1	1,285
TTRP_7_6	171,881	171,881	0	70.7	56	3	2,332
TTRP_7_7	197,741	197,741	0	0.8	10	1	1,333
TTRP_7_8	192,178	192,178	0	1.8	10	1	982
TTRP_7_9	189,818	189,818	0	1.1	8	1	1,252
TTRP_7_10	188,080	188,080	0	0.7	0	1	1,222
TTRP_7_11	183,926	183,926	0	0.9	10	1	931
TTRP_7_12	163,727	163,727	0	2.2	10	1	1,284
TTRP_7_13	181,886	181,886	0	0.6	0	1	859
TTRP_7_14	187,342	187,342	0	1.4	10	1	1,288
TTRP_7_15	192,112	192,112	0	0.3	0	1	930
TTRP_7_16	193,950	193,950	0	3.2	21	1	1,123
TTRP_7_17	172,506	172,506	0	0.7	0	1	811
TTRP_7_18	191,093	191,093	0	991.1	36	3	8,059
TTRP_7_19	190,032	190,032	0	1.4	20	1	1,106
TTRP_7_20	179,299	179,299	0	2.4	20	1	1,893
TTRP_7_21	186,217	186,217	0	0.7	10	1	1,195
TTRP_7_22	202,932	202,932	0	1.0	18	1	1,124
TTRP_7_23	186,509	186,509	0	0.7	10	1	995
TTRP_7_24	191,921	191,921	0	0.3	0	1	718
TTRP_7_25	191,909	191,909	0	3.7	18	1	1,180
TTRP_7_26	200,460	200,460	0	0.5	10	1	1,071
TTRP_7_27	188,632	188,632	0	0.2	0	1	821
TTRP_7_28	173,434	173,434	0	1.7	10	1	1,721
TTRP_7_29	159,213	159,213	0	1.6	10	1	1,440

Table 14: Detailed results for TTRP instances size 6 and 7

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
TTRP_8_0	204,862	204,862	0	3.5	10	1	2,494
TTRP_8_1	195,854	195,854	0	23.0	51	1	1,700
TTRP_8_2	191,907	191,907	0	8.8	30	1	2,448
TTRP_8_3	203,350	203,350	0	2.3	10	1	1,555
TTRP_8_4	194,293	194,293	0	2.4	24	1	1,583
TTRP_8_5	201,960	201,960	0	223.7	50	15	6,239
TTRP_8_6	205,829	205,829	0	7.1	20	1	1,399
TTRP_8_7	204,158	204,158	0	4.2	24	1	2,055
TTRP_8_8	193,122	193,122	0	2.8	10	1	2,495
TTRP_8_9	194,797	194,797	0	1.8	0	1	1,584
TTRP_8_10	163,595	163,595	0	25.3	30	1	2,909
TTRP_8_11	201,075	201,075	0	7.5	30	1	2,451
TTRP_8_12	185,829	185,829	0	36.3	48	1	2,475
TTRP_8_13	231,992	231,992	0	1,377.0	24	2 085	2,014,384
TTRP_8_14	227,510	227,510	0	3.6	29	1	1,337
TTRP_8_15	188,023	189,025	0.53	T.L.	32	978	2,039,139
TTRP_8_16	168,893	168,893	0	32.5	20	1	2,760
TTRP_8_17	220,641	220,641	0	2.7	22	1	1,185
TTRP_8_18	186,792	186,792	0	1.4	0	1	1,775
TTRP_8_19	203,236	203,236	0	1.6	0	1	2,588
TTRP_8_20	179,462	179,462	0	3.3	20	1	1,225
TTRP_8_21	208,664	208,664	0	28.7	38	1	1,631
TTRP_8_22	228,495	228,495	0	2.3	35	1	2,245
TTRP_8_23	210,647	210,647	0	12.1	18	1	2,873
TTRP_8_24	181,086	181,086	0	2.2	9	1	1,639
TTRP_8_25	185,788	185,788	0	1.5	10	1	1,660
TTRP_8_26	166,093	166,093	0	27.2	20	1	2,378
TTRP_8_27	195,398	195,398	0	1.9	0	1	988
TTRP_8_28	204,747	204,747	0	23.6	30	1	2,155
TTRP_8_29	191,064	191,064	0	1.3	10	1	1,647
TTRP_9_0	220,184	220,184	0	11.7	14	1	3,731
TTRP_9_1	192,652	192,652	0	41.7	30	1	2,162
TTRP_9_2	220,404	220,404	0	1.6	10	1	2,350
TTRP_9_3	195,389	197,215	0.93	T.L.	68	12	5,958
TTRP_9_4	207,619	207,619	0	6.6	10	1	3,019
TTRP_9_5	207,447	208,450	0.48	T.L.	87	22	13,701
TTRP_9_6	237,694	237,694	0	5.2	30	1	1,734
TTRP_9_7	229,948	229,948	0	57.3	52	1	1,654
TTRP_9_8	212,826	212,826	0	19.8	20	1	2,151
TTRP_9_9	234,673	234,673	0	1.6	10	1	1,873
TTRP_9_10	189,741	189,741	0	9.8	28	1	2,088
TTRP_9_11	190,269	190,269	0	2.9	0	1	2,240
TTRP_9_12	198,115	198,115	0	35.0	50	1	3,200
TTRP_9_13	199,631	199,631	0	3.7	0	1	2,753
TTRP_9_14	208,587	208,587	0	16.9	20	1	2,805
TTRP_9_15	199,285	199,285	0	8.6	20	1	3,539
TTRP_9_16	212,885	212,885	0	3.1	10	1	2,398
TTRP_9_17	208,499	208,499	0	21.3	30	1	3,348
TTRP_9_18	237,788	237,788	0	0.9	9	1	1,368
TTRP_9_19	214,741	214,741	0	118.0	30	1	2,783
TTRP_9_20	217,954	217,954	0	124.3	45	1	3,593
TTRP_9_21	220,973	220,973	0	10.0	20	1	2,641
TTRP_9_22	199,360	199,360	0	239.5	34	7	14,725
TTRP_9_23	265,177	265,177	0	1.0	0	1	2,123
TTRP_9_24	198,701	198,825	0.06	T.L.	70	68	128,880
TTRP_9_25	189,096	189,096	0	11.1	20	1	3,041
TTRP_9_26	239,434	239,434	0	3.5	10	1	2,300
TTRP_9_27	236,151	236,151	0	4.4	20	1	1,577
TTRP_9_28	179,036	179,036	0	51.2	20	1	4,271
TTRP_9_29	214,870	214,870	0	153.7	42	3	3,048

Table 15: Detailed results for TTRP instances size 8 and 9

Instance	LB	UB	Gap	Time	#cuts	#nodes	#columns
TTRP_10_0	250,815	250,815	0	254.5	63	3	2,893
TTRP_10_1	262,139	262,139	0	36.0	30	1	3,197
TTRP_10_2	237,509	237,509	0	37.8	36	1	2,840
TTRP_10_3	244,731	244,731	0	15.6	20	1	2,907
TTRP_10_4	266,077	266,077	0	9.7	30	1	2,480
TTRP_10_5	243,870	243,870	0	4.5	10	1	2,209
TTRP_10_6	247,492	249,605	0.85	T.L.	74	81	151,803
TTRP_10_7	231,129	231,568	0.19	T.L.	48	46	114,560
TTRP_10_8	217,021	217,021	0	11.1	20	1	3,299
TTRP_10_9	248,862	248,862	0	80.4	53	3	2,662
TTRP_10_10	262,310	262,310	0	135.8	42	3	4,644
TTRP_10_11	221,083	221,645	0.25	T.L.	66	136	316,716
TTRP_10_12	246,323	246,323	0	10.1	23	1	2,595
TTRP_10_13	261,842	261,842	0	5.4	10	1	3,916
TTRP_10_14	245,973	245,973	0	5.4	18	1	2,037
TTRP_10_15	213,753	213,753	0	51.2	0	1	4,565
TTRP_10_16	207,860	207,860	0	151.8	57	1	4,395
TTRP_10_17	209,614	210,253	0.3	T.L.	64	8	13,114
TTRP_10_18	216,106	219,933	1.74	T.L.	81	12	18,806
TTRP_10_19	207,686	207,686	0	17.1	10	1	3,727
TTRP_10_20	228,717	228,717	0	17.3	30	1	2,568
TTRP_10_21	205,494	205,494	0	57.4	10	1	5,312
TTRP_10_22	224,851	230,687	2.53	T.L.	57	1	9,499
TTRP_10_23	241,509	241,509	0	4.6	0	1	2,580
TTRP_10_24	221,766	221,766	0	202.4	40	1	6,756
TTRP_10_25	233,261	233,261	0	3.0	10	1	2,195
TTRP_10_26	238,139	238,139	0	27.7	30	1	3,456
TTRP_10_27	249,677	249,677	0	19.7	30	1	2,297
TTRP_10_28	209,008	209,008	0	2,006.1	51	3	7,833
TTRP_10_29	238,367	238,367	0	66.6	34	1	4,509
TTRP_25_0	492,306	498,259	1.19	T.L.	40	1	16,571
TTRP_25_1	484,775	487,918	0.64	T.L.	50	1	12,321
TTRP_25_2	510,294	520,253	1.91	T.L.	40	1	14,279
TTRP_25_3	523,059	524,886	0.35	T.L.	90	1	9,785
TTRP_25_4	548,321	550,491	0.39	T.L.	70	1	11,622
TTRP_25_5	533,485	559,913	4.72	T.L.	70	1	13,430
TTRP_25_6	544,724	548,438	0.68	T.L.	70	1	14,840
TTRP_25_7	534,811	534,811	0	2,291.9	60	1	11,583
TTRP_25_8	511,184	511,508	0.06	T.L.	60	1	11,061
TTRP_25_9	556,671	560,022	0.6	T.L.	50	1	13,569
TTRP_25_10	532,491	546,649	2.59	T.L.	40	1	14,853
TTRP_25_11	489,122	503,341	2.82	T.L.	40	1	13,766
TTRP_25_12	543,077	548,557	1	T.L.	113	1	9,673
TTRP_25_13	526,345	544,599	3.35	T.L.	30	1	11,882
TTRP_25_14	518,128	528,855	2.03	T.L.	50	1	14,155
TTRP_25_15	546,404	550,604	0.76	T.L.	70	1	12,882
TTRP_25_16	518,744	519,920	0.23	T.L.	50	1	12,336
TTRP_25_17	478,372	496,769	3.7	T.L.	20	1	14,174
TTRP_25_18	533,444	533,471	0.01	T.L.	60	1	12,144
TTRP_25_19	506,870	523,373	3.15	T.L.	50	1	13,181
TTRP_25_20	506,401	507,249	0.17	T.L.	80	1	11,259
TTRP_25_21	482,849	490,077	1.47	T.L.	40	1	14,300
TTRP_25_22	493,621	495,046	0.29	T.L.	70	1	14,914
TTRP_25_23	575,009	578,268	0.56	T.L.	80	6	22,709
TTRP_25_24	518,835	543,377	4.52	T.L.	40	1	13,769
TTRP_25_25	538,252	542,546	0.79	T.L.	80	1	15,357
TTRP_25_26	516,290	537,281	3.91	T.L.	40	1	14,175
TTRP_25_27	528,238	536,408	1.52	T.L.	60	1	13,041
TTRP_25_28	511,719	518,385	1.29	T.L.	50	1	12,054
TTRP_25_29	538,870	541,306	0.45	T.L.	100	1	12,048

Table 16: Detailed results for TTRP instances size 10 and 25