Branch-and-Cut Algorithms for the Vehicle Routing Problem with Trailers and Transshipments

Technical Report LM-2012-04

Michael Drexl
Chair of Logistics Management, Gutenberg School of Management and Economics,
Johannes Gutenberg University, Mainz
and
Fraunhofer Centre for Applied Research on Supply Chain Services SCS, Nuremberg
E-mail: drexl@uni-mainz.de

25th September 2012

Abstract

This paper studies the vehicle routing problem with trailers and transshipments (VRPTT), a practically relevant, but challenging, generalization of the classical vehicle routing problem. The paper makes three contributions: (i) Building on a non-trivial network representation, two mixed-integer programming formulations for the VRPTT are proposed. (ii) Based on these formulations, five different branch-and-cut algorithms are developed and implemented. (iii) The computational behaviour of the algorithms is analyzed in an extensive computational study, using a large number of test instances designed to resemble real-world VRPTTs.
Keywords: Branch-and-cut; Vehicle Routing; Synchronization; Trailer; Transshipment

1 Introduction

The vehicle routing problem with trailers and transshipments (VRPTT) was introduced by Drexl [11] in the context of raw milk collection at farmyards. It is a generalization of the classical vehicle routing problem (VRP, Dantzig and Ramser [9], Toth and Vigo [23], Golden et al. [14]). The version of the VRPTT studied in this paper can be described as follows. There is a set of customers with a given supply. To collect the supply, a limited fleet of heterogeneous vehicles (*lorries* and *trailers*) stationed at a central depot is available. Besides unequal fixed and variable costs and capacities, the vehicles differ in that lorries are *autonomous* and able to move in time and space on their own, whereas trailers are *non-autonomous* and can move in time on their own (that is, wait), but must be pulled by a lorry to move in space. In addition to the depot and the customer locations, there are so-called *transshipment locations* (TLs), where trailers can be parked and where load transfers from lorries to trailers can be performed.
Some customers can only be visited by a lorry without a trailer (a *single* lorry) and are hence called *lorry customers*. The other customers can be visited by a lorry with or without a trailer and are called *trailer customers*. There are time windows at the customers as well as at the TLs. All vehicles start and end their routes at the depot. There is no fixed assignment of a trailer to a lorry. Any trailer may be pulled, on the whole or on a part of its itinerary, by any *compatible* lorry. What is more, any lorry may transfer part or all of its load to any trailer at any TL. The time a transshipment takes depends on the amount of load transferred. Lorries need not bring back a trailer, neither one they may have pulled when leaving the depot, nor any other.
The problem is to determine routes for lorries *and routes for trailers* so that total costs are minimized, the complete supply of all customers is delivered to the depot, and loading capacities and time windows are maintained. Moreover, it must be ensured that the routes are synchronized

with respect to space, time and load, so that trailers move in space only when accompanied by compatible lorries and that the vehicles involved in a load transfer operation visit the pertinent location at the right time and transfer and receive the right amount of load. An example route plan is depicted in Figure 1.

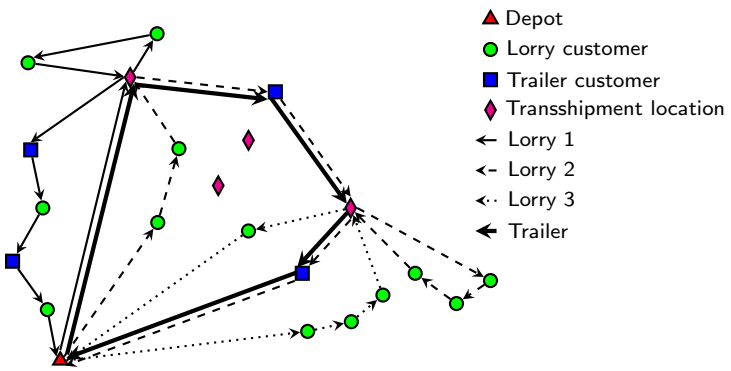

Figure 1: VRPTT example route plan

In the example, lorry 1 pulls the trailer from the depot to a TL and decouples the trailer there. Lorry 1 then visits two lorry customers, returns to the trailer, transfers load, leaves the trailer at the TL and returns to the depot via two lorry and two trailer customers. Lorry 2 starts at the depot and visits two lorry customers before coupling the trailer, after lorry 1 has performed its load transfer. Lorry 2 then visits a trailer customer, decouples the trailer at another TL, performs a load transfer, visits some lorry customers, returns to the trailer, re-couples it and pulls it back to the depot via a trailer customer. Lorry 3 starts at the depot, visits some lorry customers, and transfers some load to the trailer while lorry 2 is visiting the three rightmost lorry customers. After that, lorry 3 returns to the depot via another lorry customer. The two TLs in the centre of the figure are not used.

The justification for studying the VRPTT is that the problem, apart from being interesting and challenging in its own right, has numerous direct practical applications and, in addition, constitutes an archetypal, generic representative of the class of *vehicle routing problems with multiple synchronization constraints (VRPMSs)*. Contrary to classical VRPs, where a synchronization between vehicles is necessary only with respect to which vehicle visits which customer, VRPMSs exhibit additional synchronization requirements with regard to spacial, temporal, and load aspects. This is discussed in more detail in Section 2.

The contribution of the present paper is three-fold: (i) Building on a non-trivial network representation, two mixed-integer programming (MIP) formulations for the VRPTT are proposed. (ii) Based on these formulations, five different branch-and-cut algorithms are developed and implemented. (iii) The computational behaviour of the algorithms is analyzed in an extensive computational study, using a large number of test instances designed to resemble real-world VRPTTs.

The rest of the paper is structured as follows. The next section provides an overview of related work, including other types of VRPMSs. Section 3 presents a network representation for the VRPTT as a basis for the two arc-variable based formulations developed in Section 4. Section 5 describes the solution approaches that were implemented to solve these formulations, and Section 6 presents computational experiments performed with the implementations. The final Section 7 concludes the paper with a research outlook

2 Literature review

The available literature on the VRPTT and related VRPMSs is rather limited. As stated, the VRPTT was introduced in Drexl [11], where formulations based on arc, path, and so-called turn variables were developed and a branch-and-cut algorithm for an arc variable formulation

was presented. This author is not aware of other papers containing a solution approach for the VRPTT.

A quite well-studied problem dealing with trailers in vehicle routing is the *truck-and-trailer routing problem* (TTRP) (see, for example, Semet and Taillard [22], Chao [4], Scheuerer [21]). The TTRP is a special case of the VRPTT with a fixed lorry-trailer assignment, that is, each trailer can be pulled by only one lorry, and only this lorry can transfer load into the trailer. This makes the problem much easier. In fact, the TTRP is not a VRPMS; there are no synchronization requirements other than customer covering.

A recent survey of synchronization in vehicle routing is given in Drexl [12]. In this paper, five different types of synchronization are identified: (i) task synchronization, referring to the decision which vehicle(s) fulfil(s) which task(s); (ii) operation synchronization, which decides about time and location of some interaction between vehicles; (iii) movement synchronization, that is, the decision which vehicles join to move in space; (iv) load synchronization, which determines the amounts of load to be collected, delivered, or transshipped; and (v) resource synchronization, which decides about which vehicle(s) use(s) a scarce resource at what time.

The most important classes of VRPMSs identified in this survey are multi-echelon vehicle and location-routing problems with temporal synchronization of vehicles of different echelons (Crainic et al. [8]), simultaneous vehicle and driver *routing* problems (as opposed to integrated vehicle and crew *scheduling* problems, where the tasks to perform usually have a given fixed schedule) (Hollis et al. [15]), and pickup-and-delivery problems with transshipments and simultaneous load transfers (Cortés et al. [7]).

Works that study problems where different types of autonomous and non-autonomous objects that may join and separate en route (and not only at a central depot) are used to fulfil tasks are Bürckert et al. [3], Hollis et al. [15], Cheung et al. [5], Kim et al. [16], and Drexl et al. [13]. All of these papers develop heuristic solution approaches: Bürckert et al. [3] consider a dynamic pickup-and-delivery problem using lorries, trailers, and swap-body platforms in the context of long-haul transport and solve their problem with a holonic multi-agent system. Both Hollis et al. [15] and Drexl et al. [13] solve a simultaneous vehicle and driver *routing* problem (with application in postal logistics in Australia and long-haul road transport in Europe respectively) with two-stage algorithms. Hollis et al. [15] first compute abstract vehicle routes by solving a rich pickup-and-delivery problem by heuristic column generation. Then, concrete vehicles and drivers are assigned by taking an integrated vehicle and crew scheduling approach, again done by solving an MIP by heuristic column generation. Drexl et al. [13] determine routes for concrete vehicles in the first stage, and then compute routes for drivers, based on the vehicle routes from the first stage. They use the same large neighbourhood algorithm in both stages. Cheung et al. [5] also describe a pickup-and-delivery problem, but for short-distance container transport, using drivers, tractors, and semi-trailers. Their problem is solved by an attribute-decision model. Finally, Kim et al. [16] develop a simple but very effective heuristic for synchronizing service teams that are transported by vehicles between task locations: The teams, the vehicles, and the next tasks are stored in three lists, and in each iteration, a triplet (team, vehicle, task) is selected from the lists, using a best-fit criterion. Then the lists are updated to reflect the situation when the selected vehicle transports the selected team to the location of the selected task.

Note that all of these works assume *mandatory* synchronization of objects. This means that, to fulfil tasks, it is always required to use all or several object types (vehicles, drivers, equipment). In the VRPTT, by contrast, using trailers and performing transshipments are *optional* operations. This adds an additional degree of freedom to the problem. Moreover, none of the above papers considers a problem where all five types of synchronization described above are relevant, as is the case for the VRPTT.

3 A network representation

In contrast to standard vehicle routing problems, for which an adequate network representation is straightforward, this is not the case for the VRPTT, where the following critical modelling questions must be addressed:

- How to ensure that a trailer is accompanied by a compatible lorry on an arc, that is, how to synchronize the movements of vehicles?
- How to synchronize the visiting times of vehicles at transshipment locations?
- How to balance the load transfer amounts of vehicles exchanging load?

A decoupling, transshipment or recoupling operation is defined by (i) the *location* where the operation takes place, (ii) the *point in time* when the operation begins, (iii) the *passive vehicle* (trailer), which provides capacity, that is, may receive load, and/or may be decoupled from or (re)coupled to a lorry, (iv) the *active vehicle* (lorry), which requests capacity, that is, may transfer load, and/or may decouple or (re)couple a trailer, and (v) the *amount of load* transferred into the passive vehicle, which is zero if only decoupling or recoupling is performed. To answer the above modelling questions and handle the logic of transshipment operations, a *space-time-vehicle class-operation network* $D = (V, A)$ with an associated set F of vehicles (the *fleet*), is proposed.

The fleet F is partitioned into two sets $F_L \uplus F_T$ of *classes* of vehicles. F_L is the set of lorry classes, and F_T is the set of trailer classes. $|F^k|$ denotes the number of vehicles of class k. For all $k \in F$, q_k is the capacity of a vehicle of class k. Henceforth, k is used to denote lorry classes, and k' is used to denote trailer classes. For a lorry class $k \in F_L$, $C(k)$ is the set of *compatible* classes of trailers, that is, the set of classes of trailers that a lorry of class k can pull; for a trailer class $k' \in F_T$, it is the set of compatible classes of lorries, that is, the set of classes of lorries that can pull a trailer of class k'. τ^l is the load transfer time per unit of load, which is assumed to be the same for all vehicle classes. It is assumed for simplicity that any lorry can visit any customer, and that any trailer can visit any trailer customer.

Each vertex in V corresponds to a location in space, an absolute and/or relative period of time, a type of operation, and a class of passive vehicle. $L = \{Depot\} \uplus L_C \uplus L_I$ is the set of relevant real-world locations. $L_C = L_{C_L} \uplus L_{C_{LT}}$ is the set of customer locations, which is partitioned into L_{C_L}, the set of lorry customers, and $L_{C_{LT}}$, the set of trailer customers. L_I is the set of *pure* transshipment locations. For each $i \in V$, $L(i) \in L$ denotes the location corresponding to i. s_i is the supply of vertex i, which is zero for depot and transshipment vertices. For $S \subseteq V$, $s_S := \sum_{i \in S} s_i$. There is one start depot vertex o and one end depot vertex e. For each customer, there is one customer vertex. Let $V_C = V_{C_L} \uplus V_{C_{LT}}$ be the set of customer vertices, where V_{C_L} is the set of lorry customers, and where $V_{C_{LT}}$ is the set of trailer customers. Following Chao [4] and Scheuerer [21], it is assumed that the locations corresponding to trailer customers can also be used for parking and transshipment operations.

Each vertex $i \in V$ has a time window $[a_i, b_i]$, where $0 \leq a_i \leq b_i \leq T$, and T is the length of the planning horizon. o and e both have a time window of $[0, T]$. Usually, in the literature, a_i and b_i respectively denote the earliest and latest point in time when the service to be performed at i can start. In the VRPTT, a_i has the same meaning, but, because of the load-dependent load transfer times, b_i denotes the latest point in time when i must be *left*, that is, the service at i must be finished no later than b_i. To provide a unified treatment, this convention is adopted also for the depot and customer vertices.

For each combination of physical transshipment location (*pure* transshipment location or trailer customer location) and trailer class k', there are $|F^{k'}|$ vertex tuples representing the operations decoupling, transshipment, and recoupling. To be precise, for each transshipment location l and each trailer class k', there are $|F^{k'}|$ tuples $v^d_{k'lp}$, $v^t_{k'lp}$, $v^r_{k'lp}$, $p = 1, \ldots, |F^{k'}|$. V_d is the set of decoupling vertices, V_t is the set of transshipment vertices, and V_r is the set of recoupling vertices. $V_I = V_d \uplus V_t \uplus V_r$. The idea behind this separation of transshipment locations and processes is the following: A vertex $v^d_{k'lp} \in V_d$ can only be reached by a lorry pulling a trailer of

the corresponding class k'. The lorry then leaves the vertex singly, the trailer moves on to $v^t_{k'lp}$, the pertinent transshipment vertex. A vertex $v^t_{k'lp} \in V_t$ can only be reached and left by a single lorry and by a trailer of the corresponding class k', where the latter comes from $v^d_{k'lp}$. A vertex $v^r_{k'lp} \in V_r$ can only be reached by a single lorry and by a trailer of the corresponding class, where the latter comes from the pertinent transshipment vertex $v^t_{k'lp}$, and be left by the lorry pulling that trailer. At any vertex of a transshipment vertex tuple, that is, also at the decoupling and the recoupling vertex, a lorry visiting the vertex can transfer load to a trailer.
All vehicles are initially at the start depot vertex o and all vehicles that are used end their route at the end depot vertex e. Lorries can visit all vertices of D, except for decoupling and recoupling vertices of incompatible trailers. Trailers can only visit their corresponding transshipment vertices, trailer customers, and, of course, the start and the end depot vertex. $F(i)$, for all $i \in V_I$, is the class of passive vehicle associated with i. For $(i,j) \in A$, F^L_{ij} and F^T_{ij} respectively denote the set of lorry and trailer classes that can traverse (i,j). For all $k \in F$, V^k (A^k) is the set of vertices (arcs) that can be reached (traversed) by vehicles of class k. Moreover, for $S \subseteq V$, let $A^k(S) := \{(i,j) \in A^k : i,j \in S\}$, $\delta^-_k(S) := \{(i,j) \in A^k : i \notin S \ni j\}$, and $\delta^+_k(S) := \{(i,j) \in A^k : i \in S \not\ni j\}$.
The arc set A contains the following elements:

- (o,i) and (i,e) for all customer vertices $i \in V_C$
- (o,i) for all decoupling and recoupling vertices $i \in V_d \uplus V_r$
- (i,j) for all customer vertices $i,j \in V_C$ with $i \neq j$
- $(v^d_{k'lp}, j)$ for all decoupling vertices $v^d_{k'lp} \in V_d$ and all vertices $j \in V \setminus (\{o\} \uplus V_d)$
- $(v^t_{k'lp}, j)$ for all transshipment vertices $v^t_{k'lp} \in V_t$ and all other vertices $j \in V \setminus (\{o,e\} \uplus V_d)$
- $(v^r_{k'lp}, j)$ for all recoupling vertices $v^r_{k'lp} \in V_r$ and all other vertices $j \in V^{k'}_d \setminus \{v^d_{k'lp'} : p' \leq p\} \uplus V_{C_{LT}} \uplus \{e\}$
- (i,j) for all $i \in V_{C_{LT}}, j \in V_I$
- (i,j) for all $i \in V_{C_L}, j \in V_t \uplus V_r$

Figure 2 visualizes the vertex types present in the network and the arc types that can be traversed by lorry-trailer combinations, single lorries, and single trailers. To keep the figure concise, there is only one arc for each arc type present in the network. For example, in the subfigure for LTCs, an arc from the left trailer customer to the right one is depicted to indicate that LTCs can freely move from any trailer customer to any other trailer customer (unless capacity or time window restrictions prohibit this). The absence of an arc from the right trailer customer to the left one in the subfigure does not mean that there is no such arc. The arc from the recoupling to the decoupling vertex in the subfigure for LTCs exists only if the two transshipment vertex tuples (TVTs) represent the same trailer class, and if they lie at different locations or the upper TVT has a higher p value. The indicated arcs to, from, and between transshipment vertices in the subfigure for single lorries exist between all types of TVTs, that is, those representing the same or different trailers at the same or different locations. For TVTs representing the same location, the arcs exist only if the upper TVT has a higher p value.
A cost coefficient c^k_{ij} indicates the distance-dependent costs of a vehicle of class $k \in F$ traversing arc $(i,j) \in A^k$. For the arcs emanating from the start depot vertex, fixed costs for using a vehicle of class k are also contained in c^k_{oj}. It is assumed that all arc costs and arc travel times are non-negative. For each arc $(i,j) \in A$, τ_{ij} is the traversal time, which is assumed to be the same for all vehicle classes. Moreover, it is assumed that $\tau_{oi} \leq a_i$ for all $i \in V \setminus \{o\}$, and that $b_i + \tau_{ie} \leq T$ for all $i \in V \setminus \{e\}$.

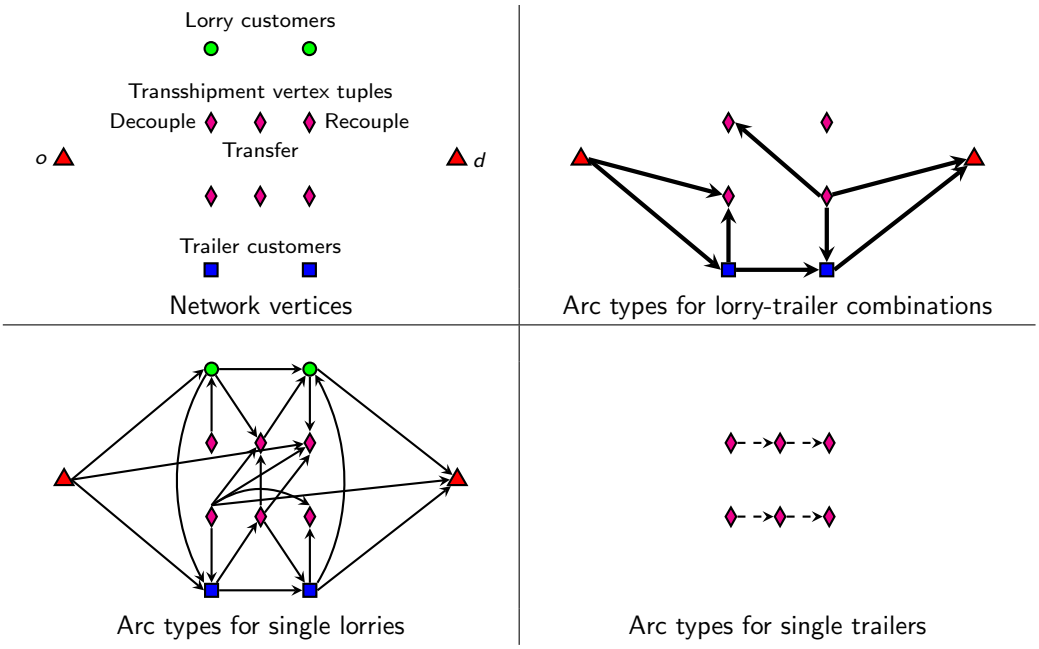

Figure 2: Network vertices and arc types

Lorries cannot traverse arcs $(v^d_{k'lp}, v^t_{k'lp})$ and $(v^t_{k'lp}, v^r_{k'lp})$. The corresponding trailers of class k' can traverse these arcs, which, consequently, form the set $A^{k',s}$. An arc $(v^d_{k'lp}, v^r_{k'lp})$ models the possibility that a lorry can wait for a trailer at a transshipment location while another lorry performs a load transfer. If a lorry currently pulling a trailer k' wants to transfer load to another trailer k'' at a certain location l, the lorry decouples k' at a decoupling vertex at l, transfers load to k'', and re-couples k'.

4 Two arc-variable based formulations

Both subsequent formulations are modifications of a formulation developed in Drexl [11]. For both formulations, the following three fundamental assumptions are made:

(i) *Each lorry customer vertex is visited by exactly one lorry.*

(ii) *Each trailer customer vertex is visited by exactly one lorry and at most one trailer.*

(iii) *Each transshipment vertex is reached by at most one lorry and at most one trailer.*

These assumptions ensure that there is at most one transshipment operation at each transshipment vertex, and that it is clear between which lorry and trailer classes each of these operations is performed. In this way, the three modelling issues mentioned at the beginning of Section 3 are addressed.
With the network defined as above, that is, with one transshipment vertex between a decoupling and a recoupling vertex, these assumptions allow at most $3|F^{k'}|$ load transfers per trailer class k' at each transshipment location. These load transfers can all be made into one and the same trailer of class k', if this trailer is pulled from $v^r_{k'lp}$ to $v^d_{k'l,p+1}$ for $p = 1, \ldots, |F^{k'}| - 1$, or into several or all trailers of class k'.

4.1 First formulation

The following variables are used:

- $x_{ij}^k \in \{0,1\} \; \forall \; k \in F, (i,j) \in A^k$.
  $x_{ij}^k = \begin{cases} 1, \text{a vehicle of class } k \text{ traverses arc } (i,j) \\ 0, \text{otherwise} \end{cases}$
- $l_i^k \in \mathbb{R}_+^0 \; \forall \; k \in F, i \in V^k \setminus \{o\}$.
  The amount of load a vehicle of class k is carrying when reaching vertex i.
- $t_i \in \mathbb{R}_+^0 \; \forall \; i \in V \setminus \{o\}$.
  The unique point in time when vertex i is reached.

The resulting formulation is:

(VRPTT1):

$$\sum_{k \in F} \sum_{(i,j) \in A^k} c_{ij}^k x_{ij}^k \rightarrow \min \tag{1}$$

subject to

$$\sum_{k \in F_L} \sum_{(h,i) \in A^k} x_{hi}^k = 1 \quad \forall \; i \in V_C \tag{2a}$$

$$\sum_{(h,i) \in A^{F(i)}} x_{hi}^{F(i)} \leq 1 \quad \forall \; i \in V_I \tag{2b}$$

$$\sum_{k \in F_L} \sum_{(h,i) \in A^k} x_{hi}^k = x_{ij}^{F(i)} \quad \forall \; i \in V_d, (i,j) \in A^{F(i)} \tag{2c}$$

$$\sum_{k \in F_L} \sum_{(h,i) \in A^k} x_{hi}^k \leq x_{ij}^{F(i)} \quad \forall \; i \in V_t, (i,j) \in A^{F(i)} \tag{2d}$$

$$\sum_{k \in F_L} \sum_{(h,i) \in A^k} x_{hi}^k = x_{h'i}^{F(i)} \quad \forall \; i \in V_r, (h',i) \in A^{F(i)} \tag{2e}$$

$$x_{ij}^{k'} \leq \sum_{k \in C(k') \cap F_{ij}^L} x_{ij}^k \qquad \forall \; k' \in F_T, i \in \{o\} \uplus V_{C_{LT}}, (i,j) \in A^{k'} \tag{2f}$$

$$x_{ij}^{F(i)} = \sum_{k \in C(k') \cap F_{ij}^L} x_{ij}^k \qquad \forall \; i \in V_r, (i,j) \in A^{F(i)} \tag{2g}$$

$$\sum_{(i,e) \in A^k} x_{ie}^k \leq |F^k| \quad \forall \; k \in F \tag{2h}$$

$$\sum_{(h,i) \in A^k} x_{hi}^k = \sum_{(i,j) \in A^k} x_{ij}^k \quad \forall \; k \in F, i \in V^k \setminus \{o,e\} \tag{2i}$$

$$x_{ij}^k = 1 \wedge x_{ij}^{k'} = 1 \Rightarrow l_i^k + l_i^{k'} + s_i \leq l_j^k + l_j^{k'} \quad \forall \; i \in V_{C_{LT}}, k \in F_L, k' \in C(k), (i,j) \in A^k \cap A^{k'} \tag{3a}$$

$$x_{ij'}^k = 1 \Rightarrow l_i^{F(i)} + \left(l_i^k - l_{j'}^k \right) \leq l_j^{F(i)} \quad \forall \; i \in V_d, k \in C(F(i)), (i,j) \in A^{F(i)}, (i,j') \in A^k \tag{3b}$$

$$x^k_{ij'} = 1 \Rightarrow l^{F(i)}_i + \left(l^k_i - l^k_{j'}\right) \leq l^{F(i)}_j \quad \forall\, i \in V_t, k \in F_L, (i,j) \in A^{F(i)}, (i,j') \in A^k, j' \neq e \tag{3c}$$

$$x^k_{ij} = 1 \Rightarrow l^{F(i)}_i + \left(l^k_i - l^k_j\right) \leq l^{F(i)}_j \quad \forall\, i \in V_r, k \in C(F(i)), (i,j) \in A^{F(i)} \cap A^k, j \neq e \tag{3d}$$

$$x^k_{ij} = 1 \Rightarrow l^k_i + s_i \leq l^k_j \quad \forall\, k \in F_L, i \in V_C, (i,j) \in A^k, F^T_{ij} = \emptyset \tag{3e}$$

$$x^k_{ij} = 1 \Rightarrow l^k_i \leq l^k_j \quad \forall\, k \in F, i \in V_{C_{LT}}, (i,j) \in A^k, F^T_{ij} \neq \emptyset \tag{3f}$$

$$x^k_{ij} = 1 \wedge \sum_{k' \in C(k) \cap F^T_{ij}} x^{k'}_{ij} = 0 \Rightarrow l^k_i + s_i \leq l^k_j \quad \forall\, k \in F_L, i \in V_{C_{LT}}, (i,j) \in A^k, F^T_{ij} \neq \emptyset \tag{3g}$$

$$x^k_{ij} = 1 \Rightarrow l^k_j \leq l^k_i \quad \forall\, k \in F_L, i \in V_I, (i,j) \in A^k \tag{3h}$$

$$x^{F(i)}_{ij} = 1 \Rightarrow t_i + \left(l^{F(i)}_j - l^{F(i)}_i\right)\tau^l \leq b_i \quad \forall\, i \in V_r, (i,j) \in A^{F(i)} \tag{3i}$$

$$x^k_{ij} = 1 \Rightarrow t_i + s_i\tau^l + \tau_{ij} \leq t_j \quad \forall\, k \in F_L, i \in V_C, (i,j) \in A^k \tag{3j}$$

$$x^k_{ij} = 1 \Rightarrow t_i + \left(l^k_i - l^k_j\right)\tau^l + \tau_{ij} \leq t_j \quad \forall\, k \in F_L, i \in V_I, (i,j) \in A^k \tag{3k}$$

$$l^{F(i)}_i \leq l^{F(i)}_j \quad \forall\, i \in V_t, (i,j) \in A^{F(i)} \tag{4a}$$

$$t_i + \left(l^{F(i)}_j - l^{F(i)}_i\right)\tau^l + \tau_{ij} \leq t_j \quad \forall\, i \in V_d \uplus V_t, (i,j) \in A^{F(i)} \tag{4b}$$

$$x^k_{ij} \in \{0,1\} \quad \forall\, k \in F, (i,j) \in A^k \tag{5a}$$

$$0 \leq l^k_i \leq q_k \quad \forall\, k \in F, i \in V^k \setminus \{o\} \tag{5b}$$

$$a_i \leq t_i \leq b_i - s_i\tau^l \quad \forall\, i \in V \setminus \{o\} \tag{5c}$$

(1) is the objective function, which minimizes total costs. Constraints (2) are those involving only flow variables. (2a)–(2g) are the routing synchronization constraints, and (2h)–(2i) are vehicle-class specific routing constraints. Constraints (3) are logical implications linking routing and resource variables. (3a)–(3d) are load synchronization constraints, (3e)–(3h) are vehicle-class specific load update constraints, and (3i)–(3k) are time update constraints. Constraints (4) are resource update constraints not involving flow variables, and (5) determine the ranges of the variables.

(2a) are the usual customer covering constraints. (2b) ensure that each transshipment vertex is visited at most once by a corresponding trailer. (2c)–(2e) make sure that a transshipment vertex i is only visited by a lorry if the corresponding trailer visits this vertex, and, hence, that at most one lorry visits a transshipment vertex. Observe that, for $i \in V_d$, the only arc in $A^{F(i)}$ leaving i is the arc to the subsequent transshipment vertex. Similarly, for $i \in V_t$, the only arc in $A^{F(i)}$ leaving i is the arc to the subsequent recoupling vertex. (2a)–(2e) imply that each vertex in $V \setminus \{o, e\}$ is visited by at most one lorry and one trailer. (2f) and (2g) guarantee that a trailer k' is pulled by a compatible lorry if k' traverses a spacial arc.

For the remaining constraints, note that a lorry that brings a trailer to a decoupling vertex and then moves on to the depot will not transfer any load at this vertex. Likewise, a lorry that pulls a trailer from a recoupling vertex to the depot will also not transfer any load. Moreover, a lorry will never visit a transshipment vertex $i \in V_t$ directly before moving to the depot; thus, as stated above, no arcs (i, e) with $i \in V_t$ exist.

For each vehicle class, (2h) limit the number of vehicles reaching the end depot to the number of vehicles of each class. (2i) are the flow conservation constraints.

(3a)–(3d) are the load update constraints for both lorries and trailers at trailer customer and transshipment vertices. (3a) state that the supply of a trailer customer is divided up arbitrarily between the lorry and the trailer visiting the customer. The next constraints, (3e), are the load update constraints for the lorries at customer vertices. Constraints (3f) state that, at a trailer customer vertex, no load transfer is possible. This is a sensible requirement, because, as stated above, for each trailer customer location, there are also tuples of transshipment vertices for each trailer class, and movements between vertices corresponding to the same physical location incur virtually no costs and take virtually no time. Constraints (3g) are for the correct update of the load variables at trailer customer vertices visited by a single lorry. Constraints (3h) make sure that no load transfer from a trailer to a lorry (passive to active vehicle) is possible, respectively, that the amount of load transferred from a lorry to a trailer is non-negative. Without this constraint, negative load transfer times can result. (3i) make sure that the load transfer at transshipment vertices is finished by the end of the time window. Note that, because of (3a)–(3d), it is sufficient to require (3i) for $F(i)$. Moreover, it is sufficient to require (3i) for $i \in V_r$, since then the time windows will be maintained also for the preceding transshipment vertices. Constraints (3j) and (3k) are for the timing update for lorries on arcs emanating from customer vertices and at transshipment vertices respectively.
(4a) make sure that the load of a trailer does not decrease at transshipment intermediate vertices not visited by a lorry, and (4b) are for the timing update of trailers at their transshipment vertices. The latter two constraint types are no implications, because these constraints can be fulfilled also when the arc $(i, j) \in A^{F(i)}$ for $i \in V_d \uplus V_t$ is not used.
Synchronization constraints. In standard VRPs such as the capacitated VRP (CVRP) or the VRP with time windows (VRPTW), the customer covering constraints (2a) are the only logically coupling (linking, joint) ones. In the VRPTT, the close interdependency between the vehicles must be dealt with by several additional types of logically coupling constraints, namely, (2b)–(2g) and (3a)–(3d).
'Inflating' resource variable values. At the end depot vertex, there is only one load variable, l_e^k, for all vehicles of each class $k \in F$. This means that if one vehicle of a class reaches the end depot fully loaded (according to the variable values), any other vehicle belonging to the same class does so also, even if this vehicle actually (in reality) carries less load. The constraints allow to 'inflate' load variables, meaning that the load variable of a vehicle class at the vertices on a tour may be set higher than the real load, while still not exceeding the real vehicle capacity. In this way, the *differences* between the values of the load variables along a tour reflect the real situation, so that the correct synchronization of load transfer amounts and visit times, which depends on differences in load variable values, remains possible. This technique of inflating the values of resource variables allows to avoid introducing routing and resource variables for each vehicle, and, when using variables for vehicle classes instead of individual vehicles, makes the creation of start and end depot vertices for each vehicle unnecessary.

4.2 Second formulation

The second formulation is based on the same network as the first formulation, and it uses the same type of binary variables. However, following an idea introduced independently by van Eijl [24] and Maffioli and Sciomachen [18], it uses different types of continuous resource variables:

- $l_{ij}^k \in \mathbb{R}_+^0 \ \forall\ k \in F, (i, j) \in A^k$.
  If the arc (i, j) is used by a vehicle of class k, the amount of load the vehicle is carrying when *leaving* vertex i; zero otherwise.

- $t_{ij} \in \mathbb{R}_+^0 \ \forall\ (i, j) \in A$.
  If the arc (i, j) is used by any vehicle, the unique point in time when vertex i is *left*; zero otherwise.

The resulting formulation is:

(VRPTT2):

(1) subject to (2) and

$$\sum_{(h,i)\in A}\left(\sum_{k\in F_{hi}^{L}}\left(l_{hi}^{k}+s_i x_{hi}^{k}\right)+\sum_{k'\in F_{hi}^{T}} l_{hi}^{k'}\right)\leq\sum_{(i,j)\in A}\left(\sum_{k\in F_{ij}^{L}} l_{ij}^{k}+\sum_{k'\in F_{ij}^{T}} l_{ij}^{k'}\right)\quad\forall\, i\in V_{C_{LT}} \tag{6a}$$

$$\sum_{(h,i)\in A}\left(\sum_{k\in C(F(i))} l_{hi}^{k}+l_{hi}^{F(i)}\right)\leq\sum_{k\in C(F(i))}\sum_{(i,j')\in A^{k}} l_{ij'}^{k}+\sum_{(i,j)\in A^{F(i)}} l_{ij}^{F(i)}\quad\forall\, i\in V_{d} \tag{6b}$$

$$\sum_{k\in F_L}\sum_{(h,i)\in A^{k}} l_{hi}^{k}+\sum_{(h',i)\in A^{F(i)}} l_{h'i}^{F(i)}\leq\sum_{k\in F_L}\sum_{(i,j')\in A^{k}} l_{ij'}^{k}+\sum_{(i,j)\in A^{F(i)}} l_{ij}^{F(i)}\quad\forall\, i\in V_{t} \tag{6c}$$

$$\sum_{k\in C(F(i))}\sum_{(h,i)\in A^{k}} l_{hi}^{k}+\sum_{(h',i)\in A^{F(i)}} l_{h'i}^{F(i)}\leq\sum_{(i,j)\in A}\left(\sum_{k\in C(F(i))\cap F_{ij}^{L}} l_{ij}^{k}+l_{ij}^{F(i)}\right)\quad\forall\, i\in V_{r} \tag{6d}$$

$$\sum_{k\in F_L}\sum_{(h,i)\in A^{k}}\left(l_{hi}^{k}+s_i x_{hi}^{k}\right)\leq\sum_{k\in F_L}\sum_{(i,j)\in A^{k}} l_{ij}^{k}\quad\forall\, i\in V_{C_L} \tag{6e}$$

$$\sum_{k'\in F_T}\sum_{(h,i)\in A^{k'}} l_{hi}^{k'}\leq\sum_{k'\in F_T}\sum_{(i,j)\in A^{k'}} l_{ij}^{k'}\quad\forall\, i\in V_{C_{LT}} \tag{6f}$$

$$\sum_{k\in F_L}\sum_{(h,i)\in A^{k}}\left(l_{hi}^{k}+s_i x_{hi}^{k}\right)-s_i\sum_{k'\in F_T}\sum_{(h',i)\in A^{k'}} x_{h'i}^{k'}\leq\sum_{k\in F_L}\sum_{(i,j)\in A^{k}} l_{ij}^{k}\quad\forall\, i\in V_{C_{LT}} \tag{6g}$$

$$\sum_{k\in F_L}\sum_{(i,j)\in A^{k}} l_{ij}^{k}\leq\sum_{k\in F_L}\sum_{(h,i)\in A^{k}} l_{hi}^{k}\quad\forall\, i\in V_{I} \tag{6h}$$

$$\sum_{(h,i)\in A^{F(i)}} l_{hi}^{F(i)}\leq\sum_{(i,j)\in A^{F(i)}} l_{ij}^{F(i)}\quad\forall\, i\in V_{t} \tag{6i}$$

$$\sum_{(h,i)\in A}\left(t_{hi}+\tau_{hi}\sum_{k\in F_{hi}^{L}} x_{hi}^{k}\right)+s_i\tau^{l}\leq\sum_{(i,j)\in A} t_{ij}\quad\forall\, i\in V_{C} \tag{6j}$$

$$\sum_{(h,i)\in A^{F(i)}}\left(t_{hi}+\tau_{hi}x_{hi}^{F(i)}\right)+\left(\sum_{(i,j)\in A^{F(i)}} l_{ij}^{F(i)}-\sum_{(h,i)\in A^{F(i)}} l_{hi}^{F(i)}\right)\tau^{l}\leq\sum_{(i,j)\in A^{F(i)}} t_{ij}\quad\forall\, i\in V_{I} \tag{6k}$$

$$\sum_{(i,j)\in A^{F(i)}} t_{ij}=\sum_{(i,j')\in A\setminus A^{F(i)}} t_{ij'}\quad\forall\, i\in V_{d} \tag{6l}$$

$$\sum_{(i,j)\in A^{F(i)}} t_{ij}-b_i\left(1-\sum_{(i,j')\in A\setminus A^{F(i)}}\sum_{k\in F_{ij'}^{L}} x_{ij'}^{k}\right)\leq\sum_{(i,j')\in A\setminus A^{F(i)}} t_{ij'}\quad\forall\, i\in V_{t} \tag{6m}$$

$$\sum_{(h,i)\in A\setminus A^{F(i)}}\left(t_{hi}+\tau_{hi}\sum_{k\in F_{hi}^{L}} x_{hi}^{k}\right)$$

$$+\left(\sum_{(h,i)\in A^k\setminus A^{F(i)}}\sum_{k\in F^L_{hi}} l^k_{hi} - \sum_{(i,j')\in A^k\setminus A^{F(i)}}\sum_{k\in F^L_{ij'}} l^k_{ij'}\right)\tau^l \le \sum_{(i,j)\in A^{F(i)}} t_{ij} \quad \forall\, i\in V_t \tag{6n}$$

$$\sum_{(h,i)\in A\setminus A^{F(i)}}\left(t_{hi}+\tau_{hi}\sum_{k\in F^L_{hi}} x^k_{hi}\right)+\left(\sum_{(h,i)\in A}\sum_{k\in F^L_{hi}} l^k_{hi} - \sum_{(i,j)\in A}\sum_{k\in F^L_{ij}} l^k_{ij}\right)\tau^l \le \sum_{(i,j)\in A} t_{ij} \quad \forall\, i\in V_r \tag{6o}$$

$$x^k_{ij}\in\{0,1\} \quad \forall\, k\in F,(i,j)\in A^k \tag{7a}$$

$$0\le l^k_{ij}\le q_k x^k_{ij} \quad \forall\, k\in F,(i,j)\in A^k \tag{7b}$$

$$\left(a_i+s_i\tau^l\right)\sum_{k\in F^L_{ij}} x^k_{ij}\le t_{ij}\le b_i\sum_{k\in F^L_{ij}} x^k_{ij} \quad \forall\,(i,j)\in A, i\in\{o\}\uplus V_C \tag{7c}$$

$$a_i\sum_{k\in F^L_{ij}} x^k_{ij}\le t_{ij}\le b_i\sum_{k\in F^L_{ij}} x^k_{ij} \quad \forall\, i\in V_I,(i,j)\in A\setminus A^{F(i),s} \tag{7d}$$

$$a_i x^{F(i)}_{ij}\le t_{ij}\le b_i x^{F(i)}_{ij} \quad \forall\, i\in V_d\uplus V_t,(i,j)\in A^{F(i)} \tag{7e}$$

$$t_{ie}\le b_e-\tau_{ie} \quad \forall\,(i,e)\in A \tag{7f}$$

Constraints (6a)–(6i) and (6j)–(6n) are for the update of the load and time variables respectively; constraints (7) specify the ranges of the variables.
Constraints (6a)–(6h) correspond to (3a)–(3h) in formulation (VRPTT1). Similarly, (6i) corresponds to (4a), and (6j)–(6k) correspond to (3j)–(3k). (6k)–(6o) ensure the correct update of the timing variables at decoupling, transshipment intermediate, and recoupling vertices respectively. The corresponding constraint to (3i), which makes sure that the load transfer at transshipment vertices is finished by the end of the time window, is (7c).
(7b)–(7e) couple the flow and resource variables. These constraints allow to avoid implications in formulation (VRPTT2).

4.3 Valid inequalities

The following valid inequalities were used in the computational experiments described in the next section.

Lorry flow cut:

$$\sum_{k\in F_L}\sum_{(i,e)\in A^k} x^k_{ie}\ge n^{L,min} \tag{8}$$

This inequality requires that the flow of lorries into the end depot be at least equal to the minimum number of necessary lorries, $n^{L,min}$. As the total customer supply must be brought to the depot, the inequality is valid. A lower bound on $n^{L,min}$ can be determined as follows. All possible lorry-trailer combinations and single lorries are sorted by non-increasing total capacity and the capacities are summed up until the sum exceeds the total customer supply. For each summand, $n^{L,min}$ is increased by one. As trailers are usually compatible with more than one lorry, all lorry-trailer combinations that are no longer possible because the respective lorry or trailer or both has/have already been used with another lorry or trailer must be deleted during the summation. A similar cut is possible for trailers, but was not useful in the computational experiments described below.

Connectivity cuts:

$$\sum_{(i',j')\in\delta_k^-(S),i'\neq j} x^k_{i'j'} \geq x^k_{ij} \quad \forall\, k\in F, S\subseteq V^k\setminus\{e\}, (i,j)\in A^k, i\in S \tag{9}$$

These cuts require that, for each vehicle class and for each subset of the vertex set not containing the end depot vertex, if a vehicle of this class uses an arc (i,j) whose tail is in the subset, there must be another arc (i',j') entering this subset whose tail must not be j. This is implied by the flow conservation constraints (2h) and (2i).
An exact procedure for the separation of violated connectivity cuts works as follows. For each $x^k_{ij} > 0$, a maximum flow problem from j to e is solved in the support graph where the arc capacities are equal to the values of the flow variables of vehicle class k, but where vertex i, and, hence, all arcs incident to i, are deleted. This yields a cut with j on one side and e on the other. If the value of the maximum flow is less than x^k_{ij}, a violated connectivity cut has been found.
This procedure is correct, because in any feasible solution, $x^k_{ij} > 0$ means $x^k_{ij} = 1$, and i is on the same side as j in any j-e-cut, because each vertex is visited by each vehicle class at most once. In other words, there is no feasible solution where i is on the unique path of a vehicle of class k from j to e, if arc (i,j) is used by a vehicle of class k. This means that there must be a flow of at least x^k_{ij} from the j side of the cut to the e side of the cut, but this flow must not pass through i.
Cuts (9) can still be lifted. If one of the incident vertices i, j of an arc (i,j) is a recoupling vertex, the corresponding decoupling and intermediate vertices cannot be visited any more, and, hence, any potential cut arc emanating from or leading to such a vertex can be discarded. If i or j is an intermediate vertex, the same holds for arcs incident to the corresponding decoupling vertex.

κ-path cuts:
κ-path cuts are well-known valid inequalities for the VRP(TW), cf. Kohl et al. [17]. In the VRPTW with homogeneous fleet (in particular, without trailers), the idea of κ-path cuts for $\kappa \geq 1$ is that, for any customer subset S for which at least $\kappa(S)$ vehicles are necessary to serve all customers in the subset, at least κ vehicles must visit this subset, and, consequently, the flow into this subset must be at least κ.

2-path cuts for heterogeneous fleet:
Kohl et al. [17] have very successfully used 2-path cuts in a branch-and-price-and-cut algorithm for the VRPTW. For vehicle routing problems with heterogeneous fleet as well as for the VRPTT, 2-path cuts can be generalized as follows. For each subset $S \subseteq V_C$, let $\kappa_k(S)$ be a lower bound on the minimal number of routes of vehicles of class $k \in F$ necessary to collect the complete supply of the customers in S when no transshipments are allowed and no vehicle of a different class is used (that is, when it is assumed that each vehicle class can visit each customer and that trailers can visit customers without being pulled by a lorry). For example, if there are no time windows, the bound $\lceil s_S/q_k \rceil$ can be used. Then, the following inequality is valid:

$$\sum_{\{k\in F:\kappa_k(S)\geq 2\}} \sum_{(i,j)\in\delta_k^-(S)} x^k_{ij} + 2\sum_{\{k\in F:\kappa_k(S)=1\}} \sum_{(i,j)\in\delta_k^-(S)} x^k_{ij} \geq 2 \quad \forall\, S\subseteq V_C. \tag{10}$$

The validity of (10) follows from the customer covering constraints (2a), the load update constraints (3a)–(3h), and the vehicle capacity constraints (5b).

κ-path cuts for heterogeneous fleet for general $\kappa \geq 2$:
Setting $\kappa := \min_{k\in F}\{\kappa_k(S)\}$ for all $S \subseteq V_C$, a κ-path cut for general $\kappa \geq 2$ can be written as follows:

$$\sum_{k\in F}\sum_{(i,j)\in\delta_k^-(S)} x^k_{ij} \geq \kappa \quad \forall\, S\subseteq V_C \tag{11}$$

Note that (10) and (11) require summation over all $k \in F$, that is, over all lorry *and* trailer classes, and that (10) as well as (11) are valid for sets S containing an arbitrary number of lorry and/or trailer customers.

Lifted generalized large multistar cuts:
Yaman [25] has shown that the following cuts are valid for heterogeneous fleet VRPs:

$$\sum_{k \in F} \sum_{(i,j) \in \delta_k^-(S)} \xi_i^k x_{ij}^k \geq s_S + \sum_{k \in F} \sum_{(i,j) \in \delta_k^+(S)} s_j x_{ij}^k \quad \forall\, S \subseteq V_C, \tag{12}$$

where $\xi_i^k := \min\{q_k - s_i, s_S + \max_{(i,j) \in \delta_k^+(S)}\{s_j\}\}$. Essentially, this type of cut requires that the remaining capacity of the vehicles visiting customers in S be at least equal to the supply of the customers in S plus the supply of those customers (if any) visited immediately after leaving S.

Symmetry-breaking cuts:
All trailer routes visiting the same locations in the same order and receiving the same amount of load at each location are equivalent, irrespective of which tuple of transshipment vertices is used at a certain location. That is, two otherwise identical trailer routes r_1 and r_2, where r_1 uses the transshipment vertices $v^d_{k'lp_1}$, $v^t_{k'lp_1}$, $v^r_{k'lp_1}$ and r_2 uses $v^d_{k'lp_2}$, $v^t_{k'lp_2}$, $v^r_{k'lp_2}$, are equivalent and incur the same costs, irrespective of whether or not $p_1 = p_2$, since they describe the same movements in reality. Such symmetries can be removed by the following cuts:

$$\sum_{k \in F_L} x^k_{hi'} \leq \sum_{k \in F_L} x^k_{hi} \quad \forall\, i, i' \in V_d, L(i) = L(i'), F(i) = F(i'), p(i) = p(i') - 1 \tag{13}$$

These cuts require that a transshipment vertex tuple with order number p is only visited if all preceding transshipment vertex tuples, that is, those with lower order number, are also visited. If two trailers k'_1 and k'_2 of the same class use the same transshipment location l, similar symmetries arise: It does not make a difference whether k'_1 uses the transshipment vertex tuple with order number p_1 and k'_2 uses the tuple with order number p_2 or the other way round. To remove such symmetries, the following cuts can be used in formulation (VRPTT1):

$$t_i \leq t_{i'} \quad \forall\, i, i' \in V_d, L(i) = L(i'), F(i) = F(i'), p(i) = p(i') - 1 \tag{14}$$

These cuts require that the service at the decoupling vertex of a transshipment vertex tuple with order number p starts not earlier than the service at the decoupling vertex of the transshipment vertex tuple with order number $p - 1$. For formulation (VRPTT2), the following cuts can be used instead of (14):

$$\sum_{(i,j) \in A^{F(i)}} t_{ij} - (b_i + 1) x_{ij}^{F(i)} \leq \sum_{(i',j') \in A^{F(i')}} t_{i'j'} - (b_{i'} + 1) x_{i'j'}^{F(i')}$$

$$\forall\, i, i' \in V_d, L(i) = L(i'), F(i) = F(i'), p(i) = p(i') - 1 \tag{15}$$

Some remarks on valid inequalities follow. The lorry flow, connectivity, and 2-path cuts were already used in Drexl [11]. Baldacci et al. [2] propose a lifting for the κ-path cuts of Kohl et al. [17], but this cannot be used for arc variable formulations. Desaulniers et al. [10] have introduced generalized κ-path cuts for the VRPTW with homogeneous fleet. Although these can be generalized to heterogeneous fleet, they were not used in the computational experiments described below, because the wide time windows in the test instances used essentially made these cuts equivalent to cuts (11). Ascheuer et al. [1] have introduced, for the asymmetric TSP with time windows, the so-called infeasible path constraints. These cuts are valid also for the VRPTT, but are not useful when time windows are non-restrictive. Finally, Pessoa et al. [19] describe cuts for a heterogeneous fleet VRP, but these are based on a different formulation.

5 Solution approaches

Formulation (VRPTT1), the 'original' formulation, contains a lot of logical constraints (implications). To deal with these, the following five solution approaches were implemented in C++, using Visual C++ 2010 and IBM Ilog Cplex Concert Technology, version 12.2:

(i) Formulation (VRPTT1), *linearizing the implications using tailored Big-M values*, taking into account the vehicle capacities and the length of the planning horizon.

(ii) Formulation (VRPTT2), thereby *avoiding logical constraints altogether.*

(iii) Formulation (VRPTT1), *employing a Cplex feature that allows entering implications directly as such* and letting the solver automatically handle them. How this is done exactly, though, is not documented.

(iv) Formulation (VRPTT1), *applying the combinatorial Benders' cuts* introduced by Codato and Fischetti [6] and successfully used by Cortés et al. [7]. The basic idea of this approach is to decompose the problem into a master problem containing only the x variables and the constraints involving only these variables, and a subproblem containing only the continuous variables and all constraints not involving x variables. Then, after solving the LP relaxation of the master problem, all implications activated by x variables that are one or zero in the solution of the LP relaxation are added to the subproblem. If the subproblem, which has no objective function, is feasible, then evidently the x variables solution to the master problem is feasible and optimal for the complete problem. Otherwise, at least one of the integer x variables must change its value. This is enforced by identifying an inclusion-minimal infeasible subset (by means of a built-in Cplex function) of the subproblem constraints and adding the constraint

$$\sum_{kij \in S_1} (1 - x^k_{ij}) + \sum_{kij \in S_0} x^k_{ij} \geq 1 \tag{16}$$

to the master, where the set S_1 (S_0) contains those variables that activate a constraint when they take value one (zero) and do take this value in the current LP solution.

(v) Formulation (VRPTT1), *using deferred consideration of logical constraints.* This works as follows. At the root node of the branch-and-bound tree, the LP relaxation of a relaxed problem is solved, where all implications are ignored. Then, at each subsequent node, it is checked for which x variables the lower bound is greater than zero. All such variables will have value one in each integer feasible solution (if any) at the subtree rooted at this node. Similarly, it is checked which x variables have an upper bound of less than one. All such variables will have value zero in each integer feasible solution at the subtree rooted at this node. Then, instead of having linearized constraints containing Big-M or adding combinatorial Benders' cuts, only the right hand sides of the activated implications, that is, the respective linear constraints to the right of '$\Rightarrow$' in (3), are added as locally valid constraints to the LP at the respective node.

All approaches use the cuts described in Section 4.3. For formulation (VRPTT2), cuts (15) are used instead of (14). The lorry flow cut as well as both types of symmetry-breaking cuts are directly added to the formulation as static cuts. The connectivity cuts are dynamically separated in the course of the algorithm. As for the κ-path and multistar cuts, it turned out that all instances that could be solved with or without them are small enough to allow complete enumeration of all cuts ex ante. Therefore, all cuts (11) and (12) are also added as static cuts.

As branching strategy, the following four-stage strategy outperformed the default best-bound single variable branching strategy provided by Cplex for approaches (i), (ii), and (iv).

(i) Branch on the sum of all arc variables leaving the start depot vertex.

(ii) Branch on the sum of all lorry class arc variables leaving the start depot vertex.

(iii) Branch on the sum of all trailer class arc variables leaving the start depot vertex.

(iv) Use the default Cplex branching strategy.

For approaches (iii) and (v), the Cplex default branching performed better. Strong branching was used in all cases.

6 Computational experiments

In this section, the results of an extensive computational study with the approaches described in the previous section are presented.

6.1 Test instances

Drexl [11] has created a set of VRPTT test instances structured to resemble the situation in raw milk collection. These use two lorry and two trailer classes as specified in Table 1.

	Capacity	Fixed costs	Costs per km	Compatible trailer classes
Lorry class 1	10,000	180	0.65	1, 2
Lorry class 2	15,000	200	0.70	1
Trailer class 1	10,000	20	0.04	
Trailer class 2	15,000	25	0.04	

Table 1: Fleet data

The customer and transshipment locations are randomly selected on a 100×100 km grid with the depot located in the centre. The resulting Euclidean distances between each pair of vertices are multiplied by a distance factor of 1.3. The customer supplies are chosen randomly from [1,000; 10,000]. As load transfer time, two minutes per 1,000 units of supply are assumed.
The length of the planning horizon is 12 hours (1,320 minutes). All customers and transshipment locations have a time window of $[0; 1{,}320]$. Such non-restrictive time windows represent the situation in raw milk collection where only very few customers or transshipment locations actually have time windows.
The test instances were created with enough vehicles of each class such that the complete supply could be collected with either only lorries of class 1 and trailers of class 2 or only lorries of class 2 and trailers of class 1.
With these parameters, a set of so-called x_y_z instances was created, where $2 \leq x \leq 20$ stands for the number of customers, $2 \leq y \leq 20$ for the number of potential transshipment locations, and $4 \leq z \leq 24$ for the number of vehicles. It is always $x = y$, and there are always $x/2$ lorry customers and $x/2$ trailer customers. $y/2$ potential transshipment locations correspond to the trailer customer locations, and $y/2$ are pure transshipment locations. For each of the four vehicle classes, there are $z/4$ vehicles. This means that an x_y_z instance has $1+x+3\cdot\sum_{k' \in F_T} |F^{k'}| \cdot y+1$ vertices: one for the start depot, x for the customers, three for each trailer at each of the y potential transshipment locations to represent decoupling, transfer, and recoupling, and one for the end depot. Table 2 shows details on instance sizes.

Instance class	No. of instances	Vertices	Arcs	Variables			Constraints		Cuts		
				Binary	Continuous				Symmetry-	κ-path	Multistar
					(VRPTT1)	(VRPTT2)	(VRPTT1)	(VRPTT2)	breaking 1 / 2	(avg.)	
2_2_4	30	16	126	222	62	348	619	634	0 / 0	0	0
4_4_4	20	30	484	856	114	1,340	2,384	2,176	0 / 0	2	10
4_4_8	10	54	1,516	2,600	202	4,116	7,408	6,344	8 / 8	7	10
6_6_4	7	44	1,074	1,902	166	2,976	5,299	4,630	0 / 0	13	56
6_6_8	23	80	3,390	5,826	298	9,216	16,597	13,822	12 / 12	37	56
8_8_8	19	106	6,008	10,336	394	16,344	29,444	24,172	16 / 16	188	246

Table 2: Instance sizes

6.2 Algorithm setup and system parameters

Some remarks on the setup of the algorithms are appropriate. In approach (iv), combinatorial Benders' cuts can either be separated only when the LP solution is integer or also when some variables are fractional. Moreover, it is possible to add the cuts as global or local ones, and either only one or several violated cuts can be added in each iteration. Tests showed that the

best setup is to separate cuts also when there are fractional variables, and to add one violated cut per iteration as a global one. Similarly, in approach (v), it is possible to check for newly fixed variables either only when the LP solution is integer in the x variables or after each solution of the LP relaxation. Here, tests showed that the former variant is slightly but consistently better. In all approaches, connectivity cuts are added only if the violation is at least 0.3. In each iteration, all connectivity cuts that reach this violation threshold are added. Tests with adding only one violated connectivity cut per vehicle class and iteration yielded a worse performance. The maximum flow problems for the separation of the connectivity cuts are solved with the Edmonds-Karp algorithm provided by the Boost Graph library (`boost.org`).
The results presented subsequently were obtained on a computer with a 2.80 GHz CPU and 16 GB of main memory for instance classes 2_2_4–6_6_8, and on a machine with a 2.83 GHz CPU and 8 GB of main memory for instance class 8_8_8, both running Win XP Sp 2, 64-bit. A CPU time limit of 10,800 seconds was set.

6.3 Evaluation of the valid inequalities

Table 3 reports the average relative increase in the root lower bound for formulations (VRPTT1) and (VRPTT2) when the different classes of valid inequalities are used. As can be seen, the root lower bounds of formulation (VRPTT2) are significantly higher than those of (VRPTT1). Moreover, the table shows that the lorry flow cut is very important: It cuts off fractional solutions where lorries are used only partially, and the resulting better consideration of the fixed costs raises the lower bounds considerably.

Formulation:	(VRPTT1)	(VRPTT2)	(VRPTT2)
Average relative increase compared to:	(VRPTT1), no cuts	(VRPTT1), no cuts	(VRPTT2), no cuts
No cuts	n.a.	320.3 %	n.a.
Only lorry flow cut	403.2 %	480.0 %	36.8 %
Only connectivity cuts	67.8 %	336.4 %	3.3 %
Only κ-path cuts	7.6 %	322.2 %	0.4 %
Only multistar cuts	17.7 %	321.8 %	0.3 %

Table 3: Comparison of root lower bounds (instances 4_4_4–8_8_8)

6.4 Comparison of the different solution approaches

Table 4 presents the computational results obtained with the different approaches and the described settings. Columns 'Optimal' and 'Feasible' respectively indicate the percentage of instances that could be solved optimally and for which a feasible solution could be computed within 10,800 seconds. Column 'Gap' specifies the average relative gap between the upper and the lower bound, UB and LB respectively, upon termination of the optimization, for those instances for which a feasible solution could be computed, as $(UB - LB)/LB$. Column 'CPU time' indicates the average running time over all instances of a class, whether or not a feasible solution could be found. Column 'None better' reports the percentage of instances that no approach solved better. No approach solves an instance i better than a certain approach A if either A solves i to optimality and no other approach that also solves i optimally is faster, or if no approach finds the optimal solution within the time limit and no approach finds a better feasible solution than A. Column 'Best' indicates the percentage of instances that were solved best with the respective approach. An approach A is best in solving an instance i if A finds the optimal solution faster than any other approach, or if it finds a better solution than any other approach. To account for measuring inaccuracies, computation times in seconds were rounded to integer multiples of 10 seconds when computing the values in columns 'None better' and 'Best'. Finally, column 'No. B & B nodes' gives the average number of nodes in the branch-and-bound tree upon termination of the optimization, irrespective of the solution status.

Approach	Instance class	Optimal [%]	Feasible [%]	Gap [%]	CPU time [secs.]	None better [%]	Best [%]	No. B & B nodes
(i): (VRPTT1), linearizing the implications using tailored Big-M values	2_2_4	100.0	100.0	0.0	0.3	100.0	0.0	4
	4_4_4	100.0	100.0	0.0	20.3	25.0	0.0	228
	4_4_8	90.0	100.0	1.0	1,405.4	0.0	0.0	2,662
	6_6_4	100.0	100.0	0.0	523.7	0.0	0.0	1,421
	6_6_8	30.4	100.0	4.8	8,551.8	26.1	4.3	3,747
	8_8_8	0.0	89.5	40.6	10,800.1	5.3	5.3	1,221
	Overall	67.0	98.2	7.6	3,853.5	38.5	1.8	1,382
(ii): (VRPTT2), avoiding logical constraints altogether	2_2_4	100.0	100.0	0.0	0.9	100.0	0.0	5
	4_4_4	100.0	100.0	0.0	17.4	15.0	0.0	90
	4_4_8	100.0	100.0	0.0	124.8	50.0	50.0	121
	6_6_4	100.0	100.0	0.0	566.5	0.0	0.0	588
	6_6_8	13.0	100.0	7.0	9,811.4	65.2	30.4	1,429
	8_8_8	0.0	89.5	21.5	10,800.1	26.3	26.3	574
	Overall	64.2	98.2	4.9	4,004.1	53.2	15.6	468
(iii): (VRPTT1), employing a Cplex feature that allows entering implications directly	2_2_4	100.0	100.0	0.0	2.9	86.7	0.0	400
	4_4_4	60.0	100.0	14.8	4,832.6	5.0	0.0	28,682
	4_4_8	50.0	50.0	0.0	7,118.5	0.0	0.0	28,302
	6_6_4	–	–	–	–	–	–	–
	6_6_8	–	–	–	–	–	–	–
	8_8_8	–	–	–	–	–	–	–
	Overall	43.1	50.5	5.4	2,798.7	24.8	0.0	14,478
(iv): (VRPTT1), applying the combinatorial Benders' cuts	2_2_4	100.0	100.0	0.0	0.3	100.0	0.0	5
	4_4_4	100.0	100.0	0.0	8.2	85.0	5.0	170
	4_4_8	90.0	100.0	2.2	1,197.4	10.0	0.0	1,595
	6_6_4	100.0	100.0	0.0	197.7	57.1	42.9	1,137
	6_6_8	34.8	100.0	4.7	7,530.2	34.8	8.7	6,570
	8_8_8	0.0	89.5	22.6	10,800.1	10.5	10.5	2,060
	Overall	67.9	98.2	4.8	3,595.6	56.9	7.3	1,997
(v): (VRPTT1), using deferred consideration of logical constraints	2_2_4	100.0	100.0	0.0	0.2	100.0	0.0	44
	4_4_4	100.0	100.0	0.0	5.9	95.0	10.0	1,435
	4_4_8	90.0	100.0	1.7	1,226.9	50.0	40.0	23,742
	6_6_4	100.0	100.0	0.0	154.3	57.1	42.9	10,192
	6_6_8	34.8	100.0	4.7	7,598.3	52.2	30.4	91,686
	8_8_8	5.3	73.7	26.8	10,496.7	15.8	15.8	10,642
	Overall	68.8	95.4	4.8	3,556.6	67.0	17.4	24,310

Table 4: Computational results

The following observations can be made in Table 4:

- Overall, there is no approach that outperforms all others with respect to every criterion. However, approach (v) performs best in four out of seven criteria, and, for one additional criterion, there is no better one.
- Approach (iii) clearly performs worst. Instance classes 6_6_4–8_8_8 were not even tried with this approach. It seems that the automatic handling of implications built into Cplex is inadequate for the problem structure at hand.
- Although formulation (VRPTT2) has a significantly stronger LP relaxation than formulation (VRPTT1), approach (ii) solves fewer instances to optimality than approach (i), and it requires slightly more computation time on average. Apparently, the solution of the LP relaxations takes longer.
- Approach (v) solves the highest number of instances to optimality, 75 out of 109, which corresponds to 68.8 %. It is the only approach to solve an instance of class 8_8_8 to optimality. However, it fails to find a feasible solution for five instances, which is worse than approaches (i), (ii), and (iv). There is one particularly hard instance in class 4_4_8, which gets solved optimally only by approach (ii).
- The overall optimality gap is comparatively high for approach (i). In particular, though feasible solutions are found for all but two instances of class 8_8_8, the gap for this class is almost twice as high as for the other approaches.

- With respect to computation time, approaches (iv) and (v) are, on average, faster than the other approaches, although the speed-up obtained by these decompositions is less than expected.
- For more than two thirds of all instances, no approach was better than approach (v), according to the above definition of 'better'. Moreover, approach (v) performed best on more than 17 % of all instances.
- The number of branch-and-bound nodes is smallest for approach (ii), but apparently, the LP relaxation takes considerably longer to solve. Approach (v) creates by far the highest number of branch-and-bound nodes, but even the largest tree, which has some 500,000 nodes, fits easily into main memory.

As mentioned, the largest instance that could be solved to optimality belongs to class 8_8_8, that is, it has eight customers, eight transshipment locations, and eight vehicles. Due to the long computation times for the instances of this class, larger instances were not tackled. However, note that, as can be seen in Table 2, an 8_8_8 instance has more than 10,000 binary variables. In this respect, it corresponds to a CVRP instance with 100 customers (assuming a two-index arc variable formulation), which is quite large. The results also compare well with those obtained by Cortés et al. [7], who studied a pickup-and-delivery problem with transfer option (see Section 2), and who solved to optimality instances with at most six pickup-and-delivery requests, one transshipment location, and two vehicles.

7 Research outlook

Several enhancements and extensions to the models and algorithms presented in this paper are possible. From an algorithmic point of view, the following research avenues can be identified:
Better lower bounds. The computational experiments have shown that for many instances, the weak lower bounds prevent a (faster) solution. To improve these bounds, a detailed study of valid inequalities for the VRPTT, that is, for a time-constrained VRP with heterogeneous fleet, transshipment possibilities and split collection between lorries and trailers, would be interesting. In particular, problem-specific infeasible path cuts could be helpful.
Column generation. Another way to obtain better lower bounds is to use a path formulation and apply column generation. However, doing so is a highly involved project for the VRPTT, because the resulting pricing problems become very difficult and cannot be solved by standard dynamic programming algorithms. It is beyond the scope of this paper to go into details on this issue; the reader is referred to Drexl [11].
Reduced number of arc variables. Rieck and Zimmermann [20] have presented a method to reduce the number of binary variables for heterogeneous fleet VRPs. Their approach requires introducing even more implications and complicates the modelling of compatibility constraints. Nevertheless, since the number of binary variables strongly influences the computational difficulty of a problem, considering the method also for the VRPTT may be worthwhile.
Heuristics. Solving real-world instances of VRPs to optimality is still impossible. This is of course also true for the VRPTT. Therefore, a heuristic procedure for the VRPTT is needed, but still missing. However, the close interdependency between the vehicles and the fact that load transfers are optional considerably complicate the use of classical local or large neighbourhood search methods.
There are also several interesting model extensions to the VRPTT version presented in this paper:
Increase the number of allowed load transfers. This can easily be done by inserting a sequence $v^{t,1}_{k'lp}, v^{t,2}_{k'lp}, \ldots, v^{t,n_t}_{k'lp}$ of n_t transshipment vertices between each pair of decoupling and recoupling vertices and linking them via arcs $(v^d_{k'lp}, v^{t,1}_{k'lp})$, $(v^{t,1}_{k'lp}, v^{t,2}_{k'lp}), \ldots, (v^{t,n_t}_{k'lp}, v^r_{k'lp})$ that can only be traversed by a single trailer of class k'.
Allow load transfers from trailers to trailers and from lorries and trailers to lorries. To this end, it is sufficient to introduce transshipment vertex tuples also for lorries. To represent a load transfer

in the network, for example, from a trailer k' to a lorry k_1 at location l, the trailer decouples the lorry currently pulling it, say, k_2, at the decoupling vertex of one of the transshipment vertex tuples of k_2's class at location l, moves to a transshipment intermediate vertex of k_1's class, and recouples k_2.

Consider support vehicles. A third potential extension is the introduction of support vehicles, lorries as well as trailers. These cannot visit customers, but serve as mobile depots into which the other vehicles, which may be called *task vehicles*, can transfer load. This is sensible, for example, when visiting a customer requires costly technical equipment, so that support vehicles are much cheaper to operate, or when small task vehicles are needed to visit customers due to lack of manoeuvring space, and large support vehicles can be used to transport the customer supply to the depot. The consideration of support vehicles is particularly easy: It is sufficient to simply create *no* support vehicle flow variables for arcs leading to or emanating from customer vertices.

These extensions are practically relevant and very easy to accommodate in the network as well as in the formulations presented above. When doing so, however, the size of both the network and the formulations quickly becomes intractable. Techniques that allow to contain this size have yet to be developed.

Summing up, it can be said that the VRPTT is a challenging and practically relevant problem that needs and deserves further study. Research on the VRPTT will also be helpful to obtain insights for the solution of other types of VRPs with multiple synchronization constraints.

Acknowledgement. This research was funded by the Deutsche Forschungsgemeinschaft (DFG) under grant no. IR 122/5-1.

References

[1] Ascheuer N, Fischetti M, Grötschel M (2000):
A Polyhedral Study of the Asymmetric Traveling Salesman Problem with Time Windows
Networks 36: 69–79

[2] Baldacci R, Mingozzi A, Roberti R (2012):
Recent Exact Algorithms for Solving the Vehicle Routing Problem under Capacity and Time Window Constraints
European Journal of Operational Research 218: 1–6

[3] Bürckert H, Fischer K, Vierke G (2000):
Holonic Transport Scheduling with TELETRUCK
Applied Artificial Intelligence 14: 697–725

[4] Chao I (2002):
A Tabu Search Method for the Truck and Trailer Routing Problem
Computers & Operations Research 29: 33–51

[5] Cheung R, Shi N, Powell W, Simão H (2008):
An Attribute-Decision Model for Cross-Border Drayage Problem
Transportation Research Part E 44: 217–234

[6] Codato G, Fischetti M (2006):
Combinatorial Benders' Cuts for Mixed-Integer Linear Programming
Operations Research 54: 756–766

[7] Cortés C, Matamala M, Contardo C (2010):
The Pickup and Delivery Problem with Transfers: Formulation and a Branch-and-Cut Solution Method
European Journal of Operational Research 200: 711–724

[8] Crainic T, Ricciardi N, Storchi G (2009):
Models for Evaluating and Planning City Logistics Systems
Transportation Science 43: 432–454

[9] Dantzig G, Ramser J (1959):
The Truck Dispatching Problem
Management Science 6: 80–91

[10] Desaulniers G, Lessard F, Hadjar A (2008):
Tabu Search, Partial Elementarity, and Generalized k-Path Inequalities for the Vehicle Routing Problem with Time Windows
Transportation Science 42: 387–404

[11] Drexl M (2007):
On Some Generalized Routing Problems
PhD Thesis, Faculty of Business and Economics, RWTH Aachen University
URL `http://darwin.bth.rwth-aachen.de/opus3/volltexte/2007/2091/pdf/Drexl_Michael.pdf`

[12] Drexl M (2012):
Synchronization in Vehicle Routing—A Survey of VRPs with Multiple Synchronization Constraints
Transportation Science 46: 297–316

[13] Drexl M, Rieck J, Sigl T, Berning B (2011):
Simultaneous Vehicle and Crew Routing and Scheduling for Partial and Full Load Long-Distance Road Transport
Technical Report 1112, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz

[14] Golden B, Raghavan S, Wasil E (eds) (2008):
The Vehicle Routing Problem: Latest Advances and New Challenges
Operations Research/Computer Science Interfaces Series 43
Springer, Berlin

[15] Hollis B, Forbes M, Douglas B (2006):
Vehicle Routing and Crew Scheduling for Metropolitan Mail Distribution at Australia Post
European Journal of Operational Research 173: 133–150

[16] Kim B, Koo J, Park J (2010):
The Combined Manpower-Vehicle Routing Problem for Multi-Staged Services
Expert Systems with Applications 37: 8424–8431

[17] Kohl N, Desrosiers J, Madsen O, Solomon M, Soumis F (1999):
2-Path Cuts for the Vehicle Routing Problem with Time Windows
Transportation Science 33: 101–116

[18] Maffioli F, Sciomachen A (1997):
A Mixed-Integer Model for Solving Ordering Problems with Side Constraints
Annals of Operations Research 69: 277–297

[19] Pessoa A, Uchoa E, Poggi de Aragão M (2009):
A Robust Branch-Cut-and-Price Algorithm for the Heterogeneous Fleet Vehicle Routing Problem
Networks 54: 167–177

[20] Rieck J, Zimmermann J (2010):
A New Mixed Integer Linear Model for a Rich Vehicle Routing Problem with Docking Constraints
Annals of Operations Research 181: 337–358

[21] Scheuerer S (2006):
A Tabu Search Heuristic for the Truck and Trailer Routing Problem
Computers & Operations Research 33: 894–909

[22] Semet F, Taillard E (1993):
Solving Real-Life Vehicle Routing Problems Efficiently Using Tabu Search
Annals of Operations Research 41: 469–488

[23] Toth P, Vigo D (eds) (2002):
The Vehicle Routing Problem
SIAM Monographs on Discrete Mathematics and Applications, Philadelphia

[24] van Eijl C (1995):
A Polyhedral Approach to the Delivery Man Problem
Technical Report 95-19, Department of Mathematics and Computer Science, Eindhoven University of Technology

[25] Yaman H (2006):
Formulations and Valid Inequalities for the Heterogeneous Vehicle Routing Problem
Mathematical Programming 106: 365–390