Computers & Operations Research 40 (2013) 536–546

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Truck and trailer routing—Problems, heuristics and computational experience

Ulrich Derigs*, Markus Pullmann, Ulrich Vogel

Universität zu Köln, Pohligstr. 1, 50969 Köln, Germany

A R T I C L E I N F O

Available online 27 August 2012

Keywords:
Truck and trailer routing problem
Load transfer
Time windows
Metaheuristics
Computational study

A B S T R A C T

In the truck and trailer routing problem (TTRP) the vehicle fleet consists of truck units and trailer units with some customers only accessible by truck. For that purpose trailers can be uncoupled en-route at customers where truck sub-tours are built. We discuss several variants of this specific rich vehicle routing problem (RVRP): the TTRP with and without the option of load transfer between truck and trailer as well as the requirement of time windows for delivery. We present computational experience with a simple and flexible hybrid approach which is based on local search and large neighborhood search as well as standard metaheuristic control strategies. This approach which has shown to be rather effective on several other RVRP-classes before can compete with complex state-of-the-art approaches with respect to speed and accuracy on the TTRP too.

1. Introduction

In the classical vehicle routing problem (VRP), first studied by Dantzig and Ramser [7], we want to determine a set of routes with minimal total cost for a fleet of homogeneous vehicles to supply goods to customers which have a known demand. Here each route has to start and end at a common depot, and the demand of every customer has to be fulfilled by a single delivery respecting the capacity of the vehicles. Due to the fact that the VRP is NP-hard, it is usually solved by heuristic approaches, see for example Laporte et al. [23] or Cordeau et al. [6]. Over the years numerous variants of this classical VRP have been introduced in the literature, as for instance the VRP with Time Windows (VRPTW) where routes have to fulfill the additional constraints that the delivery at each customer has to be within a specific time window or the problem where pick-up and delivery operations have to be combined etc. This development, which was certainly guided by the fact that such specific enrichments of the classical model by additional features are required in practice, has led to the coining of the field of rich vehicle routing problems (RVRP). A good survey on the development of the VRP-field is given in Toth and Vigo [37].

Chao [3] introduced the truck and trailer routing problem (TTRP). Here a vehicle is composed of a truck and a trailer, and we distinguish two types of customers: truck customers which can only be served by the truck without trailer and vehicle customers which can also be served by the complete unit. Now a vehicle can start a route from the depot with or without a coupled trailer. In the former case it is necessary that before visiting a truck customer the trailer is uncoupled on the route and left at a vehicle customer which forms the so-called root for a truck-only sub-tour. Note that the trailer has to be picked up by the truck before returning to the depot, i.e. it is not allowed to exchange trailers between trucks on the route. In the model of Chao [3] and in the benchmark instances which he has published, the number of available trucks and trailers is limited, though there is no primary objective to minimize the number of vehicles.

The literature on the applications of TTRP is quite scarce up to now. Gerdessen [17] described two real-world applications in the Dutch dairy industry. Here the vehicles have to distribute dairy products to customers located in cities with heavy traffic and limited parking space. Maneuvering a vehicle in this environment is difficult. Therefore the trailer gets uncoupled at parking places outside the cities and the truck continues the delivery of customers in the city without trailer. Hoff [22] studied a real-world TTRP which occurred at a Norwegian dairy company. This company collects raw milk from different farmers. As in the example before several farms are inaccessible for complete vehicles, and while the truck collects the milk the trailer is parked at a parking place. Then after returning to the trailer the collected milk can be filled from the tank of the truck to the tank of the trailer.

Note that in the model of Chao [3] goods can be transferred on a route from the truck to the trailer or vice versa. This is possible if such a reloading does not require specific devices which are not on board. The examples by Gerdessen [17] and Hoff [22] describe applications where such a transfer of goods between truck and trailer is possible because the goods are rather light or fluid. In our research we have also treated truck and trailer routing

* Corresponding author. Tel.: +49 221 470 5327; fax: +49 221 470 5329.
E-mail address: ulrich.derigs@uni-koeln.de (U. Derigs).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.08.007

problems without load transfer where such a transfer is not possible. Here the placement of goods into the truck or the trailer has to be decided during order picking and loading.

The earliest heuristics for the TTRP are based on neighborhood search: Chao [3] and Scheuerer [32] present tabu search approaches based on neighborhoods which contain standard VRP-moves as well as a new TTRP-specific root-refining move where a complete truck sub-tour is attached to a different customer.

Lin et al. [25] have developed a specific indirect search approach (see Gottlieb [21]) for solving the TTRP which outperforms these earlier developments significantly, especially with respect to running time. The search in the auxiliary search space is guided by a simulated annealing control and uses three rather simple string operators.

Recently Villegas et al. [38] have presented a rather sophisticated hybrid metaheuristic which is the current state-of-the-art method for the TTRP. This approach combines different concepts such as the greedy randomized adaptive search procedure (GRASP) using a route-first cluster-second construction heuristic, variable neighborhood search (VNS) in the improvement phase with standard VRP-moves as well as a root-refining move, and path relinking with respect to a pool of elite solutions.

All these heuristics treat the TTRP with load transfer. To the best of our knowledge there are no approaches devoted to the TTRP without load transfer. Note that the construction of feasible solutions is a problem in itself and the approaches respond to this problem with different concepts: Chao [3] applies the generalized assignment approach for vehicle routing problems of Fisher and Jaikumar [16], while Scheuerer [32] adapted the classical sweep approach by Gillett and Miller [18]. Lin et al. [26] studied the effect of relaxing the truck and trailer availability constraint and could observe that (on the benchmark instances) the trailer/truck ratio for the relaxed problem is closer to 1 than for the TTRP.

Recently, Lin et al. [27] have introduced the truck and trailer routing problem with time windows (TTRPTW) where customers only accept being served during a specific time period. They extended their approach by having the decoder check that the service of the customer starts during the time window. As we will demonstrate in the sequel this straightforward extension is not exploiting the full complexity of the problem. Finally, Drexl [13,14] presented the generalized truck and trailer routing problem (GTTRP) which is a rather complex generalization motivated by a real-world scenario. Here trucks and trailers can be either collection vehicles or support vehicles. Collection vehicles are used to collect the supplies of customers, while support vehicles are used as “mobile depots” that cannot visit customers. Also a trailer may be pulled by different trucks during the course of its tour, using the equipment of a collection truck load may be transferred from any vehicle to any other vehicle during a tour, and any intermediate locations en-route can be used either for parking or for load transfer.

According to Cordeau et al. [6] a good VRP heuristic should fulfill four criteria: accuracy, speed, simplicity, and flexibility. While simplicity refers to the characteristic to which extent the basic principle of a heuristic is easy to be understood and coded, and whether it does not contain too many (obscure) parameters which have to be configured, a flexible heuristic should be able to accommodate the various side constraints encountered in real-life applications. In this paper we apply a heuristic approach to the TTRP which combines local search and large neighborhood search and which can be applied to all problem variants mentioned above by only slightly modifying the implementation of the moves as well as choosing the start heuristic and the metaheuristic control appropriately. This hybrid approach is simple and flexible in the above sense and it has shown to be rather effective on other VRP-variants like the split delivery vehicle routing problem [10], the open vehicle routing problem [11], or the vehicle routing problem with compartments [8] where we could show the approach to be highly competitive, i.e producing solutions which are competitive with the best known solutions published in the literature or even creating new best solutions. Vogel [39] has developed a VRP-software framework based on this heuristic concept and shown how solvers for specific rich VRP can be customized in a rather simple and flexible manner. A survey on the concept and design of the framework as well as computational experience showing the competitiveness with respect to speed and accuracy is presented in Derigs and Vogel [12].

In this paper we show how this approach can be applied to the TTRP. The paper is structured as follows. In Section 2 we describe the TTRP formally introducing the specific terminology. In Section 3 we introduce the algorithmic approach and in Section 4 we report computational results on benchmark instances from literature before closing with final remarks in Section 5.

2. Problem formulations

In the TTRP a fleet of m_k homogeneous trucks and m_l homogeneous trailers with $m_k \geq m_l$ serves a set of customers $C=\{1,2,\ldots,n\}$ from a depot, denoted by 0. Each truck has a capacity of Q_k and each trailer has a capacity of Q_l and each customer $i \in C$ has a positive demand q_i. In case of the TTRPTW each customer i has a time window $[ET_i,LT_i]$, during which the service has to start, and a service time ST_i. The time window of the depot is $[0,H]$, with H denoting the length of the planning horizon, thus all vehicles must be back within the planning period. The distance $c_{i,j}$ for $i,j \in C_0 := C \cup \{0\}$ are assumed to be non-negative, symmetric and to satisfy the triangle inequality. We distinguish two types of customers:

- *vehicle customers* can be served by a truck with coupled trailer (vehicle) or by a single truck while
- *truck customers* can only be served by trucks without coupled trailer.

Chao [3] distinguishes three different types of routes in a TTRP-solution:

- A *pure truck route* (PTR) is a route, where a truck serves all customers without using a trailer. Thus a customer on the route may be a truck customer or a vehicle customer.
- A *pure vehicle route* (PVR) is a route, where all customers are served by a truck with a coupled trailer. Thus only vehicle customers are served and the trailer is never uncoupled.
- A *complete vehicle route* (CVR) is a route on which at least once the trailer is uncoupled from the truck at a vehicle customer and the truck continues serving a subset of customers on a truck sub-tour. After serving this subset of customers the truck returns to the vehicle customer where the trailer is parked, and either it starts another truck sub-tour or the trailer gets coupled again and the vehicle continues the route.

Now every route, r say, can be partitioned into

- its *main-tour* $t_m(r)$ which is the sequence of customers served by the complete vehicle, and,
- a set $S(r)$ of *truck sub-tours* where the trailer is parked at a customer of the main-tour, the so-called *root* of the truck sub-tour, or left at the depot.

Thus a PTR consists of exactly one truck sub-tour rooted at the depot and a PVR consists of exactly one main-tour, while a CVR consist of exactly one main-tour and at least one truck sub-tour which is rooted at a customer of the main-tour. Fig. 1 exemplifies the three types of tours in a TTRP solution.

Note that in the case of a TTRP with load transfer it is possible to transfer goods from the trailer to the truck after serving a truck sub-tour when the truck returns to the vehicle customer where the trailer was left.

Now let $q(t_m(r))$ be the demand of all customers on the main-tour $t_m(r)$ of route r and $q(t_s)$ the demand of all customers served on a truck sub-tour $t_s \in S(r)$, then $q(r) := q(t_m(r)) + \sum_{t_s \in S(r)} q(t_s)$ gives the total demand on route r. Now the following two conditions must hold for a route to be feasible for TTRP with load transfer

$$q(r) \le Q_k + Q_l \tag{1}$$

$$q(t_s) \le Q_k \quad \forall t_s \in S(r) \tag{2}$$

If load transfer is not allowed then all goods for customers served on a truck sub-tour must be already loaded onto the truck at the depot. Note that it is possible to put goods for customers who are served on the main-tour into the truck as well. Let o be the amount of goods which is loaded onto the truck and used for serving vehicle customers on the main-tour. Then the following two additional conditions must hold for a route to be feasible in the case of a TTRP without load transfer

$$\sum_{t_s \in S(r)} q(t_s) + o \le Q_k \tag{3}$$

$$q(t_m(r)) - o \le Q_l \tag{4}$$

A set R of feasible routes is a feasible solution for the TTRP if the demand of all customers is fulfilled on exactly one route in R. In case of the TTRPTW routes must additionally respect time windows to be feasible. Here, if a vehicle/truck arrives at a customer i before ET_i then the vehicle has to wait until ET_i before serving can start. Note that the vehicle continues to the next customer as soon as the service at the current customer ends, therefore waiting times only occur before a service starts. Now, the objective is to find a feasible solution R with minimal total tour length.

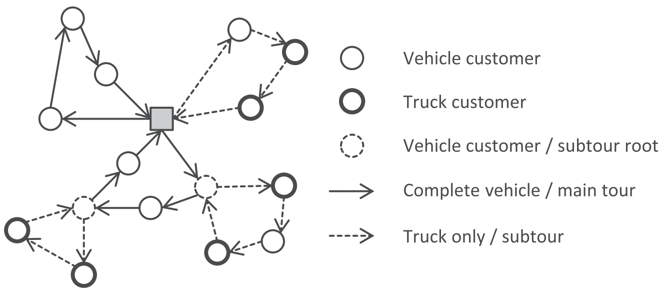


Fig. 1. Tour types in the TTRP.

3. The heuristic framework

Our heuristic for solving the TTRP-variants is a two-phase approach starting with constructing an initial feasible solution using a problem-specific strategy followed by a neighborhood search improvement phase where several standard as well as new problem-specific local search (LS) as well as large neighborhood search (LNS) moves are applied guided by two simple metaheuristic controls.

The purpose of a construction heuristic is to generate a feasible initial solution of acceptable quality in a rather short time for an improvement method. Since our improvement heuristics do not handle infeasibilities being able to produce feasible solutions is crucial. Due to the different complexities with respect to the TTRP-feasibility constraints we have implemented different construction heuristics for the different problem variants: the savings heuristic and the sweep heuristic for the TTRP with load transfer, the generalized assignment heuristic for the TTRP without load transfer as well as an insertion-based heuristic that uses regret insertion to construct an initial solution from scratch by inserting all customers into an empty solution for the TTRPTW. These are described in separate subsections of Section 4 together with other customizations of the general framework and computational results. The other building blocks are described in the following.

3.1. Neighborhood search

3.1.1. Local search

Traditional *local search* (LS) is based on moves which apply only small modifications to a solution. Thus the strength of LS is *intensification* of the search. We have implemented a set $\mathcal{N}$ of moves containing the standard operators *2-opt*, *2-opt**, *exchange* and *relocate* (cf. [24,29,31]), which are applied in many VRP implementations, as well as two TTRP-specific moves. To illustrate the complexity of these moves for TTRP Fig. 2 shows several cases which can occur during an exchange move.

Enabling the transfer of complete sub-tours the *relocate-sub-tour* neighborhood considers selecting an alternative root customer for a sub-tour, potentially within another main-tour, and at the same time selecting another customer within the sub-tour

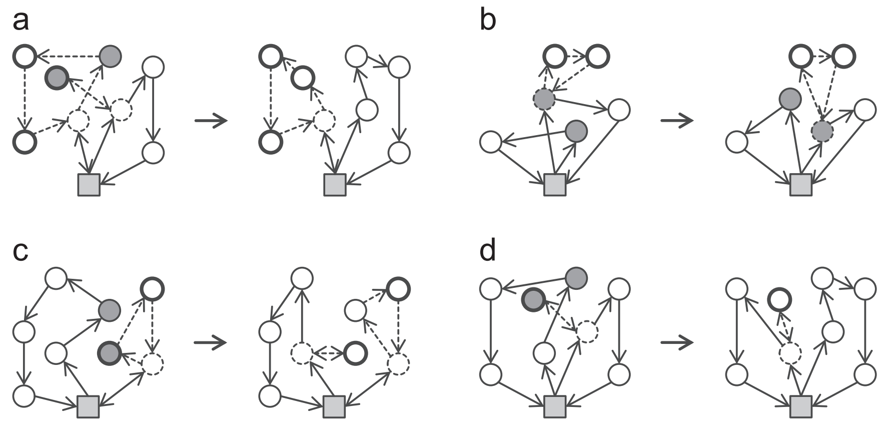


Fig. 2. Special exchange moves in the TTRP. (a) Replacing the single customer of a sub-tour with a vehicle customer, (b) replacing a sub-tour root with another vehicle customer, (c) replacing a main-tour customer with a truck customer and (d) combination of cases (a) and (c).

which is visited first. The relocate-sub-tour neighborhood is an extension of the *sub-tour root-refining step* by Chao [3], which is designed as an intra-tour move. Fig. 3 shows the relocation of a sub-tour from one main-tour to another with simultaneously selecting another starting customer.

The *switch-vehicle-type* neighborhood, denoted as *change of service vehicle type* in Lin et al. [25], changes the visit of a customer from a complete vehicle visit to a truck-only visit and vice versa. Fig. 4 shows the different cases which can occur.

3.1.2. Large neighborhood search

The neighborhood concept of *ruin-and-recreate* which copes for *diversification* of the search has been introduced by Schrimpf et al. [33]. In VRP the ruining step is done by removing q customers from their tours, i.e. not serving them temporarily, and recreating is done by reinserting these customers again into the solution with a (cheapest) insertion heuristic. In the *large neighborhood search* (LNS) heuristic proposed by Shaw [34,35] the ruin step is based on the random selection of customers which are "similar" with respect to their distances. Large neighborhood search based on combining different removal and insertion strategies was popularized by Ropke and Pisinger [30] and Pisinger and Ropke [28].

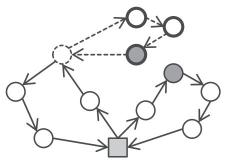


Fig. 3. Relocate-sub-tour neighborhood.

For the large neighborhood search module of our framework we have implemented the set of standard removal heuristics $\mathcal{R} := \{\text{random removal, worst removal, Shaw removal}\}$ and insertion heuristics $\mathcal{I} := \{\text{greedy insertion, regret}-2\ \text{insertion, regret}-3\ \text{insertion,regret}-4\ \text{insertion}\}$ which have been described in Ropke and Pisinger [30] and which we have customized for TTRP as well as a TTRP-specific removal operator allowing to remove sub-tours completely and thus improving the chance that new sub-tours can be created and potentially attached to other tours. Fig. 5 displays four special cases which may occur for the TTRP after removal and insertion, respectively.

Shaw removal is based on the concept of relatedness between two customers and has to be adapted in a problem-specific way. For TTRP we define customers to be related based on three attributes: geographical position, demand level, and customer type. Let c_{max} be the largest distance between two customers and q_{max} be the highest demand of all customers. For $i,j \in C$ let $y_{i,j}$ be 0 if both customers are of the same type, 1 else. Then the relatedness $\Gamma(i,j)$ is defined as follows

$$\Gamma(i,j) := f_c \frac{c_{i,j}}{c_{max}} + f_q \frac{|q_i - q_j|}{q_{max}} + f_t y_{i,j} \tag{5}$$

For TTRPTW we use time windows and service times as additional attributes and $\Gamma(i,j)$ is defined as follows

$$\Gamma(i,j) := f_c \frac{c_{i,j}}{c_{max}} + f_q \frac{|q_i - q_j|}{q_{max}} + f_t y_{i,j}$$

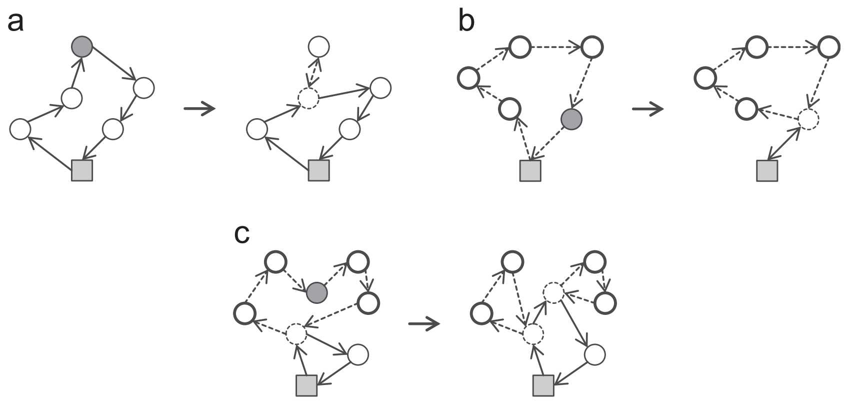


Fig. 4. Switch-vehicle-type neighborhood. (a) Moving a main-tour customer to a new sub-tour, (b) converting a pure truck tour customer into a sub-tour root in a complete vehicle tour and (c) moving a sub-tour customer to the main-tour and splitting the sub-tour.

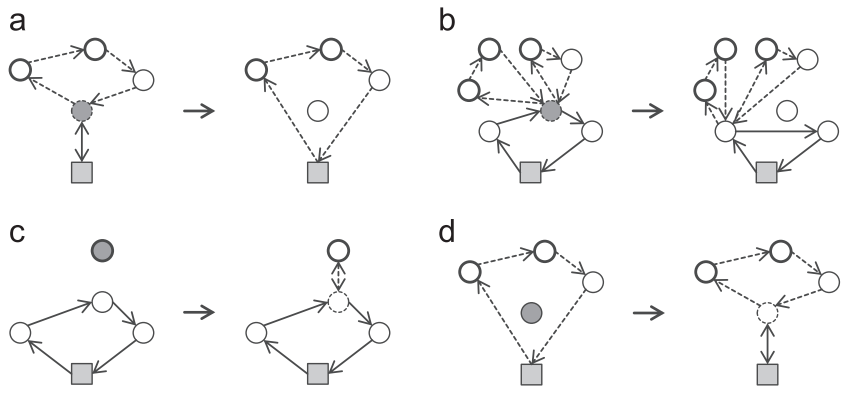


Fig. 5. Special removal and insertion operations in the TTRP. (a) Removing a sub-tour root which is the only customer within its main-tour, (b) removing a sub-tour root which is *not* the only customer within its main-tour, (c) inserting a truck customer by opening a new sub-tour and (d) inserting a vehicle customer by converting a pure truck tour into a complete vehicle tour.

$$+f_w\left(\frac{|ET_i-ET_j|+|LT_i-LT_j|+|ST_i-ST_j|}{t_{max}}\right) \tag{6}$$

Here t_{max} is denoting the largest difference of the time windows and service times of two customers, calculated as follows

$$t_{max}=\max_{i,j\in V}(|ET_i-ET_j|+|LT_i-LT_j|+|ST_i-ST_j|) \tag{7}$$

and f_c, f_q, f_t, f_w are parameters measuring the importance of the attributes.

Algorithm 1 shows the procedure of the TTRP-specific move *sub-tour removal* which enforces the rebuilding of sub-tours from scratch by removing a set of sub-tours from one area of the solution. Removing sub-tours completely (and not a few sub-tour customers only) improves the chance that new sub-tours are created with new root customers, potentially in other tours than before. Set P maintains the sub-tours of the solution which have not yet been selected for removal during the course of the procedure, and set R holds the customers selected to be removed on the other hand. In this context $i\in p$ denotes a customer of a sub-tour $p\in P$, and $|p|$ is the number of customers within the sub-tour. After selecting a random sub-tour as a seed further sub-tours are selected iteratively; since complete sub-tours are removed $|R|$ may exceed q in the end. In each iteration the selection of a sub-tour is biased towards such sub-tours which are near the sub-tours already selected. For each unselected sub-tour $p\in P$ the average distance from its customers to the customers in R is calculated as

$$\overline{d}_p := \frac{\sum_{i\in p, j\in R} c_{i,j}}{|R|\cdot|p|}$$

and based on these distance values a sub-tour p' is selected using a randomized selection mechanism controlled by a randomization parameter β^{STR}.

Algorithm 1. Sub-tour removal.

```
1:  function SubtourRemoval (solution $S$, $q\in\mathbb{N}$, randomization $\beta^{STR}$)
2:    $P :=$ sub−tours of $S$
3:    select sub-tour $p\in P$ randomly, $P := P\backslash\{p\}$
4:    $R :=$ customers of $p$
5:    while $|R| < q$ do
6:      $\overline{d}_p :=$ average distance between customers in $R$ and $p$, $\forall p\in P$
7:      array $P' := \langle p\in P\rangle$, with $k<l \Rightarrow \overline{d}_{P'[k]} \le \overline{d}_{P'[l]}$
8:      select $r\in[0,1)$ randomly
9:      $p' := P'[\lfloor r^{\beta^{STR}}\cdot|P'|\rfloor]$
10:     $R := R\cup$ customers of $p'$
11:     $P := P\backslash\{p'\}$
12:   end while
13:   remove customers $R$ from $S$
14: end function
```

3.2. Metaheuristic controls

The purpose of a metaheuristic is to guide the search and prevent (early) termination in a (bad) local optimum. Unlike steepest descent a metaheuristic control provides a strategy to perform non-improving moves on certain occasions. The two most prominent principles of metaheuristic control are *annealing techniques* and *tabu search*.

Annealing techniques refrain from the idea of selecting the best neighbor in each iteration. Instead, random moves are evaluated and accepted, provided that the current solution is not deteriorated "too much" by non-improving moves. The term "annealing" refers to the characteristic that non-improving moves are prohibited gradually while the search proceeds. *Record-to-record travel* (RRT), cf. Dueck [15], uses the cost of the best solution found so far during the search – the *record* – to define acceptability: the selected neighbor is accepted if it is not worse than the best solution found so far by a prespecified absolute or relative *deviation* δ. Note that RRT is *not* a deterministic control since neighbor selection is random.

Tabu search (TS), proposed by Glover [19,20], always scans entire neighborhoods in each iteration and moves to the best neighbor even if it does not lead to an improvement, and in order to prevent cycling solutions are temporarily declared *tabu* for a number of iterations, unless their cost is below a so-called *aspiration level*. The tabu search idea leaves many degrees of freedom for the actual implementation. Whittley and Smith [40] present the *attribute based hill climber* (ABHC) heuristic as a parameter-free variant of tabu search. ABHC uses a generic attribute concept for specifying non-tabu neighbors, which has to be specialized for every problem domain. Given a set A of *attributes* over the set of feasible solutions, let $A(S)\subset A$ denote the set of attributes that a specific solution S possesses. During the search an *attribute memory* is maintained which stores for every attribute a the objective value of the best solution S^* visited so far with $a\in A(S^*)$. Now, a solution is acceptable if it would be the best solution visited so far for at least one attribute that it possesses. The procedure stops when no acceptable neighbors exist, but it can be restarted by resetting the attribute memory, see Derigs and Kaiser [9]. After specifying the attributes ABHC is parameter-free and, except for some tie-breaking, completely deterministic.

3.3. Modules of the framework

We have implemented three so-called *modules* LS-ABHC, LS-RRT and LNS-RRT, that concentrate on a single neighborhood paradigm and which are the basis for two *hybrid methods* which combine local search with large neighborhood search in the search process: HYBRID-ABHC and HYBRID-RRT. We first describe the three modules:

The LS-ABHC module. Starting from a given initial solution the trajectory of an attribute based hill climber heuristic through the solution space is determined by the set of solution attributes A, which is defined over the set of feasible solutions, and the set of neighborhoods evaluated. In our implementation A is the union of two subsets: $A1$ contains all connections, i.e. $A1 := \{(i,j) : i,j\in C_0\}$, and $A2 := \{(i) : i\in C$ is a vehicle customer$\}$ indicating whether a vehicle customer is served within a sub-tour or not. In one iteration the following moves are evaluated, and the best acceptable move is performed:

- Relocate: Every customer is moved to every other position in the solution, including the move to a new, empty tour.
- Exchange: Every customer is exchanged with every other customer.
- 2-opt*: Every subsequence of customers of every tour ending at the depot is exchanged with every such subsequence of every other tour, considering also empty subsequences, subsequences comprising the complete tour, and moves of subsequences to a new, empty tour.
- Relocate^I: Every customer is moved to every other position within its current tour.
- 2-opt: Every subsequence of customers of every tour is reversed.
- Relocate-sub-tour: Every sub-tour is transferred to every potential root customer, and every customer of the sub-tour is considered being visited first afterwards.

- Switch-vehicle-type: Every vehicle customer is switched from a complete vehicle visit to a truck-only visit or vice versa.

Algorithm 2. The LS-ABHC module.

```
1:  function LS-ABHC (initial solution $S$, attributes $A$, neighborhood $N$)
2:   $amem[a] := cost(S) \quad \forall a \in A(S)$
3:   $amem[a] := \infty \quad \forall a \notin A(S)$
4:   $S_{best} := S$
5:   repeat
6:     $S' := null, cost(S') = \infty$ ▷ Evaluate neighborhood
7:     for all $S^* \in N(S)$ do
8:       if $cost(S^*) < cost(S')$ and $\exists a \in A(S^*) : cost(S^*) < amem[a]$ then
9:         $S' := S^*$
10:      end if
11:    end for
12:    if $S' \neq null$ then ▷ Apply move and update memory
13:      $S := S'$
14:      $amem[a] := \min\{amem[a], cost(S')\} \quad \forall a \in A(S')$
15:      if $cost(S') < cost(S_{best})$ then
16:        $S_{best} := S'$
17:      end if
18:    else ▷ Reset memory
19:      $amem[a] := cost(S) \quad \forall a \in A(S)$
20:      $amem[a] := \infty \quad \forall a \notin A(S)$
21:    end if
22:  until stop criterion fulfilled
23:  return $S_{best}$
24: end function
```

Algorithm 2 states a completely generic version of ABHC i.e. A and N hide all problem-specific aspects. Before starting the search from an initial solution S the attribute memory *amem*, which stores for every attribute $a \in A$ the objective value of the best solution visited so far containing this attribute, is initialized with solution S (lines 2–3). S_{best} records the best solution found during the search. In each iteration all neighborhoods are scanned completely for the best feasible and acceptable neighbor S' of the current solution S (lines 6–11). A solution S^* is acceptable if it contains at least one attribute that has not yet occurred in a solution visited at least as good as S^*. The acceptance check is implemented efficiently as described by Whittley and Smith [40]. After moving to the best acceptable neighbor S' the attribute memory is updated (line 14). If, instead, no acceptable neighbor exists, the procedure is restarted by resetting *amem* with the current solution (lines 19–20). LS-ABHC terminates when a given time or iteration limit is reached.

The LS-RRT module. Record-to-record travel is designed to select neighbors randomly. Depending on the specific neighborhood there is some degree of freedom how this random selection is actually done. In the exchange neighborhood, for example, an intuitive selection scheme would be to simply select two random customers for an exchange. When generating such completely random moves the search process makes a lot of iterations, but usually the vast majority of evaluated moves is discarded due to infeasibility or insufficient quality. Instead, we use the same type of moves as for LS-ABHC but apply a selection scheme proposed by Bent and Van Hentenryck [2] that steers towards high quality neighbors. First, we select randomly one specific customer i and one specific type of neighborhood N leading to a set $N(S,i)$ of neighbors; for example we move a random customer to every other position in the solution in the case of relocate. Then, we sort $N(S,i)$ into an array $N' := \langle S^* \in N(S,i) \rangle$ with $i < j \Rightarrow cost(N'[i]) \leq cost(N'[j])$, and if $cost(N'[0]) \geq cost(S_{best})$, we select $r \in [0,1)$ randomly and evaluate $S' := N'[\lfloor r^\beta \cdot |N'| \rfloor]$ for acceptance. Here, β is a prespecified parameter. LS-RRT is displayed in Algorithm 3.

Algorithm 3. The LS-RRT module.

```
1:  function LS-RRT (initial solution $S$, deviation $\delta$, randomization $\beta$, neighborhoods $\mathcal{N}$)
2:   $S_{best} := S$
3:   repeat
4:     select neighborhood $N \in \mathcal{N}$ and customer $i \in C$ randomly
5:     array $N' := \langle S^* \in N(S,i) \rangle$ with
    $k < l \Rightarrow cost(N'[k]) \leq cost(N'[l])$
6:     if $cost(N'[0]) < cost(S_{best})$ then
7:       $S_{best} := N'[0]$
8:       $S := N'[0]$
9:     else
10:      select $r \in [0,1)$ randomly
11:      $S' := N'[\lfloor r^\beta \cdot |N'| \rfloor]$
12:      if $cost(S') < cost(S_{best}) \cdot (1+\delta)$ then
13:        $S := S'$
14:      end if
15:    end if
16:  until stop criterion fulfilled
17:  return $S_{best}$
18: end function
```

The LNS-RRT module. LNS-RRT is implemented rather straightforward as outlined in Algorithm 4: at the beginning of each iteration one removal heuristic and one insertion heuristic is selected randomly as well as the number q of customers to be removed (lines 4 and 5). Ropke and Pisinger [30] have introduced a learning mechanism to guide the selection of subheuristics under the name *adaptive* large neighborhood search; such a mechanism is not used within our framework since it did not show to pay off and would certainly make the approach more complex. Apparently it could be implemented easily. Then, the two selected subheuristics are applied to the current solution. After removing and reinserting the customers we perform a post-optimization step by applying 2-opt steepest descent to all modified tours (lines 6–9). As in the LS-RRT counterpart the resulting solution is accepted if it is not worse than the best solution found so far by deviation δ. LNS-RRT terminates when a given time or iteration limit is reached.

Algorithm 4. The LNS-RRT module.

```
1:  function LNS-RRT (initial solution $S$, deviation $\delta$, removal percentage $r$, removal heuristics $\mathcal{R}$, insertion heuristics $\mathcal{I}$)
2:   $S_{best} := S$
3:   repeat
4:     select subheuristics $R \in \mathcal{R}$ and $I \in \mathcal{I}$ randomly
5:     select $q \in \{4, \ldots, \min\{60, r \cdot |C|\}\}$ randomly
6:     $S' := S$
7:     remove $q$ customers from $S'$ using $R$
8:     reinsert removed customers into $S'$ using $I$
9:     apply 2-opt steepest descent to modified tours
10:    if $cost(S') < cost(S_{best}) \cdot (1+\delta)$ then
11:      $S := S'$
12:    end if
13:    if $cost(S') < cost(S_{best})$ then
14:      $S_{best} := S'$
15:    end if
```

```
16:   until stop criterion fulfilled
17:   return $S_{best}$
18: end function
```

The performance of LNS is highly influenced by the number q of customers which are removed. As recommended by Pisinger and Ropke [28] the maximum value that q may assume depends on a removal percentage parameter r and an absolute upper bound preventing the overburden of the simple insertion heuristics solving large instances. Following the proposal in Ropke and Pisinger [30] we select $q \in \{4, \ldots, \min\{60, r \cdot |C|\}\}$ randomly and here the removal percentage r is a prespecified parameter.

3.4. Hybrid methods

Using these modules we have implemented two algorithms which mix local search moves for the intensification part and large neighborhood search moves for the diversification part. In previous studies [1,8] we have already experienced that combined LS/LNS methods easily improve the individual methods. The first hybrid method is based on the attribute based hill climber, the second method uses record-to-record travel as the metaheuristic control.

HYBRID-ABHC and HYBRID-RRT decide randomly which type of neighborhood, i.e. LS or LNS, to use in each iteration of the search. Setting a probability parameter p^{LS} the selection is biased towards either LS or LNS. Here, one has to keep in mind the different computational efforts per LS iteration in the context of different metaheuristic controls: while LS-ABHC puts significant effort into one iteration and the search process is characterized by a sequence of rather few but high quality moves, a lot of iterations elapse in LS-RRT, but most moves are discarded due to a lack of quality.

The HYBRID-ABHC heuristic. Actually, combining LNS moves with the ABHC control is inconsistent with the *steepest descent–mildest ascent* idea of tabu search since LNS does not scan complete neighborhoods but generates only one (random) neighbor per iteration. Yet, this conceptual mismatch is ignored. After initializing the attribute memory in the same way as it is done in LS-ABHC the neighborhood search is a sequence of LS and LNS moves, deciding randomly in each iteration between LS (with probability p^{LS}) and LNS (with probability $1-p^{LS}$).

If LS is selected, all neighborhoods are scanned completely for the best acceptable solution, as in LS-ABHC. If such a solution exists, the search goes on to that solution; otherwise the memory is reset. If LNS is selected instead, one neighbor is generated by applying a removal heuristic and an insertion heuristic, as in LNS-RRT. This solution is accepted if (a) costs decrease and (b) the solution is acceptable according to the memory. Since LNS moves lead to distant solutions condition (b) is fulfilled very often and hardly imposes a restriction on the quality of a move. Consequently, we introduce (a) as a mandatory condition to prevent the search from accepting too many bad solutions. Still, we keep condition (b) for LNS moves since demanding (a) *and* (b) produced slightly better results in some preliminary tests than a) alone. Note that b) is the same criterion that applies to LS moves, i.e. a solution S' is acceptable if $\exists a \in A(S'), cost(S') < amem[a]$. Yet, unlike the acceptance check for LS moves described by Whittley and Smith [40], which considers the sets of entering and leaving attributes beforehand, the attributes of the LNS-generated neighbor are enumerated explicitly to check whether one attribute value has improved. If a move is accepted, the memory is updated accordingly.

The HYBRID-RRT heuristic. While HYBRID-ABHC is slightly unconventional, RRT can be combined perfectly with the two neighborhood concepts. An LS neighbor is selected from a subneighborhood in the same way as it is done in LS-RRT, and an LNS neighbor is generated as in LNS-RRT. In both cases the selected neighbor is accepted if its cost is not worse than the cost of the best solution found so far by deviation δ.

4. Computational results

In this section we present computational results for the different TTRP-variants. We compare our hybrid approaches, which have been implemented in C#, with state-of-the-art methods using benchmark instances from the literature. For these benchmark instances customer locations are given as points in the plane and the distances are euclidian distances. Our testing platform for all experiments is an Intel Xeon E5430 2.66 GHz PC with operating system Microsoft Windows 7.

Note that the hybrid LS/LNS approach can be regarded as a generic framework. Vogel [39] has developed a general software framework allowing to customize appropriate solvers for rich vehicle routing problems and analyzed the performance on several specific RVRP. Here, a standard parametrization has been determined using empirical tests on the standard VRP, then for every specific RVRP problem-specific parametrizations have been determined analogously. The results reveal that the approach is rather robust with respect to parametrization. For HYBRID-ABHC it turns out that in general an even proportion between the neighborhood searches produces the best results, and thus $p^{LS} = 0.5$ is selected as the standard parametrization. The range of suitable p^{LS} values for HYBRID-RRT is very different since an LS-RRT move consumes a much smaller computational effort than an LNS-RRT move. In other words, many LS moves need to be evaluated and performed to yield the progress of a single LNS move, and for a "good mixture" of the neighborhood concepts p^{LS} must be set to a rather high value. $p^{LS} = 0.9925$ turns out to be the best with the best deviation parameter $\delta = 0.005$. Details can be found in Derigs and Vogel [12].

As it is commonly done in such computational tests we conduct runs with fixed numbers of iterations instead of fixed running times. During all tests some minor parameters for subheuristic randomization and Shaw relatedness weights are set to fixed values according to Ropke and Pisinger [30] without any tuning. The performance of LNS is highly influenced by the number q of customers which are removed. We select a removal percentage of $r = 0.3$.

4.1. Results for the TTRP

Construction of start solutions. The TTRP formulation includes limits on the numbers of trucks and trailers used. In the TTRP with load transfer we initially ignore these limits to generate a solution using adaptations of the two most prominent construction methods for the VRP, the *savings method* by Clarke and Wright [5] and the *sweep method* by Gillett and Miller [18], to which we apply a simple repair heuristic to restore feasibility. We then use the best feasible solution found to start the improvement phase.

If a solution generated by the savings or sweep heuristic uses more trucks and/or trailers than allowed, the repair heuristic tries to rearrange customers into a smaller number of tours. In the first step the number of trailers is reduced, if necessary, by transforming pure or complete vehicle tours into pure truck tours. We allow the number of trucks to be increased in this step, even exceeding the truck limit, to guarantee that the solution complies with the trailer limit in the end. Finally, when the solution is feasible with respect to the trailer limit we repeatedly try to remove tours completely and reinsert the customers into the remaining tours using regret insertion until the truck limit is respected as well. Here, we consider all types of tours but try to remove those pure truck tours first which have been generated during the previous trailer saving step.

For the TTRP without load transfer these procedures have shown to be inappropriate to produce feasible start solutions. Here we have applied a modification of the generalized assignment heuristic (GAP) by Fisher and Jaikumar [16]. The modification contains two constraints for every tour: one for limiting the load of the vehicle to Q_k+Q_l and one for limiting the load of the trailer to Q_l. Also, it turned out that defining sophisticated objective functions using seed customers for tour clusters did not pay off. Thus the GAP is used to determine a feasible clustering only. Then we apply regret insertion for each cluster to construct a feasible routing by first building truck sub-tours.

Computational comparison. We evaluate our heuristics for the TTRP on a set of 21 instances generated by Chao [3]. These instances are derived from seven CMT problems for the standard VRP, cf. Christofides et al. [4], having between 50 and 199 customers by specifying 25%, 50%, and 75% of the customers as truck customers, respectively. In Table 2 we use instance names indicating the basic CMT problem and the truck customer percentage, separated by an underscore. Villegas et al. [38] cite the best known solutions (BKS) for these instances. We report these values together with the indication where this solution has been reported for the first time in column *BKS* of Table 2.

After initial tuning we use a TTRP-specific parametrization of $p^{LS}=0.9925$ and $\delta=0.025$ for HYBRID-RRT; we could not improve the standard parametrization of HYBRID-ABHC. Table 1 compares the aggregated results, i.e. the deviations (*dev*) from the BKS and the running time (*time*), for the two hybrid methods under standard parametrization and improved parametrization based on an experiment with five trials per instance. We define three iteration limit scenarios that lead to comparable (wall clock) running times among the two hybrid methods to examine how the solution quality depends on the running time and how much computational effort needs to be invested to obtain results of an appropriate quality. We run HYBRID-ABHC with 10,000, 50,000, and 200,000 iterations, and we run HYBRID-RRT with 1,500,000, 7,500,000, and 30,000,000 iterations. The instance sizes of the data set are rather small, but nevertheless the short running times indicate that the computational effort of the complex TTRP-specific neighborhoods/moves is moderate; especially for LNS moves the effort per iteration is small. Evidently, HYBRID-ABHC in its standard configuration is the best and HYBRID-RRT can compete with HYBRID-ABHC during shorter running times.

For TTRP with load transfer we compare HYBRID-ABHC with the recently published hybrid metaheuristic of Villegas et al. [38] which is the current state-of-the-art method for the TTRP. The authors obtain their best results using a configuration *GRASP/VNS with evolutionary path relinking*, and we state their average tour length (*TL*) from ten replications and the deviation from the BKS as well as the average running time in Table 2. The next group of columns lists the average results of HYBRID-ABHC with 100,000 iterations (not listed in Table 1). Evidently, HYBRID-ABHC can yield the same solution quality within slightly shorter running times on average. Note that Villegas et al. [38] run their tests on the same CPU as we do, so that computation times can be compared directly. Doubling the number of iterations to 200,000 the best results of five replications each turn out to be very close to the BKS and include four new best solutions. Thus with longer running time the heuristic is able to produce solutions which are competitive in quality or even slightly better than the current state-of-the-art methods for the TTRP.

A simple calculation comparing the capacity of trucks and the total demand of truck customers reveals that for two instances no feasible solution for the TTRP without load transfer exists. These are marked with *inf* in column *w/o load transfer* of Table 2 . For another eight instances marked by *feas* the solution obtained for the TTRP with load transfer does not require any load transfer and thus is feasible for the TTRP without load transfer too. For the remaining instances we give the tour length (best of five) as well as the deviation from the associated solution for the TTRP with load transfer. As can be seen, the additional distance to be traveled is dependent on the specific data. Note, that the running time is comparable for both variants.

4.2. Results for the TTRPTW

Solomon [36] has shown how insertion methods for the VRP can be adapted for the VRPTW by implementing a clever checking routine. As will be shown such a straightforward extension of TTRP-insertion methods to the TTRPTW where the feasibility of insertions is not only checked with respect to vehicle capacity but with respect to time windows, too, as done for instance by Lin et al. [27] is not sufficient since there are more options.

In Fig. 6 we display a complete vehicle tour where customer 2 is the root customer for two truck sub-tours. Note that the labels on the arcs represent the travel times. This route would be infeasible in the VRPTW-logic since after serving customer 2 at $ET=50$ the service of customers 4 and 6 within their time window becomes impossible. In the TTRP we may uncouple the trailer at root customer 2 first and then serve customer 4 and 6 within their time windows. After returning to the root customer 2 the truck has to wait one time unit before it can serve customer 2. Then after serving customer 2 the truck can serve customers 5 and 3 on the second truck sub-tour within their time windows. After

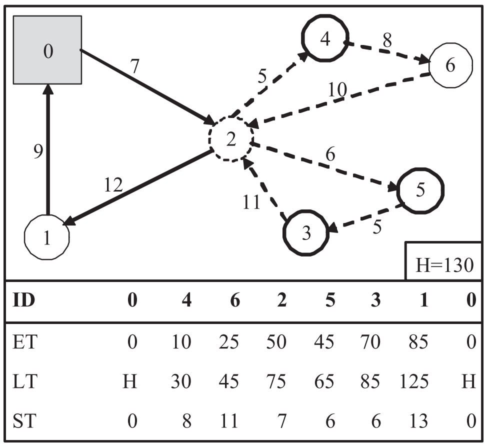
Table 1
Aggregated results for the TTRP.

			HYBRID-ABHC			HYBRID-RRT		
			10 K	50 K	200 K	1.5 M	7.5 M	30 M
Standard parametrization	*Best of five*	Avg. dev. (%)	1.72	0.71	0.14	1.68	1.14	1.12
		Max. dev. (%)	5.01	2.81	1.13	3.73	2.94	2.81
	Average of five	Avg. dev. (%)	2.76	1.20	0.49	2.41	1.84	1.68
		Max. dev. (%)	8.15	3.59	1.67	4.85	4.24	3.57
		Avg. time (s)	68.06	332.64	1339.21	69.04	337.64	1342.36
Improved parametrization	*Best of five*	Avg. dev. (%)	–	–	–	1.23	0.83	0.52
		Max. dev. (%)	–	–	–	4.12	3.48	1.66
	Average of five	Avg. dev. (%)	–	–	–	1.96	1.16	0.77
		Max. dev. (%)	–	–	–	4.53	3.96	3.14
		Avg. time (s)	–	–	–	75.89	368.67	1453.84

Table 2
Comparison with a state-of-the-art method for the TTRP.

Instance	Trucks		Trailers		BKS	Villegas et al. [38] *with load transfer Avg. of ten*			HYBRID-ABHC *with load transfer* 100 K, *Avg. of five*			HYBRID-ABHC *with load transfer* 200 K, *Best of five*		HYBRID-ABHC *w/o load transfer* 200 K, *Best of five*	
Name	m_k	Q_k	m_l	Q_l	TL	TL	Dev (%)	Time (s)	TL	Dev (%)	Time (s)	TL	Dev (%)	TL	Dev (%)
CMT1_25	5	100	3	100	564.68^S	565.99	0.23	70.20	564.81	0.02	103.87	564.68	0.00	*feas*	
CMT1_50					611.53^L	614.23	0.44	77.40	612.44	0.15	114.79	611.53	0.00	641.37	4.72
CMT1_75					618.04^S	618.04	0.00	63.00	618.04	0.00	114.42	618.04	0.00	*inf*	
CMT2_25	9	100	5	100	798.53^S	803.51	0.62	161.40	804.49	0.75	199.76	799.34	0.10	*feas*	
CMT2_50					839.62^S	841.63	0.24	169.20	839.62	0.00	218.12	839.62	0.00	878.50	4.63
CMT2_75					930.64^L	961.47	3.31	173.40	945.36	1.58	224.12	940.69	1.08	*inf*	
CMT3_25	8	150	4	100	830.48^S	830.48	0.00	363.00	830.48	0.00	460.24	830.48	0.00	*feas*	
CMT3_50					872.56^L	876.21	0.42	417.60	876.45	0.45	490.70	871.98	−0.07	*feas*	
CMT3_75					912.02^L	918.45	0.71	502.80	926.14	1.55	473.17	922.36	1.13	960.10	3.67
CMT4_25	12	150	6	100	1039.07^S	1050.11	1.06	1130.40	1046.58	0.72	851.15	1040.46	0.13	*feas*	
CMT4_50					1093.37^V	1100.95	0.69	1272.00	1102.07	0.80	1042.95	1093.89	0.05	*feas*	
CMT4_75					1152.32^V	1158.88	0.57	1546.80	1170.32	1.56	1082.40	1154.82	0.22	1261.24	7.77
CMT5_25	17	150	9	100	1287.18^S	1305.83	1.45	2636.40	1319.07	2.48	1681.80	1289.20	0.16	*feas*	
CMT5_50					1339.36^V	1354.04	1.10	2734.20	1352.50	0.98	1707.98	1341.25	0.14	1413.23	5.37
CMT5_75					1420.72^V	1437.52	1.18	3589.80	1452.25	2.22	1834.77	1423.55	0.20	1562.28	9.75
CMT11_25	7	150	4	100	1002.49^S	1003.07	0.06	883.80	1026.59	2.40	582.68	1001.48	−0.10	1047.24	4.57
CMT11_50					1026.20^L	1042.61	1.60	790.20	1031.81	0.55	629.41	1026.20	0.00	1138.91	10.98
CMT11_75					1098.15^L	1118.63	1.86	761.40	1098.46	0.03	667.54	1098.15	0.00	1297.51	19.15
CMT12_25	10	150	5	100	813.30^L	819.81	0.80	312.60	819.43	0.75	417.63	812.69	−0.08	*feas*	
CMT12_50					848.93^S	860.12	1.32	337.20	848.20	−0.09	493.06	848.12	−0.10	848.68	0.07
CMT12_75					909.06^S	909.06	0.00	378.60	909.25	0.02	567.25	909.06	0.00	941.45	3.56
Total					20,008.25	20,190.64		18,371.40	20,194.36		13,957.79	20037.59			
Average							0.84	874.83		0.81	664.66		0.14		
Max							3.31	3589.80		2.48	1834.77		1.13		

S: Scheuerer [32]; L: Lin et al. [25]; V: Villegas et al. [38].



ID	0	4	6	2	5	3	1	0
ET	0	10	25	50	45	70	85	0
LT	H	30	45	75	65	85	125	H
ST	0	8	11	7	6	6	13	0

Fig. 6. Time windows.

returning to the root customer the trailer gets coupled again and customer 1 can be served within its time window. The options to deliver a root customer before, after or in-between sub-tours are respected in our approach.

Construction of start solutions. Since the TTRPTW formulation does not include limits on the numbers of trucks and trailers used, we are able to use the simple regret insertion heuristic to construct a start solution, which guarantees that the time window constraints of each customer and the capacity constraints for the vehicles are respected.

Computational comparison. We evaluate our heuristics for the TTRPTW on a set of 18 instances generated by Lin et al. [27]. Lin et al. [27] converted selected VRPTW benchmark instances given by Solomon [36] to TTRPTW benchmark instances in the same manner as Chao [3] converted the VRP benchmark instances to TTRP benchmark instances. In the following we denote these instances by the Solomon notation and the truck customer percentage, separated by an underscore.

We have experienced that the effort for checking time window constraints slows down the search dramatically in HYBRID-ABHC due to the huge number of solutions evaluated. This pays off in very long runs only. In our experiments HYBRID-RRT outperformed HYBRID-ABHC when setting $p^{LS} = 0.5$, i.e. shifting the search towards LNS moves and thereby reducing the checking effort.

For TTRPTW with load transfer we compare HYBRID-RRT with the indirect search approach by Lin et al. [27]. We state their average tour lengths from five replications in the first group of columns of Table 3. The next group of columns lists the average results of HYBRID-RRT with 100,000 iterations. The running times are similar to the running times of Lin et al. [27] taking into account that they use a slower CPU. Next we list the best results of HYBRID-RRT with 200,000 iterations. When increasing the iterations the solution quality can be improved while the running time stays acceptable. After all, we could find new best solutions for 12 out of the 18 benchmark instances. Note that the best known solution values reported in column *BKS* are taken from Lin et al. [27] who generated and recorded the solutions during parameter testing.

We observe that the tour lengths do not differ significantly within the R101, R201, and RC201 subsets although the number of truck customers is increasing significantly. A closer look at the data shows that for these instances there is no need to attach a trailer to a truck on a route because of two reasons: The time windows are that restrictive that a vehicle cannot serve many customers and/or the capacity of the truck is high enough for transporting the entire demand. Thus in both cases, there is no need to solve the problem as TTRPTW, yet we can solve a simple VRPTW with vehicle capacity equal to the truck capacity. In the last group of columns of Table 3 we show that when solving the corresponding VRPTW instances solutions with about the same tour length can be obtained. To solve the VRPTW instance we

Table 3
Comparison with a state-of-the-art method for the TTRPTW.

Instance	Capacity		BKS	Lin et al. [27] *with load transfer Avg. of ten*			HYBRID-RRT *with load transfer*100 K, *Avg. of five*			HYBRID-RRT *with load transfer*200 K, *Best of five*				HYBRID-RRT *w/o load transfer*200 K, *Best of five*		VRPTW	
Name	Q_k	Q_l	TL	TL	Dev (%)	Time (min)	TL	Dev (%)	Time (min)	TL	K	L	Dev (%)	TL	Dev (%)	TL	V
C101_25	100	100	971.82	971.92	0.01	52.71	908.71	−6.49	34.76	905.35	11	9	−6.84	921.37	1.77		
C101_50			1106.90	1111.72	0.44	62.00	1019.02	−7.94	43.34	1009.96	12	8	−8.76	1064.66	5.42	1393.45	19
C101_75			1154.80	1166.35	1.00	58.19	1102.86	−4.50	40.09	1081.61	14	6	−6.34	1128.59	4.34		
C201_25	350	350	671.31	685.14	2.06	74.70	687.98	2.48	26.61	683.88	5	1	1.87	687.98	0.60		
C201_50			713.06	723.68	1.49	47.72	711.46	−0.22	18.03	700.63	6	0	−1.74	711.46	1.55	711.45	6
C201_75			711.45	711.45	0.00	44.61	711.46	0.00	18.14	711.46	6	0	0.00	*feas*			
R101_25	100	100	1651.16	1659.96	0.53	69.89	1648.77	−0.14	30.39	1647.15	20	0	−0.24	*feas*			
R101_50			1644.64	1660.73	0.98	60.13	1649.38	0.29	35.01	1646.00	20	1	0.08	1647.64	0.10	1644.64	20
R101_75			1644.64	1645.68	0.06	49.29	1647.62	0.18	36.84	1645.79	20	0	0.09	*feas*			
R201_25	500	500	1166.64	1193.16	2.27	56.52	1161.92	−0.40	16.57	1152.25	8	0	−1.23	*feas*			
R201_50			1148.17	1172.90	2.15	53.65	1152.22	0.35	14.01	1147.81	8	0	−0.03	*feas*		1147.80	8
R201_75			1147.80	1157.80	0.87	43.78	1154.32	0.57	19.90	1148.88	8	0	0.09	*feas*			
RC101_25	100	100	1717.01	1739.52	1.31	70.86	1710.61	−0.37	33.23	1702.09	17	5	−0.87	1709.16	0.42		
RC101_50			1770.40	1793.11	1.28	62.75	1747.62	−1.29	40.53	1739.10	18	4	−1.77	1742.42	0.19	1797.46	20
RC101_75			1784.06	1794.70	0.60	50.12	1748.60	−1.99	39.85	1741.93	18	5	−2.36	*feas*			
RC201_25	500	500	1277.05	1287.16	0.79	60.95	1275.48	−0.12	14.50	1265.56	9	0	−0.90	*feas*			
RC201_50			1273.69	1289.75	1.26	45.84	1282.31	0.68	13.55	1266.11	9	0	−0.60	*feas*		1265.56	9
RC201_75			1266.11	1276.67	0.83	38.34	1282.69	1.31	15.27	1267.27	8	0	0.09	*feas*			
Total			22,820.71	23,041.40		1002.05	22,603.03		490.62	22,462.82							
Average					1.00	55.67		−0.98	27.26				−1.64				
Max					2.27	74.70		2.48	43.34				1.87				

K, number of trucks used; L, number of trailers used; V, number of vehicles used.

used an implementation of the state-of-the-art VRPTW heuristic by Pisinger and Ropke [28] from Derigs and Vogel [11].

In the group of columns named *w/o load transfer* in Table 3 we present our results for TTRPTW without load transfer. For the 10 instances marked by *feas* the solution obtained for the TTRPTW with load transfer does not require any load transfer and thus is feasible for the TTRPTW without load transfer, too. For the other eight instances we state the total tour length (best of five replications) and deviation from the best solution for the TTRPTW with load transfer. Compared to the TTRP the deviations are significantly smaller here. Again, the running time is comparable for both variants.

5. Conclusion

In this paper we have shown how a simple and flexible metaheuristic framework can be customized to solve various variants of the truck and trailer routing problem, i.e the TTRP with/without load transfer as well as with/without time windows. The approach combines local search and large neighborhood search under standard metaheuristic control: the attribute hill climber method which is a specific implementation of tabu-search for problems without time windows and record-to-record travel for problems with time windows. In addition to standard VRP neigbor-hoods/moves we propose natural TTRP-specific moves. Computational tests on benchmark instances from the literature show that our approach is at least competitive to state-of-the-art approaches for the TTRP without time windows. For the TTRP with time windows we are outperforming previously published methods.

Our computational study has revealed that the definition of the basic problem is too simplistic and that the benchmark instances are not exploiting the full complexity of the problem. So for instance

1. Uncoupling and recoupling en-route occurs at no cost and without any consumption of time. This is not very realistic and especially for the TTRPTW time has to be considered.
2. The same holds for the load transfer between trailer and truck which is allowed in no time and without cost.
3. For the TTRPTW it might be suitable to uncouple the trailer at a customer and to serve the attached truck sub-tours before serving the customer. These options increase the algorithmic complexity when introducing time windows significantly more than when switching from the VRP to the VRPTW.
4. The benchmark instances which have been constructed from standard VRPTW instances by clustering the customers into two classes are not appropriate since some instances do not offer options for sub-tours and are "equivalent" to simple VRPTW in a sense.

Although Drexl [13,14] has presented the GTTRP, there is still much room for research on the TTRP. First, more realistic instances have to be developed which consider cost and time for serving such difficult customers only accessible by truck. Then it would be interesting to see how flexible the different approaches can be modified and to study their computational behavior on these scenarios.

References

[1] Bartodziej P, Derigs U, Malcherek D, Vogel U. Models and algorithms for solving combined vehicle and crew scheduling problems with rest constraints: an application to road feeder service planning in air cargo transportationOR Spectrum 2009;31:405–29.
[2] Bent R, Van Hentenryck P. A two-stage hybrid local search for the vehicle routing problem with time windows. Transportation Science 2004;38: 515–30.
[3] Chao IM. A tabu search method for the truck and trailer routing problem. Computers and Operations Research 2002;29:33–51.
[4] Christofides N, Mingozzi A, Toth P. The vehicle routing problem. In: Christofides N, Mingozzi A, Toth P, Sandi C, editors. Combinatorial Optimization. Wiley, Chichester; 1979. p. 315–38.
[5] Clarke G, Wright JW. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 1964;12:568–81.
[6] Cordeau JF, Gendreau M, Laporte G, Potvin JY, Semet F. A guide to vehicle routing heuristics. Journal of the Operational Research Society 2002;53: 512–22.
[7] Dantzig GB, Ramser JH. The truck dispatching problem. Management Science 1959;6:80–91.

[8] Derigs U, Gottlieb J, Kalkoff J, Piesche M, Rothlauf F, Vogel U. Vehicle routing with compartments: applications, modelling and heuristics. OR Spectrum 2011;33:885–914.
[9] Derigs U, Kaiser R. Applying the attribute based hill climber heuristic to the vehicle routing problem. European Journal of Operational Research 2007;177:719–32.
[10] Derigs U, Li B, Vogel U. Local search-based metaheuristics for the split delivery vehicle routing problem. Journal of the Operational Research Society 2010;61:1356–64.
[11] Derigs U, Vogel U. A computational study on neighborhood search heuristics for the open vehicle routing problem with time windows. In: Proceedings of the 8th metaheuristic international conference (MIC 2009); 2009. p. 109–22.
[12] Derigs U, Vogel U. Experience with a framework for developing heuristics for solving rich vehicle routing problems. Working paper, University of Cologne; 2011.
[13] Drexl M. On some generalized routing problems. PhD thesis. Germany: RWTH Aachen; 2007.
[14] Drexl M. Branch-and-price and heuristic column generation for the generalized truck-and-trailer routing problem. Journal of Quantitative Methods for Economics and Business Administration 2011;12:5–38.
[15] Dueck G. New optimization heuristics: the great deluge algorithm and the record-to-record travel. Journal of Computational Physics 1993;104:86–92.
[16] Fisher ML, Jaikumar R. Heuristics for rich vehicle routing problems. Networks 1981;11:109–24.
[17] Gerdessen JC. Vehicle routing problem with trailers. European Journal of Operational Research 1996;93:135–47.
[18] Gillett BE, Miller LR. A heuristic algorithm for the vehicle-dispatch problem. Operations Research 1974;22:340–9.
[19] Glover F. Tabu search – Part I. ORSA Journal on Computing 1989;1:190–206.
[20] Glover F. Tabu search – Part II. ORSA Journal on Computing 1990;2:4–32.
[21] Gottlieb J, editor. Evolutionary Algorithms for Constrained Optimization Problems. Aachen: Shaker Verlag; 2000.
[22] Hoff A. Heuristics for rich vehicle routing problems. PhD thesis. Norway: Molde University College; 2006.
[23] Laporte G, Gendreau M, Potvin JY, Semet F. Classical and modern heuristics for the vehicle routing problem. International Transactions in Operational Research 2000;7:285–300.
[24] Lin S, Kernighan BW. An effective heuristic algorithm for the travelling-salesman problem. Operations Research 1973;21:498–516.
[25] Lin SW, Yu VF, Chou SY. Solving the truck and trailer routing problem based on a simulated annealing heuristic. Computers and Operations Research 2009;36:1683–92.
[26] Lin SW, Yu VF, Chou SY. A note on the truck and trailer routing problem. Expert Systems with Applications 2010;37:899–903.
[27] Lin SW, Yu VF, Lu CC. A simulated annealing heuristic for the truck and trailer routing problem with time windows. Expert Systems with Applications 2011;38:15244–52.
[28] Pisinger D, Ropke S. A general heuristic for vehicle routing problems. Computers and Operations Research 2007;34:2403–35.
[29] Potvin JI, Rosseau JM. An exchange heuristic for routeing problems with time windows. Journal of the Operational Research Society 1995;46:1433–46.
[30] Ropke S, Pisinger D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation Science 2006;40:455–72.
[31] Savelsbergh MWP. The vehicle routing problem with time windows: Minimizing route duration. ORSA Journal on Computing 1992;4:146–54.
[32] Scheuerer S. A tabu search heuristic for the truck and trailer routing problem. Computers and Operations Research 2006;33:894–909.
[33] Schrimpf G, Schneider J, Stamm-Wilbrandt H, Dueck G. Record breaking optimization results using the ruin and recreate principle. Journal of Computational Physics 2000;159:139–71.
[34] Shaw P. A new local search algorithm providing high quality solutions to vehicle routing problems. Technical report. APES group; 1998.
[35] Shaw P. Using constraint programming and local search methods to solve vehicle routing problems. In: Proceedings CP-98 (fourth international conference on principles and practice of constraint programming); 1998.
[36] Solomon MM. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 1987;35:254–65.
[37] P. Toth, D. Vigo. The vehicle routing problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, PA; 2002.
[38] Villegas JG, Prins C, Prodhon C, Medaglia AL, Velasco N. A GRASP with evolutionary path relinking for the truck and trailer routing problem. Computers and Operations Research 2011;38:1319–34.
[39] Vogel U. A flexible metaheuristic framework for solving rich vehicle routing problems. PhD thesis. Germany: University of Cologne; 2011.
[40] Whittley IM, Smith GD. The attribute based hill climber. Journal of Mathematical Modelling and Algorithms 2004;3:167–78.