A Generic Heuristic for Vehicle Routing Problems with Multiple Synchronization Constraints

Technical Report LM-2014-05

Michael Drexl
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

4th November 2014

Abstract

Vehicle routing problems with multiple synchronization constraints (VRPMSs) are problems that exhibit, in addition to the usual task covering constraints present in any VRP, further synchronization requirements between the vehicles, concerning spatial, temporal, and load aspects. They have only recently caught the attention of scientists and are currently a very active field of research. This paper describes a generic heuristic that allows to solve many important types of VRPMSs.
Keywords: Heuristic; Vehicle Routing; Synchronization; Coordination; Trailer; Transshipment

Introduction

VRPs with multiple synchronization constraints (VRPMSs) are problems that exhibit, in addition to the usual task covering constraints present in any VRP, further synchronization requirements between the vehicles, concerning spatial, temporal, load, or other aspects. Despite the considerable practical relevance of VRPMSs, the scientific community has only recently begun to study them more intensively, which may be due to their inherent difficulty. Currently, VRPMSs are a very active field of research, and numerous publications have appeared in the last few years. Ref. [16] gives a survey.
Ref. [15] introduces the VRP with trailers and transshipments (VRPTT). This problem may be regarded as *the* archetypal VRPMS. Ref. [17] describes how numerous types of VRPs and VRPMSs can be *modelled* as VRPTTs. The contribution of the present paper is to present a heuristic for the VRPTT with which also many other VRPMSs can be *solved*.
The paper is structured as follows. In the next section, the VRPTT is described. Then, a classification of synchronization is given, the most important types of VRPMSs are listed, and the central difficulties in solving VRPMSs are illustrated. Afterwards, a heuristic for the VRPTT is presented, and it is explained many how other VRPMSs can be solved with this heuristic. The paper concludes with a summary and an outlook.

The VRP with trailers and transshipments

The VRPTT extends the classical (single- as well as multi-depot) VRP with time windows in two respects: It considers a heterogeneous fleet, distinguishing between autonomous and non-autonomous vehicles (lorries and trailers), and, in addition to depot and customer locations,

it introduces another type of locations, so-called transshipment locations. Autonomous vehicles can move on their own in space and time; non-autonomous vehicles can only move in time on their own (i.e., wait at a location), but must be accompanied (pulled) by an autonomous vehicle to move in space. At transshipment locations, trailers can be decoupled and parked, and, as the name implies, load transfers between vehicles can be performed.
Most importantly, there is no fixed one-to-one assignment of a lorry to a trailer. In general, any lorry can pull any trailer on any part of its itinerary, and a load transfer is possible between any pair of vehicles in any direction. Application- or instance-dependent compatibility restrictions between some vehicles may of course apply. Also, there usually are accessibility constraints specifying which vehicles can visit which customers and which transshipment locations.

A classification of synchronization

In ref. [16], the five types of synchronization described in Table 1 are identified.

Type	Description	Example in VRPTT
Task synchronization	Decision on which vehicle(s) fulfil(s) which task(s)	Which (type of) lorry and which (type of) trailer (if any) visit which customer(s)?
Operation synchronization	Decision if, when, and where which vehicle(s) perform(s) which operation that may be necessary or desirable to fulfil a task	Which vehicle transfers load when, where, into which other vehicle?
Movement synchronization	Decision on which vehicles join to form an autonomous composite vehicle able to move in space	Which lorry pulls which trailer along which road segment(s)?
Load synchronization	Decision on how load is partitioned between vehicles upon pickup at a customer and during transshipment operations	How much is loaded into the lorry and how much is loaded into the trailer if a lorry-trailer combination visits a customer to collect the latter's supply? How much load should a given vehicle transfer into another given vehicle at a transshipment location?
Resource synchronization	Decision on the use of common scarce resources by vehicles	Which vehicle is allowed to transfer load into a certain trailer in a particular time interval (assuming that the trailer can only receive load from one vehicle at a time)?

Table 1: Types of Synchronization

Operation synchronization (o.s.) can be further subdivided with respect to the time aspect:
- Pure spatial operation synchronization. This is the case when the temporal aspect is ignored and only the spatial aspect matters. There are quite a few papers considering only pure spatial o. s., cf. Section 5.3 in ref. [16], but in the present paper, the time aspect of synchronization is always taken into account.
- Operation synchronization with precedences. This is the case when two vehicles must start their part of a to-be-synchronized operation with a variable temporal offset Δ with $a \leq \Delta \leq b$, $0 \leq a < b$.
- Exact operation synchronization. This means that two vehicles must either start their respective part at the same time, i.e., with offset $\Delta = 0$ (simultaneous o. s.), or with a fixed offset $\Delta = a$ with $a > 0$ (deferred o. s.).

Two remarks concerning terminology: In this paper, the term *task* is used to denote a mandatory duty in a problem, something that must be done and requires zero or more units of some limited

resource. Examples are collecting supply or delivering demand at one location in a VRP, picking up load at one location and delivering this load to another location in a pickup-and-delivery problem, visiting a location to render a service in a service technician dispatching application etc. In some papers, the term *resource* denotes equipment such as vehicles and containers, or persons such as service staff and drivers, which is/are used to fulfil tasks. In this paper, the term *resource* is used to denote an arbitrarily scaled one-dimensional quantity that can be determined or computed. Examples are cost, time, load, or the information 'Is lorry k currently pulling trailer k'?'.

Important types of VRPMSs

Ref. [17] identifies several special cases of VRPMSs, which are described in Table 2. These VRPMS types may be considered particularly important because of their practical relevance and the number of scientific publications dealing with them. Publications up to the year 2012 are listed in the above-mentioned survey [16]; selected newer papers are listed in the table.

Problem type	Description and applications	References
VRPs with split pickup or split delivery, SDVRP	More than one vehicle may visit a customer and collect or deliver only part of the supply or demand; applications in ship routing; requires load s.	Nishi and Izuno [38], Archetti et al. [3]
PDPTWs with transshipments, PDPTWT	Pickup-and-delivery with time windows and optional load transfers; applications in less-than-truckload and passenger transport; requires operation and load s.	Qu and Bard [42], Masson et al. [32]
Multi- or N-echelon VRPs/LRPs, NEVRP/LRP	Multi-layer vehicle or location-routing problem with transshipment locations; hierarchy of transport levels with fixed assignment of vehicles to levels and mandatory transshipments between vehicles of different levels; applications in city logistics; requires operation s.	Hemmelmayr et al. [23], Nguyen et al. [37]
Simultaneous vehicle and crew routing and scheduling problems, SVCRSP	Planning of routes and rotations for vehicles and drivers **without given fixed schedule and without given lines to be serviced**; vehicles and drivers are separate, non-autonomous objects that must join pairwise to move in space; applications in long-distance road transport; requires movement s.	Drexl et al. [19]; Meisel and Kopfer [33] is on movement synchronization of autonomous and non-autonomous means of transport
Personnel dispatching problems with spatio-temporal synchronization, PDPSTS	Staff planning problems where tasks must be fulfilled simultaneously or with a known offset by several persons with possibly different qualifications; applications in service technician dispatching and home care staff routing; requires operation s.	Rasmussen et al. [43], Mankowska et al. [29]
VRPTT	Description see above; contains the preceding types as special cases, cf. [17]; specific applications in raw milk collection, fuel oil delivery, supermarket replenishment, among others; requires operation, movement, load, and resource s.	No heuristic published so far for the complete problem

Table 2: Important types of VRPMSs

The three central difficulties in solving VRPMSs with heuristics

There are three main causes why it is difficult to solve VRPMSs heuristically:
- The so-called *interdependence problem*.
- The existence of *different ways of performing a task*.

• *Optional synchronization possibilities* as opposed to mandatory synchronization requirements.

The interdependence problem has implications for exact solution approaches based on mathematical programming, too. Details on this are beyond the scope of the present paper; the reader is referred to ref. [16]. Different ways of performing a task can be modelled rather easily within mixed-integer programs, although they make the solution more difficult, as the solution space is enlarged. The same holds for optional synchronization possibilities.

The interdependence problem

The interdependence problem is that, because of the additional synchronization requirements, be it with regard to spatial, temporal, load, or other aspects, the vehicle routes become interdependent, and changes in one route may make one or several or all other routes infeasible. For example: If a change in the route of a lorry k means that it no longer moves from A to B, a trailer that k should have pulled from A to B can no longer perform its route. If a change in the route of a vehicle k means that k reaches a transshipment location later, the other vehicle involved in the transshipment must wait and may thus miss the time windows of customers that it should visit after the transshipment. If two vehicles k_1 and k_2 are to collect the supply of a customer i, and if k_1's route is changed so that k_1 no longer visits the customer, k_2's route may become infeasible because of insufficient capacity to collect the complete supply of i. If a lorry k_1 transfers load into a trailer at a transshipment location, the route of a second lorry k_2 supposed to transfer load into the trailer afterwards will become infeasible if k_1 transfers more than initially planned and thereby reduces the amount k_2 can transfer, leaving k_2 with insufficient capacity to perform the rest of its route. Such effects can of course propagate further and thus affect a complete route plan.

It is crucial to understand how the interdependence problem changes the nature of the problem. This is all the more important as heuristics for standard VRPs, most notably, local search approaches, usually exploit the *independence* of routes and, hence, are not directly applicable to VRPMSs. Consider, for example, local search moves such as an intra-route shift of a customer to another position or an inter-route swap of two customers between two routes in a standard VRP. The change such a move incurs on the overall route plan is limited to the route(s) where customers are actually relocated, that is, in most cases, to one (in the case of an intra-route move) or two routes only. Cost changes and feasibility need only be checked for the affected routes. For many such neighbourhoods, techniques are available to evaluate cost and feasibility in amortized constant time, independent of the length of the involved routes (measured in number of customers) and independent of the number of routes in the solution (Savelsbergh [47], Irnich et al. [25], Vidal et al. [50]).

If, as it is common in VRPs, the objective function strives to minimize the number of vehicles used and/or to minimize the costs or overall distance travelled, evaluation of the objective function after a move can be done in constant time also in the case of VRPMSs. This is because the objective function contribution of each route is still independent of other routes. The situation is different if the objective function consists in minimizing the maximal route duration or the duration of execution of the complete route plan. Then, to re-evaluate the objective function after a move in a standard VRP still requires only the recomputation of the schedule of the modified route(s), whereas in VRPMSs, the interdependence problem may require a computationally intensive, linear-time rescheduling of all routes.

In the existing literature on heuristics for VRPMSs, different approaches are employed to deal with the interdependence problem (cf. ref. [16]). The three most important ones are (i) allowing intermediate infeasible (iii) solutions and penalizing them in the objective function, (ii) only estimating or incompletely evaluating moves, and performing indirect search (to this author's knowledge, these techniques were first used in the context of VRPMSs by Oertel [39], De Rosa et al. [10], and Li et al. [28] respectively; Derigs and Döhmer [12] give an introduction to indirect search).

Different ways of performing a task

The term *transport* essentially means to bring a physical object from one location (point in space and time) to another. Therefore, almost any task to be performed in any type of (vehicle) routing problem can be represented by one or more such basic activities, which are henceforth called *elementary (transport) tasks*. In standard vehicle routing problems, such as the VRP or the pickup-and-delivery problem with time windows (VRPTW and PDPTW respectively), there is exactly one way of performing a task: In the VRPTW, each task consists in collecting a certain amount of load at a customer location and transporting this load to the depot (or the other way round in VRPTWs where the customers have demands instead of supplies). In the PDPTW, a task consists in picking up a good at a certain location and transporting the good to another location. In both cases, each task is performed by exactly one vehicle. In many VRPMSs, as will be shown shortly, more than one way of performing a task may exist, and more than one vehicle may be necessary to fulfil a task.

In *generalized routing problems*, an additional degree of freedom exists. The simplest such problem (which is still NP-hard in the strong sense) is the generalized travelling salesman problem (GTSP), where there is a set of disjoint clusters of vertices, and a closed route is sought that visits exactly one vertex from each cluster. Hence, a decision must be made which vertex to visit from each cluster. Put differently, this means that there are different ways of performing a task: For fulfilling the task of visiting a certain cluster of vertices $\{1, \ldots, n\}$, there are n different ways, namely, visiting vertex i, $i = 1, \ldots, n$. Such generalized (vehicle) routing problems have been studied by Karapetyan and Gutin [26] (GTSP), Blais and Laporte [6] (generalized routing problem, GRP), Baldacci et al. [5] (generalized VRP, GVRP), and Moccia et al. [35] (GVRP with time windows, GVRPTW). However, these problems are not VRPMSs in the sense the term is used here, because they lack the synchronization requirements (other than task synchronization in the GVRP and the GVRPTW). Nevertheless, solution approaches for VRPMSs must be able to handle the existence of different ways of performing a task and are thus in principle also suitable for generalized routing problems.

Whereas the interdependence problem is a constitutive property of VRPMSs, in some problems, there is only one way of performing a task. In general, this is regularly the case when only task and movement synchronization are necessary, but it can also occur in cases with operation and resource synchronization. In Table 2, the pure simultaneous vehicle and driver routing problems and personnel dispatching problems with synchronization fall into this category.

A property of the generalized routing problems just described is that the number of different ways of performing a task is finite (and usually rather small). For some VRPMSs where there are different ways of performing a task, this is also true. For example, in PDPTWTs, it is sometimes assumed that a transport task must either be fulfilled without transshipment or with one transshipment at one of a small number of potential transshipment locations. In such cases, it is possible and sensible to enumerate and explicitly consider all different ways of performing a task in a solution heuristic (see, e.g., Mitrović-Minić and Laporte [34]). On the other hand, in many cases, the number of different ways of performing a task is large or even infinite. Above all, this is the case in most split delivery VRPs, but also in PDPTWTs with a larger number of potential transshipment locations and/or the possibility of transshipping a task more than once.

Many authors have considered different ways of performing a task in their solution approaches to VRP variants and VRPMSs. On an abstract level, the following three approaches to determining a way of performing a task in a solution procedure can be distinguished:

- *Implicit determination of the way a task is performed.* This means that the routing of the vehicles ensures that a task is fulfilled. A notable reference using this approach is Del Pia and Filippi [11].
- *Explicit determination of subtasks or task legs through task splitting.* This means that in each iteration of a heuristic, one concrete way of performing a task is selected/determined, and this way is then included into the solution constructed in the iteration. This approach is taken by Bock [7].

- *Explicit determination of derived tasks.* This is a variant of the preceding approach. As before, concrete elementary tasks are determined and planned, but each of these tasks may completely or partially fulfil more than one original task. (No paper using this approach known.)

To better illustrate these concepts, consider the VRPTT. There, implicit determination of a way of performing a task means that a solution procedure determines a route plan based on the customer supplies, without specifying the actual flow of each customers' supply. For example, in a solution, a customer i may be visited by a lorry k that afterwards visits transshipment location l and transfers a certain amount of load to trailer k' there. It is not specified how much of i's supply is transferred, and it is not necessary to know this. Whether the transferred load was completely or partially collected at i or at some other customer visited by k before l is irrelevant. Explicit determination of subtasks through task splitting in the VRPTT means to define, before routing the vehicles, whether, and, if so, where how much of each customer's supply is transshipped. For example, it can be specified that the supply of customer i is collected by a lorry, transported to transshipment location l and transferred there, and finally brought to the depot. Thus, the original task of transporting the supply of i to the depot, $i \to d$, is split into two elementary subtasks or task legs, $i \to l$ and $l \to d$. (Note that this does *not* mean that the vehicle k performing a subtask $u \to v$ goes directly from u to v. k may visit an arbitrary number of locations after u and before v, but the load collected or received at u will stay on k from u to v.) Explicit determination of derived tasks in the VRPTT means that a set of elementary transport tasks is computed, some of which comprise the supply of more than one customer. For example, if there are two customers i_1 and i_2 with a supply of five units each, one transshipment location l, and one depot d, there are two original elementary tasks $i_1 \to d$ and $i_2 \to d$. These can be performed by transporting i_1's and i_2's supplies to l and from there to d, resulting in the derived tasks $i_1 \to l$, $i_2 \to l$, and $l \to d$, with a capacity requirement of five, five, and ten units respectively. The derived elementary task $l \to d$ partially fulfils both original tasks. Note that derived tasks exceeding the capacity of the largest vehicle must be split into sufficiently small tasks.

Mandatory vs. optional synchronization

In many VRPMSs, synchronization is *mandatory*. This means that there is no choice whether or not to synchronize vehicles. For example, in multi-echelon problems, transshipments are unavoidable by definition. In the simultaneous vehicle and crew routing and scheduling problem, it is clear that the movements of lorries and drivers must be synchronized, because lorries as well as drivers are non-autonomous. In personnel dispatching problems, a task may, for technical reasons, require technicians to be at the same location at the same time, making operation synchronization necessary. In the VRPTT, depending on the data of each instance, synchronization (other than task s.) may be *optional*: If there are sufficiently many lorries with sufficient capacity, so that each customer can be visited by a lorry without a trailer, feasible solutions using no trailers and performing no transshipments exist, so that operation, movement, load, and resource s. are unnecessary. This implies an additional degree of freedom and, hence, complexity. In a heuristic, it is not at all obvious how to determine which trailers, if any, should be used, and which transshipments (if any) should be performed.

A heuristic for the VRPTT

To overcome the difficulties mentioned in the previous section, the proposed heuristic uses the following three ideas:

- The interdependence problem and the resulting non-trivial feasibility checks of modified solutions are addressed using the sophisticated procedures developed by Masson et al. [31] and Grangier et al. [21].
- Each original task is decomposed into an ordered sequence of one or more subtasks. All subtasks are pickup-and-delivery tasks. Separate routines are used for the systematic determina-

tion of ways how to perform the original tasks (taking into account optional synchronization) and for planning the computed subtasks. A PDPTWT is solved to consider the relationships between the subtasks.

- Movement synchronization is handled by sequential solution of appropriately defined PDPTWTs.

The heuristic consists of three steps:

(i) Determine one or more possible ways of performing each task by explicitly computing subtasks through task splitting and/or by computing derived tasks.

(ii) Solve a pickup-and-delivery problem with time windows and transshipments for planning the subtasks or derived tasks determined in step (i). Use a fleet consisting of all lorries and all trailers of the original problem. Consider all trailers autonomous and set their costs equal to the costs of a lorry-trailer combination to avoid putting the lorries at a disadvantage.

(iii) Solve a pickup-and-delivery problem with time windows and transshipments for planning the tasks corresponding to the *route segments* of a step (ii) solution. (A route segment starts and ends at a depot or a transshipment location. For example, consider the step (ii) route $d \to i \to l \to j \to d$, where d is a depot, i and j are customers, and l is a transshipment location. This route consists of the two segments $d \to i \to l$ and $l \to j \to d$, which are regarded in step (iii) as the pickup-and-delivery tasks $d \to l$ and $l \to d$.) Use a fleet consisting of the lorries only. Assign two capacities to each lorry, namely, 'number of lorry tasks currently performed' and 'number of trailer tasks currently performed', both with an upper bound of one, where a lorry (trailer) task is a segment of a route performed by a lorry (trailer) in a step (ii) solution. Moreover, consider compatibility constraints between tasks: If a lorry task is loaded onto a lorry, this lorry can perform a trailer task only if all customers visited by the lorry task are also accessible by a trailer. This can very easily be modelled by assigning a trailer capacity requirement of one to all lorry tasks that visit a customer unaccessible by a trailer.

Step (i), together with the first step (ii) iteration (which determines a complete solution) can be seen as an initial construction heuristic.

Step (ii) serves to 'route the capacities' of the available original fleet. Within the PDPTWT solution procedure used in step (ii), it is possible to further split the sub- or derived tasks computed in step (i) (cf. Masson et al. [30]). That is, new ways of performing a task can be computed also in step (ii). The underlying heuristic assumption, though, is that each sub-/derived task is always fulfilled by one vehicle, either a lorry or a trailer. This means that if a subtask is to transport the supply of a customer i to transshipment location l, then, even if i may be visited by a lorry-trailer combination, the complete supply of i is loaded into either the lorry or the trailer visiting i.

Step (iii) routes autonomous vehicles in order to move the capacities along their routes as determined in step (ii).

The PDPTWT heuristic(s) used in steps (ii) and (iii) may consider only one or more than one way of performing each task. For example, in step (ii), the heuristic may consider performing a task $i \to j$ without a transshipment or with one transshipment at the transshipment location incurring the smallest detour and check the costs resulting from using each of these ways. The ALNS described in ref. [30] takes this approach. In step (iii), different ways of performing a task can result from different partitionings of a step (ii) route into segments. For example, a route $d \to p_1 \to p_2 \to l_1 \to p_3 \to l_2 \to d_3 \to d$ can be partitioned into one segment (the complete route), the segments $d \to p_1 \to p_2 \to l_1$ and $l_1 \to p_3 \to l_2 \to d_3 \to d$, the segments $d \to p_1 \to p_2 \to l_1 \to p_3 \to l_2$ and $l_2 \to d_3 \to d$, and the segments $d \to p_1 \to p_2 \to l_1$, $l_1 \to p_3 \to l_2$, and $l_2 \to d_3 \to d$.

If the solution procedure used in step (ii) is capable of determining new ways of performing a task (as it is the case with the procedure described in ref. [30]), step (i) is not necessary. However, the approach(es) used in step (i) could be rather sophisticated and time-consuming ones to be called only a limited number of times, for example, to act as diversification mechanisms if the above three-step heuristic were to be performed repeatedly in a multi-start fashion.

An alternative interpretation of what happens in steps (ii) and (iii) is as follows. The problem to be solved in step (iii) is a pickup-and-delivery problem where the original tasks are to pick up vehicle capacities at their respective origin depots and transport them to their respective destination depots. The way of performing each of these original tasks has been determined in step (ii) by explicitly determining subtasks: The original task 'transport a trailer with a capacity of 10 from depot d to depot d' may, for example, have been split into subtasks $d \to l_1$ and $l_1 \to d$ in step (ii). Hence, steps (i) and (ii) of the procedure can be regarded as routines for computing ways of performing tasks, and steps (ii) and (iii) can be viewed as routines for planning predetermined sub- or derived tasks, so that step (ii) has a double function.
The procedure performs task, spacial operation, load, and resource synchronization simultaneously in step (ii), and movement and temporal operation synchronization in step (iii). Note that the heuristic described above is independent of a concrete procedure for each of the three steps. In fact, numerous possibilities exist for tackling step (i), and any algorithm for the PDPTWT can be used in steps (ii) and (iii). The following subsections discuss such options.

Splitting tasks

In the VRPTT, there are infinitely many ways of performing a task. Even if it is assumed that the complete supply q of a customer must be collected at the customer location by one vehicle, if the good to be collected is homogeneous, e.g., a liquid, the fraction of a certain customer's supply transferred at a transshipment location can lie anywhere in the interval $]0, q]$. For heuristic purposes, and for the sake of solvability, however, it is justified to consider only a (discrete, finite, small) subset of the number of ways. For example, the number of times a task may be transshipped can be limited to one or two, and it can be required that the complete load of an original task be transferred in each transshipment process.
In view of this, the following procedures are proposed for determining ways of performing a task ex ante:

(i) The simplest method is to limit the number of ways and to randomly select if and where a task should be transshipped as well as if and how the pickup of the supply of customers accessible to trailers should be split between a lorry and a trailer that both visit the customer.

(ii) Another possibility is to compute the movements of customer supplies by solving a kind of min-cost network flow problem (cf. Ahuja et al. [1] for a thorough treatment of flows in networks). The network consists of one vertex for each customer, each transshipment location, and each depot, plus one artificial sink vertex. In network flow terminology, the customers are obviously the supply vertices, and the artificial sink is the sole demand vertex with a demand equal to the sum of all customer supplies. Arcs exist between any pair of vertices except that all arcs leaving a depot vertex lead to the sink and no arc leaves the sink. The capacities and costs of arcs leaving and/or entering customer vertices are respectively equal to the average capacity and traversal costs of all types of lorries and trailers that can visit the customer. The capacities and costs of arcs (i, j) leaving transshipment vertices and entering transshipment or depot vertices are respectively set to $n_{ij}^{est} \cdot q_{ij}^{avg}$ and c_{ij}^{avg}, where n_{ij}^{est} is an estimate of the number of vehicles that can traverse (i, j) and might receive load at i, q_{ij}^{avg} is the average capacity of all types of lorries and trailers that can traverse (i, j), and c_{ij}^{avg} is the average traversal cost of all types of lorries and trailers that can traverse (i, j). Arcs leading to the sink are uncapacitated and have zero cost. To avoid that all flow goes directly from each customer to the closest depot, which would be the case in the above network, the number of arcs entering each depot must be limited to the number of available vehicles (if this number is finite) or to some other reasonable number. Alternatively, the costs of arcs from customers to depots can be increased. Relevant ways of performing the tasks can then extracted from a solution to this network flow problem as follows. Select an arbitrary path from a customer vertex to a depot vertex. This path is a (complete or incomplete) way of performing the task corresponding

to the customer. Determine the minimal flow on any arc of this path. Remove the pertinent arc from the solution and reduce the flow on all other arcs of the path by this minimum value. Repeat until all arcs are removed.
To better account for the temporal aspect present in the VRPTT, i.e., the time windows at depot, customer, and transshipment locations and the arc travel times of the lorries, flows over time can be computed. This can be done based on a time-space network, cf. Grünert and Sebastian [22].
Instances of network flow problems such as those just described, resulting from typical real-world VRPTT instances (several hundred customers and transshipment locations), are of a size that can be solved to optimality with standard mixed-integer programming solvers in reasonable time.

(iii) An alternative is to solve the following continuous relaxation of a type of service network design problem on a time-space network $D = (V, A)$ with the same structure as that of item (ii), where K is the set of available vehicle (or, rather, capacity) types, K^L and K^T with $K = K^L \uplus K^T$ are the sets of lorry and trailer types respectively, q^k the capacity of vehicle type k, c_{ij}^k the cost of traversing arc (i, j) with vehicle type $k \in K$, s is the total supply of all customers, s_i the supply of customer i, $y_{ij}^k \geq 0$ denotes the number of vehicles of type k moving along arc (i, j), and $x_{ij} \geq 0$ denotes the flow of load along arc (i, j):

$$\sum_{k \in K} \sum_{(i,j) \in A} c_{ij}^k y_{ij}^k \to \min \tag{1}$$

$$x_{ij} \leq \sum_{k \in K} q^k y_{ij}^k \quad \forall\ (i, j) \in A \tag{2}$$

$$\sum_{(h,i) \in A} x_{hi} - \sum_{(i,j) \in A} x_{ij} = \begin{cases} s_i, & i \text{ is a customer vertex} \\ 0, & i \text{ is a transshipment or depot vertex} \\ -s, & i \text{ is the sink vertex} \end{cases} \quad \forall\ i \in V \tag{3}$$

$$\sum_{(h,i) \in A} y_{hi}^k - \sum_{(i,j) \in A} y_{ij}^k = 0 \quad \forall\ k \in K, (i, j) \in A \tag{4}$$

$$\sum_{k \in K^L} \sum_{(i,j) \in A} y_{ij}^k = 1 \quad \forall\ i \in V \tag{5}$$

$$\sum_{k \in K^T} \sum_{(i,j) \in A} y_{ij}^k \leq 1 \quad \forall \text{ customer vertices } i \text{ accessible to a trailer} \tag{6}$$

(2) link the flow and vehicle variables. (3) are the flow conservation constraints for the supplies, (4) those for the vehicles. (5) and (6) ensure that a customer vertex is not visited too often.
From a solution to this problem, relevant ways of performing the tasks can be determined as described for the previous alternative.

(iv) A further option is to solve a warehouse location problem (Daskin [9]). All transshipment and depot locations are potential warehouses (with fixed opening costs of zero). For warehouses corresponding to transshipment locations l, the capacities are set to $n_l^{est} \cdot q_l^{avg}$, where n_l^{est} is an estimate of the number of vehicles that can visit l, and q_l^{avg} is the average capacity of all types of lorries and trailers that can visit l. For warehouses corresponding to depots, the capacity is set to infinity. The assignment costs of a customer i to a potential warehouse corresponding to a transshipment location l are set to the costs of traversing the arcs (i, l) and (l, d), where d is the depot closest to l, with an average vehicle. Assignment costs of a customer i to a potential warehouse corresponding to a depot d are set to the traversal costs of the arc (i, d) with an average vehicle. The way of performing a task i assigned to a warehouse corresponding to a transshipment location l is then the two sub-tasks $i \to l$ and $l \to d$, where again d is the depot closest to l. The way of performing a task i assigned to warehouse corresponding to a depot d is the original task $i \to d$.

The warehouse location problem instances resulting from typical real-world VRPTT instances can also be solved with standard solvers.

(v) Ref. [30] describes a procedure for determining which transshipment location, if any, should be used for a set of tasks, given a partial solution to a PDPTWT. This procedure is based on solving a variant of the k-star hub problem (Oertel [39]) and can also be applied, with only minor modifications, if no tasks at all have been planned so far.

(vi) Finally, it is also possible to compute a solution to a more restricted VRP, for example, a truck-and-trailer routing problem (TTRP), for which numerous heuristics exist (cf. Prodhon and Prins [41]). The TTRP is a special case of the VRPTT with a fixed lorry-trailer assignment, i.e., each trailer can be pulled by only one lorry, and only this lorry can transfer load into the trailer. This implies that there is no interdependence problem in the TTRP, as the route of each lorry completely determines the route of its assigned trailer (if any), even though it is allowed that trailers be decoupled and parked at transshipment locations. All tasks that are transferred from a lorry to its trailer at transshipment locations constitute split or derived tasks for step (ii). Further splitting of tasks can then be performed within the step (ii) solution procedure (cf. again ref. [30]).

Note that in the first three approaches described above, the only limit on the number of task legs that might be created for any task is the number of transshipment locations. In the last three approaches, each task is split into at most two legs.

If the number of vehicles is limited, it is not guaranteed that the sub- and derived tasks determined in step (i) can be feasibly performed with the existing fleet in step (ii) (similar for steps (ii) and (iii)). However, this can be handled by introducing a *task bank*, where original tasks are placed for which at least one sub- or derived task could not be performed (cf. Ropke and Pisinger [45]). Alternatively, infeasible solutions violating time window or capacity constraints can be allowed and penalized in the objective function.

Routing the subtasks

Heuristic solution procedures for PDPTWTs have been proposed, among others, by Shang and Cuff [48], Oertel [39], Mues and Pickl [36], Mitrović-Minić and Laporte [34], Thangiah et al. [49], Qu and Bard [42], Masson et al. [30]. The adaptive large neighbourhood search (ALNS) by Masson et al. [30] is currently probably the most powerful one, in particular as it makes use of a sophisticated procedure to check the feasibility of the insertion of a(n original, sub-, or derived) task into a route in amortized constant time. The procedure is described in detail in the seminal work Masson et al. [31]. Other authors use linear-time routines for checking feasibility. This author is, however, unaware of work on pertinent data structures and algorithms for constant-time evaluations of min-max, makespan and cumulative duration objective functions in VRPMSs. An advantage of ALNS is that it allows to determine new ways of performing a task dynamically, during the process of solving the PDPTWT in step (ii). Essentially, all procedures described in the previous section on splitting tasks can be used as part of a reconstruction heuristic in an ALNS. What is more, a constant-time feasibility check also allows efficient use of local search procedures, so that not only an ALNS can be extended by local search, but also other metaheuristics such as tabu search or memetic algorithms can be used to solve the PDPTWTs for routing the subtasks in steps (ii) and (iii) of the VRPTT heuristic.

It is allowed or required in most PDPTWT papers that the pickup of a load at a transshipment location be performed *at any time after* its delivery (operation synchronization with precedences and $0 = a \leq \Delta \leq b = \infty$ in the terminology of ref. [16], see page 2). In many applications, however, including classical ones of the VRPTT, simultaneous or exact deferred o. s. ($\Delta = 0$ or $\Delta = a > 0$ respectively) or o. s. with precedences with a variable but finite offset ($0 \leq a \leq \Delta \leq b < \infty$) is required. The above-mentioned constant-time feasibility check described in ref. [31] is suitable for o. s. with precedences with unlimited offset ($0 = a \leq \Delta \leq b = \infty$) and is extended by Grangier et al. [21] to handle the case of simultaneous o. s., i.e., the case that, for a transshipment, both involved vehicles must be at the transshipment location at the

same time. The approach of ref. [21], in turn, can be generalized to cover all cases of operation synchronization. To explain how this works, the original approach is described in the following. In PDPTWTs, two aspects must be checked to ensure feasibility of a route plan with respect to time: (i) The dynamic time windows of every vertex must respect the static time windows of all other vertices within its route as well as the dynamic time windows of the vertices of all sub- or derived tasks with the same original task. (ii) The route plan must be logically consistent with respect to precedences between pickups and deliveries of tasks in different routes.
(i) means the following: Given an original task $i \to j$, assume that the way of performing this task is $i \to l$, $l \to j$, and that the two routes performing these subtasks are $d \to i \to l \to d$ and $d \to l \to j \to d$. Further assume that the time windows for visiting d, i, j, and l are $[0, 10]$ (static time windows), and that the driving between any pair of vertices is 2. Then, i can be visited along its route in the interval $[2, 6]$. If the static time window of l is $[0, 4]$, however, the dynamic time window of i is $[2, 2]$. If the static time window of l is $[0, 1]$, the route is infeasible. In VRPMSs, the interdependence problem must be taken into account. If the time window of j is $[6, 7]$, arriving at i at time 6 is infeasible, so the dynamic time window of i, when taking into account the subtask $j \to d$, performed in another route, becomes $[2, 5]$.
(ii) means the following: Given the above route plan, a task $i' \to j'$ with subtasks $i' \to l'$ and $l' \to j'$ cannot be inserted into the route plan as follows, even if all static time windows are ignored: $d \to l' \to j' \to i \to l \to d$ and $d \to l \to j \to i' \to l' \to d$. Although the two routes are still temporally feasible by themselves, performing the route plan would require visiting i' before l' before j' before i before l before j before i', that is, i' before i', a contradiction.
To ensure these two aspects, in ref. [31], a so-called *precedence graph* is used. This is a directed acyclic graph that, as its name implies, models the logical precedence relationships between vertices according to a given route plan for a given set of sub- or derived tasks. The precedence graph contains all vertices of all subtasks and an arc for each logical precedence relation between two vertices. The travel times of the arcs in the precedence graph are set to the values of the corresponding arcs in the original graph. To check (i) in constant time, in ref. [31], the concept of *forward time slack*, introduced by Savelsbergh [47], is extended. The forward time slack, as described in ref. [47], allows to check the feasibility of an insertion of a vertex i into a route ρ in constant time, taking into account the vertices already present in ρ. In ref. [31], vertices from other routes are included into this calculation, namely, vertices from subtasks with the same original task as i. These vertices are identified via the precedence graph. To check (ii), in ref. [31], the precedence graph is checked for cycles, which can be done in constant time using adequate data structures.
The precedence graphs for the two above route plans, without and with task $i' \to j'$ respectively, in an 'any-time-after PDPTWT' look as shown in the following Figure 1, where the depot vertex d has been duplicated into s and e, the start and the end depot vertex:

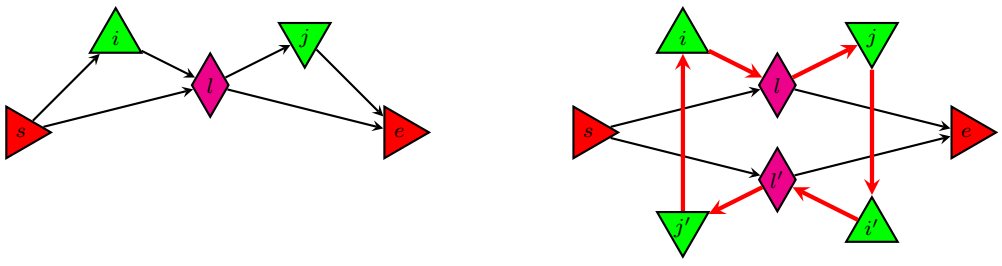

Figure 1: Precedence graphs

Given two subtasks $i \to l$ and $l \to j$ requiring simultaneous o. s., applying the idea of ref. [21] to a PDPTWT setting creates a precedence graph as depicted in the following Figure 2:
In the precedence graph of Figure 2, three vertices, l_1, l_2, and l_3, and three arcs, (l_1, l_2), (l_1, l_3), and (l_3, l_2), all with a travel time of zero, are created to model the transshipment at l. It is easy to see that the only possibility for avoiding logical inconsistencies is to perform the delivery to and the pickup from l at the same time.

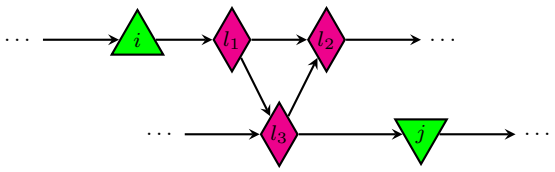

Figure 2: Modified precedence graph

In general, the offset Δ that must or may occur between the delivery to and the pickup from l fulfils $0 \leq a \leq \Delta \leq b \leq \infty$. Whatever the concrete values of a and b, the respective temporal relationship between the two subtasks $i \to l$ and $l \to j$ specifying a transshipment of an original task $i \to j$ at l is ensured if the travel times of the arcs (l_1, l_2), (l_1, l_3), and (l_3, l_2) in Figure 2 are set to zero, a, and $-b$ respectively. This approach generalizes in a straightforward manner to tasks split up into more than two legs: It is sufficient to create one transshipment vertex triple as in Figure 2 for each pair of consecutive subtasks.

Solving other types of VRPMSs

As mentioned above, the VRPTT features all types of synchronization requirements, and all important types of VRPMSs described in Table 2 are special cases of the VRPTT. Therefore, a heuristic capable of solving VRPTTs can be used to solve many other VRPMSs, too. As Ropke and Pisinger [46] and Vidal et al. [50] have shown, generic vehicle routing heuristics can be capable of obtaining extremely good results for several different problem types. Hence, a heuristic for the VRPTT is, in principle, a possible solution approach for other VRPMSs. This section describes how the procedure described in the previous section can be employed to this end.

VRPs with split pickup and/or split delivery

Problems with exclusively task and load synchronization, commonly referred to as *split delivery VRPs (SDVRPs),* are a rather special and very well-studied case of VRPMSs. In such problems, it is allowed that several vehicles visit a customer, each delivering (collecting) a part of the customer's demand (supply). Transshipments are not allowed, but load synchronization is necessary at the customer vertices. It is straightforward to see how such problems can be solved with the above heuristic: In step (i), the subtasks to be determined concern only the load splitting aspect when collecting supplies at customer vertices; transshipments locations are not considered. This means that procedures (i) to (iii) can be used for step (i) simply by relaxing the requirement that only one lorry may visit a customer; procedures (iv) to (vi) are not applicable. Step (iii) of the heuristic can simply be omitted. It is to be expected, however, that a tailored SDVRP procedure will lead to better results. For more information on the SDVRP or the pickup-and-delivery problem with split pickups and/or split deliveries, the reader is referred to Archetti and Speranza [4], Derigs et al. [13], Desaulniers [14], and Hennig et al. [24].

PDPTWs with or without transshipments

If the original problem that is transformed into a VRPTT is already a PDP, then approaches (i) and (v) for step (i) described above will work unmodified; (ii) and (iii) become multi-commodity network flow and service network design problems respectively; the necessary modifications are straightforward. (iv) can not be applied in this case. Instead, a hub location problem could be solved, cf. Alumur and Kara [2]. As for approach (vi), this author is unaware of work on a problem similar to the TTRP with pickup-and-delivery tasks. Again, step (iii) can be omitted. A related problem is the *dial-a-ride problem with transshipments (DARPT).* The DARP(T) is a PDPTW(T) occurring in passenger transport, i.e., the tasks correspond to transporting one or more persons from a pickup to a destination location, without or with the possibility to

change vehicles, i.e., without or with transshipments. A survey on the DARP is provided by Cordeau et al. [8]. For reasons of user convenience, in addition to PDPTW(T) constraints, it is usually required in DARP(T)s that the time between pickup and delivery of a person be limited to a maximal ride-time. Masson et al. have extended their results on the PDPTWT described in ref. [31] to ride-time constraints and provided an efficient feasibility check for this type of restriction. Other than that, the DARPT is equivalent to the PDPTWT, so that DARPTs can be solved with the PDPTWT heuristic of Masson et al. [30], augmented by the ride-time check of Masson et al., in step (ii).

Multi-echelon vehicle and location-routing problems

N-echelon problems can be solved with the above heuristic by considering, in step (i) (and/or, as mentioned above, within the PDPTWT heuristic used in step (ii)), the allowed ways of performing a task consisting of transporting some good from a central depot d to customer i via $N-1$ load transfers at transshipment locations. These different ways of performing a task $d \to i$ consist in all combinations of elementary tasks $d \to l^1$, $l^1 \to l^2$, ..., $l^{N-1} \to i$ with l^κ being a transshipment location at hierarchy level κ. Step (iii) is not necessary. (The above-mentioned ref. [21] presents a slightly different, one-step solution approach for NE-VRPs/LRPs with simultaneous o. s. For 2E-VRPs/LRPs with pure spatial o. s., there are considerably simpler procedures, cf. the survey in ref. [41].)

Simultaneous vehicle and crew routing and scheduling problems

The few papers on SVCRSPs published so far regard vehicles as inseparable units (i.e., either single lorries or lorry-trailer combinations that remain physically coupled together throughout the planning horizon) and do not consider the possibilities of split pickups/deliveries or of load transshipments. The issue in SVCRSPs is to perform movement synchronization between vehicles and drivers. The transshipment locations are only used as *relay stations* where no load transfers and no de- or recouplings of trailers occur, but where drivers may change vehicles. It is sensible to route the vehicles in step (ii) and the drivers in step (iii), because this allows driver changes also for non-empty vehicles, which is not possible when these steps are reversed, cf. ref. [19]. In step (ii), it must be ensured that lorries visit relay stations at all, which will not happen automatically, considering that such visits incur detours. To this end, the following approach is possible: Assume without loss that the planning horizon is p periods, each consisting of t time units (e.g., one working week of five days, each having 1440 minutes), and require that each lorry visit a relay station at least r times during the planning horizon. For each available lorry k (or, in the case of an unlimited fleet, for a sufficiently large number of lorries) with start depot d, create r artificial tasks r^k_ν, $\nu = 1, \ldots, r$, consisting in 'transporting a driver' from a location to another location. There are different ways of performing each of these tasks. The first artificial task can be fulfilled by 'transporting a driver' from the start depot to any relay station or the end depot, the other ones by 'transporting a driver' from any relay station to any other relay station or the end depot. The underlying network is such that the only arc that can be traversed by a lorry k leaving its start depot leads to the unique pickup location of r^k_1, which coincides with k's start depot. The only arc leaving a delivery location of an artificial task r^k_ν leads to a pickup location of $r^k_{\nu+1}$, or to the end depot in the case $\nu = r$. From pickup locations of artificial tasks, all customer task pickup and delivery vertices are reachable. The time window at a delivery vertex of artificial task r^k_ν is set to start at time zero and end at time $(p/r) \cdot \nu \cdot t$; alternatively, 'ride-time constraints' can be specified for the artificial tasks as described above for the DARPT (see page 12). Each artificial task can also be fulfilled at zero cost and time by visiting a special 'driver pickup and delivery task vertex' reachable from all lorry start depots and the delivery vertices of all r^k_ν, for $\nu \geq 1$ and for all k. All arcs emanating from this vertex lead to lorry end depots. In this way, all artificial tasks can and will be fulfilled by any feasible solution. Now, to solve an SVCRSP with the above heuristic, in step (i) ways of performing

tasks need be determined only for the artificial tasks, because for the 'real' tasks, there is only one way of performing them when no split pickup and no transshipments are allowed.
Note that the reason why transshipment locations are used when no transshipments are allowed is that drivers must take breaks and rests, whereas vehicles may operate around the clock. Modelling as well as algorithmic consideration of such driver rules can be extremely complicated and is beyond the scope of this paper. Pertinent references include Drexl and Prescott-Gagnon [18], Goel [20], Kok et al. [27], and Prescott-Gagnon et al. [40]. It shall be mentioned, however, that every paper on driver rules this author is aware of makes the unrealistic assumption that a break or rest can be taken anywhere en route, i.e., that, upon reaching a time limit, the driver can simply stop at the roadside. Moreover, beyond what is specified in national or supra-national (EU) law, there are numerous regulations and case law giving very detailed specifications on requirements for legal driver schedules. To this author's knowledge, none of these additional constraints has so far been considered in a paper on driver scheduling algorithms. Nevertheless, considering driver rules as it is done in the above references significantly increases the relevancy to practice of solutions to VRP instances and is an indispensable component of algorithms that are applied in commercial VRP software. If the above idea of ensuring visits to relay stations in step (ii) is used, it is sensible to consider (some) driver rules already in step (ii), cf. ref. [19].

The Meisel-Kopfer problem

The Meisel-Kopfer problem (MKP) ([33]) bears some similarity to the SVCRSP in that it requires exclusively task and movement synchronization. It can be described as follows: A set of full-load pickup-and-delivery tasks must be fulfilled using a fleet of autonomous and non-autonomous vehicles (henceforth referred to as lorries and trailers for simplicity). Fulfilling a task requires exactly one lorry and one trailer. Transshipments of load are not allowed, but it is possible that an empty trailer is delivered by a lorry k to the pickup location of a task for loading and is left there for later pickup by k or a different lorry. After having picked up a loaded trailer, a lorry k must move directly to the delivery location of the task currently loaded. There, the trailer may again be left, this time for unloading and later pickup by k or a different lorry.
The MKP can be solved with the above heuristic as follows. Step (i) is omitted. Step (ii) is a VRPTW for the non-autonomous vehicles, i.e., a special case of a PDPTWT. (A VRPTW can be transformed to a PDPTW(T) by replacing an original task of visiting customer i by a pickup-and-delivery task requiring a pickup at a depot and a delivery at i.) Step (iii) is a PDPTWT where each route of a non-autonomous vehicle is partitioned into segments in the following manner: The first segment is the transport of the non-autonomous vehicle from its depot to the pickup location of the first task, the second segment is the transport of the non-autonomous vehicle from the pickup location to the delivery location of the first task etc.

Personnel dispatching problems with spatio-temporal synchronization constraints

In such problems, usually no physical objects are transported between locations, but customers are visited to perform a service there, i.e., PDPSTSs are VRP type problems. Tasks requiring synchronization, i.e., where two persons with the same or different qualifications must visit the customer, can be represented as PDPTWT tasks as follows. Assuming that there is only one depot d (the extension to multiple depots is straightforward), the original task has one way of being performed: Perform $d \to i_1$ with a 'vehicle' with qualification A and $d \to i_2$ with a 'vehicle' with qualification B, i_1 and i_2 being two vertices corresponding to location/customer i. These tasks are replaced by two tasks $d \to i_1$ and $i_2 \to i_3$, requiring a vehicle with qualification A and B respectively, where i_1, i_2, and i_3 again correspond to location i. All arcs leaving i_1 have their travel time increased by the service/working time required for the person with qualification A for the original task at i. There is no arc (i_1, i_2). The only arc emanating from i_2 is (i_2, i_3). This arc has a travel time of zero plus the time required for the person with qualification B for the original task at i. With these constructs, all four types of operation synchronization

relationships, as mentioned on page 2 (pure spatial o. s., o. s. with precedences, simultaneous o. s., and exact deferred o. s.), can be required for $d \to i_1$ and $i_2 \to i_3$, and the extended constant-time feasibility check described in the section on routing the subtasks in the VRPTT heuristic can be applied (cf. page 10). This also works if, in deferred o. s., the same person may return later on, and also if both qualifications are of the same type.
Step (iii) of the VRPTT heuristic can be omitted in PSPSTSs where all 'vehicles'/persons are autonomous.

The garage distribution problem

The *garage distribution problem (GDP)* occurs as a real-world problem in planning the distribution of finished car garages from factories throughout Germany to customer sites using lorries and trailers. Each vehicle can carry one garage at a time. Some lorries, but not all, are equipped with a crane behind the driver cabin. In addition, there are pure crane vehicles that cannot transport any load. To deliver garages to customers and to transfer garages at transshipment locations, an appropriately equipped lorry or a pure crane vehicle must be present. This means that, if a garage is to be transferred between two vehicles, but neither has a crane, a third vehicle must be present. Thus, the GDP is a PDPTW with trailers, transshipments, and *support vehicles* (vehicles with zero capacity, cf. refs. [44], [16]; here, the pure crane vehicles) where each task requires and each vehicle offers a capacity of one, and where simultaneous operation synchronization of *three* vehicles is relevant. This can be modelled as follows: If it has been decided that an original task $i \to j$ should be transshipped at l, the way of performing the task consists in the three subtasks $i \to l_1$, $l_1 \to l_2$, and $l_2 \to j$, where l_1, l_2, and l_3 correspond to l. The first and third subtask have the usual meaning. The second subtask represents the activity 'pick up (lift) the garage from the vehicle that has transported it to l and deliver (drop) it onto the vehicle that will bring it from l to j'. Note that this approach of decomposing the simultaneous interaction of $k = 3$ vehicles generalizes to arbitrary $k \geq 2$.

Summary and outlook

This paper has presented a heuristic for solving the vehicle routing problem with trailers and transshipments, an archetype of a VRP with multiple synchronization constraints. The heuristic consists of three steps that can be performed iteratively. It appropriately addresses the central difficulties of solving VRPMSs and offers great flexibility regarding the concrete implementation of each step. In addition, it has been shown how many other types of VRPMSs can be solved by the heuristic, without any algorithmic modifications, simply by adequate representation of the respective VRPMS instance.
The next steps are of course to provide a concrete implementation of the heuristic and perform computational tests with different VRPMS instances. Moreover, the idea of 'layering' the routing problems for tasks, capacities, and autonomous vehicles, as presented in the heuristic, can be extended further. For example, the above-mentioned simultaneous vehicle and crew routing and scheduling problem (SVCRSP) can be extended to allow transshipments of the tasks, to consider a vehicle fleet consisting of lorries and trailers, and to take into account driver shuttle transports between the relay stations (cf. ref. [19]). This leads to a five-step procedure (determine ways of performing the tasks, then route sub-tasks, capacities, vehicles, drivers and shuttles in this order). Finally, existing mixed-integer programming formulations for the VRPTT use (arc or path) variables for vehicles. It might be a good idea to transfer this layering approach and to employ variables for capacities (of trailers as well as lorries) and variables for (autonomous) lorries, and to couple these variable types by appropriate constraints.

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

References

[1] Ahuja R, Magnanti T, Orlin J (1993):
Network Flows
Prentice-Hall, Upper Saddle River

[2] Alumur S, Kara B (2008):
Network Hub Location Problems: The State of the Art
European Journal of Operational Research 190(1): 1–21

[3] Archetti C, Bianchessi N, Grazia Speranza M (2014):
Branch-and-Cut Algorithms for the Split Delivery Vehicle Routing Problem
European Journal of Operational Research 238(3): 685–698

[4] Archetti C, Speranza M (2008):
The Split Delivery Vehicle Routing Problem: A Survey
in:
Golden B, Raghavan S, Wasil E (eds):
The Vehicle Routing Problem: Latest Advances and New Challenges
Springer, New York
103–122

[5] Baldacci R, Bartolini E, Laporte G (2010):
Some Applications of the Generalized Vehicle Routing Problem
Journal of the Operational Research Society 61: 1072–1077

[6] Blais M, Laporte G (2003):
Exact Solution of the Generalized Routing Problem Through Graph Transformations
Journal of the Operational Research Society 54: 906–910

[7] Bock S (2010):
Real-time Control of Freight Forwarder Transportation Networks by Integrating Multimodal Transport Chains
European Journal of Operational Research 200: 733–746

[8] Cordeau JF, Laporte G, Potvin JY, Savelsbergh M (2007):
Transportation on Demand
in:
Barnhart C, Laporte G (eds):
Transportation
Handbooks in Operations Research and Management Science 14
Elsevier, Amsterdam
429–466

[9] Daskin M (1995):
Network and Discrete Location
Wiley, New York

[10] De Rosa B, Improta G, Ghiani G, Musmanno R (2002):
The Arc Routing and Scheduling Problem with Transshipment
Transportation Science 36: 302–313

[11] Del Pia A, Filippi C (2006):
A Variable Neighborhood Descent Algorithm for a Real Waste Collection Problem with Mobile Depots
International Transactions in Operational Research 13: 125–141

[12] Derigs U, Döhmer T (2008):
Indirect Search for the Vehicle Routing Problem with Pickup and Delivery and Time Windows
OR Spectrum 30: 149–165

[13] Derigs U, Li B, Vogel U (2010):
Local Search-Based Metaheuristics for the Split Delivery Vehicle Routing Problem
Journal of the Operational Research Society 61: 1356–1364

[14] Desaulniers G (2010):
Branch-and-Price-and-Cut for the Split-Delivery Vehicle Routing Problem with Time Windows
Operations Research 58: 179–192

[15] 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`

[16] Drexl M (2012):
Synchronization in Vehicle Routing—A Survey of VRPs with Multiple Synchronization Constraints
Transportation Science 46(3): 297–316

[17] Drexl M (2013):
Applications of the Vehicle Routing Problem with Trailers and Transshipments
European Journal of Operational Research 227(2): 275–283

[18] Drexl M, Prescott-Gagnon E (2010):
Labelling Algorithms for the Elementary Shortest Path Problem with Resource Constraints Considering EU Drivers' Rules
Logistics Research 2: 79–96

[19] Drexl M, Rieck J, Sigl T, Press B (2013):
Simultaneous Vehicle and Crew Routing and Scheduling for Partial- and Full-Load Long-Distance Road Transport
BuR – Business Research 6(2): 242–264

[20] Goel A (2010):
Truck Driver Scheduling in the European Union
Transportation Science 44: 429–441

[21] Grangier P, Gendreau M, Lehuédé F, Rousseau LM (2014):
An Adaptive Large Neighborhood Search for the Two-Echelon Multiple-Trip Vehicle Routing Problem with Satellite Synchronization
Technical Report CIRRELT-2014-33, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport

[22] Grünert T, Sebastian HJ (2000):
Planning Models for Long-Haul Operations of Postal and Express Shipment Companies
European Journal of Operational Research 122(2): 289–309

[23] Hemmelmayr V, Cordeau JF, Crainic TG (2012):
An Adaptive Large Neighborhood Search Heuristic for Two-Echelon Vehicle Routing Problems Arising in City Logistics
Computers & Operations Research 39(12): 3215–3228

[24] Hennig F, Nygreen B, Lübbecke M (2012):
Nested Column Generation Applied to the Crude Oil Tanker Routing and Scheduling Problem with Split Pickup and Split Delivery
Naval Research Logistics 59(3–4): 298–310

[25] Irnich S, Funke B, Grünert T (2006):
Sequential Search and its Application to Vehicle-Routing Problems
Computers & Operations Research 33: 2405–2429

[26] Karapetyan D, Gutin G (2012):
Efficient Local Search Algorithms for Known and New Neighborhoods for the Generalized Traveling Salesman Problem
European Journal of Operational Research 219: 234–251

[27] Kok A, Meyer C, Kopfer H, Schutten J (2010):
A Dynamic Programming Heuristic for the Vehicle Routing Problem with Time Windows and European Community Social Legislation
Transportation Science 44: 442–454

[28] Li Y, Lim A, Rodrigues B (2005):
Manpower Allocation with Time Windows and Job-Teaming Constraints
Naval Research Logistics 52: 302–311

[29] Mankowska D, Meisel F, Bierwirth C (2014):
The Home Health Care Routing and Scheduling Problem with Interdependent Services
Health Care Management Science 17(1): 15–30

[30] Masson R, Lehuédé F, Péton O (2013):
An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Transfers
Transportation Science 47(3): 344–355

[31] Masson R, Lehuédé F, Péton O (2013):
Efficient Feasibility Testing for Request Insertion in the Pickup and Delivery Problem with Transfers
Operations Research Letters 41(3): 211–215

[32] Masson R, Lehuédé F, Péton O (2014):
The Dial-A-Ride Problem With Transfers
Computers & Operations Research 41: 12–23

[33] Meisel F, Kopfer H (2014):
Synchronized Routing of Active and Passive Means of Transport
OR Spectrum 36(2): 297–322

[34] Mitrović-Minić S, Laporte G (2006):
The Pickup and Delivery Problem with Time Windows and Transshipment
INFOR 44: 217–227

[35] Moccia L, Cordeau JF, Laporte G (2012):
An Incremental Tabu Search Heuristic for the Generalized Vehicle Routing Problem with Time Windows
Journal of the Operational Research Society 63(2): 232–244

[36] Mues C, Pickl S (2005):
Transshipment and Time Windows in Vehicle Routing
in:
Proceedings of the 8th International Symposium on Parallel Architectures, Algorithms and Networks
IEEE, Piscataway
113–119

[37] Nguyen V, Prins C, Prodhon C (2012):
Solving the Two-Echelon Location Routing Problem by a GRASP reinforced by a Learning Process and Path Relinking
European Journal of Operational Research 216(1): 113–126

[38] Nishi T, Izuno T (2014):
Column Generation Heuristics for Ship Routing and Scheduling Problems in Crude Oil Transportation with Split Deliveries
Computers & Chemical Engineering 60: 329–338

[39] Oertel P (2000):
Routing with Reloads
PhD Thesis, Faculty of Mathematics and Natural Sciences, University of Cologne

[40] Prescott-Gagnon E, Desaulniers G, Drexl M, Rousseau LM (2010):
European Driver Rules in Vehicle Routing with Time Windows
Transportation Science 44: 455–473

[41] Prodhon C, Prins C (2014):
A Survey of Recent Research on Location-Routing Problems
European Journal of Operational Research 238(1): 1–17

[42] Qu Y, Bard J (2012):
A GRASP with Adaptive Large Neighborhood Search for Pickup and Delivery Problems with Transshipment
Computers & Operations Research 39(10): 2439–2456

[43] Rasmussen M, Justesen T, Dohn A, Larsen J (2012):
The Home Care Crew Scheduling Problem: Preference-Based Visit Clustering and Temporal Dependencies
European Journal of Operational Research 219: 598–610

[44] Rivers C (2002):
Coordination in Vehicle Routing
PhD Thesis, Massey University

[45] Ropke S, Pisinger D (2006):
An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows
Transportation Science 40: 455–472

[46] Ropke S, Pisinger D (2006):
A Unified Heuristic for a Large Class of Vehicle Routing Problems with Backhauls
European Journal of Operational Research 171: 750–775

[47] Savelsbergh M (1992):
The Vehicle Routing Problem with Time Windows: Minimizing Route Duration
ORSA Journal on Computing 4: 146–154

[48] Shang J, Cuff C (1996):
Multicriteria Pickup and Delivery Problem with Transfer Opportunity
Computers & Industrial Engineering 30: 631–645

[49] Thangiah S, Fergany A, Awan S (2007):
Real-Time Split-Delivery Pickup and Delivery Time Window Problems with Transfers
Central European Journal of Operations Research 15: 329–349

[50] Vidal T, Crainic TG, Gendreau M, Prins C (2014):
A Unified Solution Framework for Multi-Attribute Vehicle Routing Problems
European Journal of Operational Research 234(3): 658–673