Gutenberg School of Management and Economics

Discussion Paper Series

Simultaneous Vehicle and Crew Routing and Scheduling for Partial and Full Load Long-Distance Road Transport

Michael Drexl, Julia Rieck, Thomas Sigl and Bettina Berning

August 2011

Discussion paper number 1112

Johannes Gutenberg University Mainz
Gutenberg School of Management and Economics
Jakob-Welder-Weg 9
55128 Mainz
Germany
wiwi.uni-mainz.de

Contact details

Michael Drexl
Chair of Logistics Management
Johannes Gutenberg University Mainz
Jakob-Welder-Weg 9
55128 Mainz
Germany

drexl@uni-mainz.de

Julia Rieck
Operations Research Group
Institute of Management and Economics
Clausthal University of Technology
Julius-Albert-Straße 2
38678 Clausthal-Zellerfeld
Germany

julia.rieck@tu-clausthal.de

Thomas Sigl
Fraunhofer Centre for Applied Research on Supply Chain Services SCS
Nordostpark 93
90411 Nuremberg
Germany

thomas.sigl@scs.fraunhofer.de

Bettina Berning
Fraunhofer Centre for Applied Research on Supply Chain Services SCS
Nordostpark 93
90411 Nuremberg
Germany

bettina.berning@scs.fraunhofer.de

Simultaneous Vehicle and Crew Routing and Scheduling for Partial and Full Load Long-Distance Road Transport

Technical Report LM-2011-05

Michael Drexl
Chair of Logistics Management, Gutenberg School of Management and Economics,
Johannes Gutenberg University Mainz
and
Fraunhofer Centre for Applied Research on Supply Chain Services SCS, Nuremberg

Julia Rieck
Operations Research Group
Institute of Management and Economics, Clausthal University of Technology

Thomas Sigl, Bettina Berning
Fraunhofer Centre for Applied Research on Supply Chain Services SCS, Nuremberg

29th August 2011

Abstract

This paper studies a simultaneous vehicle and crew routing and scheduling problem arising in long-distance road transport: Pickup-and-delivery requests have to be fulfilled over a multi-period planning horizon by a heterogeneous fleet of lorries and drivers. Typically, in the vehicle routing literature, a fixed assignment of a driver to a lorry is assumed. In our approach, we abandon this assumption and allow lorry/driver changes at geographically dispersed relay stations. This leads to intricate interdependencies between lorries and drivers and requires the synchronization of their routes. A solution heuristic based on a two-stage decomposition of the problem is developed, and computational experiments using real-world data provided by a major forwarder are presented and analysed.
Keywords: Simultaneous vehicle and crew routing and scheduling, Synchronization, Advanced truckload, Slipseating

1 Introduction

Vehicle routing problems (VRPs) are among the most studied mathematical optimization problems (see, for example, the monographs of Toth and Vigo [26] and Golden et al. [12]). This interest results from the importance of vehicle routing in everyday logistics practice as well as the intellectual challenge of finding feasible or optimal VRP solutions; VRPs belong to the class of NP-hard optimization problems (Lenstra and Rinnooy Kan [19]). The widespread practical use of mathematical models and algorithms for the solution of VRPs, as reflected by the large number of vendors of commercial vehicle routing software, shows that the efforts of science in this field have had a significant impact on business operations. One application area, however, where there is a large gap between practical requirements and pertinent scientific work is simultaneous vehicle and crew *routing and* scheduling. This term denotes the situation where the required transports *have no given timetable/no fixed schedule*, and where a driver-vehicle combination is

not considered an inseparable unit anymore. Thus, routes to perform the required transports have to be planned for both vehicles and drivers. This is an important issue in practice, not only because of the simple fact that drivers regularly need breaks and rests, whereas vehicles basically can be used twenty-four hours a day, but also because of the existence of social legislation or trade union rules regarding driving, break, and rest times for drivers. The large majority of papers on vehicle routing, nonetheless, does not distinguish between a vehicle and its driver. In their monograph on VRPs, for example, [26] state that throughout, 'the constraints imposed on drivers are imbedded in those associated with the corresponding vehicles'. Hence, the contribution of the present paper is to study how separate, but synchronized, routes for drivers and vehicles can be computed for transports without a given timetable.
The remainder of the paper is structured as follows. In the next section, we give a description of the proposed simultaneous vehicle and crew routing and scheduling problem (SVCRSP). Then, in Section 3, the relevant literature is briefly surveyed. Section 4 points out the distinguishing features of the present problem. In Section 5, a two-stage heuristic algorithm for solving real-world instances of the SVCRSP is described. Computational experiments with the algorithm are reported in Section 6. The paper closes with a summary and a research outlook in Section 7.

2 Problem description

In order to narrow the above-mentioned gap, we studied the following simultaneous vehicle and crew routing and scheduling problem. Partial and complete loads (henceforth referred to as requests) have to be transported by lorry from pickup to delivery locations. There is a time window for the pickup and for the delivery of each request. For the transport of the requests, a fleet of lorries and a set of drivers are available. Each driver and each lorry has a fixed home depot, where he/it is positioned at the beginning of the planning horizon and to where he/it must return at the end of the planning horizon. The planning horizon spans one calendar week, that is, seven days from Monday to Sunday. Thus, a driver/lorry route starts and ends at an assigned depot and may cover several days. The lorry fleet is heterogeneous; there are several types of lorry differing with respect to costs, capacity, and ability to perform certain requests. The drivers are homogeneous; each driver is able to drive each lorry and receives the same wage. In addition to the home depots and the pickup and delivery locations, there are so-called intermediate depots or *relay stations*. It is possible that a location acts as both a home depot and a relay station. At relay stations, and only there, drivers can change lorries. Hence, there is a free assignment of lorries to drivers and vice versa, that is, a driver may change lorries arbitrarily often within the planning horizon, respectively, a lorry may be driven by arbitrarily many drivers within the planning horizon. The fact that there are multiple depots, where drivers and lorries may be stationed, and relay stations, where a change of lorry or driver is possible, makes the synchronization of drivers and lorries a non-trivial issue. In addition to the requirements already stated, the drivers have to fulfil the European Union social legislation on driver driving and working hours on single-manned vehicles. It is assumed that a driver can take a break or rest anywhere en route. However, a driver must visit a relay station and take a daily rest there, or he must finish his route at his home depot, upon expiry of a given fixed period after leaving his home depot or ending a daily rest at a previous relay station. Furthermore, we consider the option of driver shuttle transports, where small shuttle vans (whose drivers are not subject to driver rules) are used to transport lorry drivers between relay stations. In doing so, a lorry driver is able to leave a lorry at a relay station and subsequently drive a lorry starting from a different relay station. The shuttle transports are not counted as driving or working time for the lorry drivers. The possibility of shuttle transports increases the flexibility for matching lorries and drivers, but of course shuttle transports also induce costs. A typical objective for the SVCRSP is to minimize the overall operating costs, which consist of lorry, driver, and shuttle costs.

To fulfil a request, two different objects (a driver and a lorry) must be synchronized in space and time, since both are non-autonomous objects unable to move in space on their own. (Essentially, both the lorries and the drivers can be regarded as 'vehicles'.) From these synchronization requirements results the central additional difficulty of the SVCRSP compared to classical VRPs, the so-called *interdependence problem*:

The interdependence problem refers to the fact that a change in one route may have effects on the feasibility of other routes.

The following *gedankenexperiment* illustrates this point: Consider a partial or complete SVCRSP solution, that is, a set of lorry and driver routes. Let the route of a driver d be such that he drives two different lorries, say, k_1 and k_2 (in this order), and that he changes lorries at relay station r. Now assume that the route of lorry k_1 is modified (for example, by inserting a request) so that k_1 reaches relay station r later than before, and that *the resulting route of* k_1 *is feasible.* The direct implication is that also driver d reaches r later. Consequently, the departure of d from r with lorry k_2 might be deferred. This, of course, might lead to the violation of time windows and, hence, *the routes of* d *and* k_2 *may become infeasible.* These effects may propagate further, to drivers driving lorries k_1 and k_2 after d, and from these drivers to other lorries etc. Hence, it is easy to imagine that, in the worst case, a change in the route of *one* lorry or *one* driver might make *all* other lorry and driver routes infeasible.

This is in marked contrast to classical VRPs without interdependence problem, where any operation modifying a subset of all existing routes leaves the remaining routes unaffected. Indeed, most VRP solution algorithms, notably local search procedures and approaches based on column generation, implicitly assume independence of routes. Consequently, standard VRP solution procedures known from the literature are not directly applicable to solving SVCRPs. Hence, it is necessary to use mechanisms for dealing with the synchronization requirements and the resulting interdependence problem.

3 Literature review

The most important papers influencing our research are briefly discussed in the following. These works may be partitioned into two classes: First, papers considering cases where motor vehicles and drivers are synchronized at a central depot and where a change of driver/vehicle is possible only at the depot. Second, papers where different types of elementary autonomous and/or non-autonomous object are required to fulfil tasks, and where these objects may join and separate at several different locations. Compared to the first class, the locational flexibility of the second class adds an additional degree of freedom. Hence, problems in the second class turn out to be significantly harder to solve than those in the first class.

Synchronization at a central depot. Laurent and Hao [18] consider the problem of simultaneously scheduling vehicles and drivers for a limousine rental company. The required transports are pickup-and-delivery trips with given time windows. The authors use a two-stage solution approach which aims to find a feasible crew and vehicle schedule by assigning a driver-limousine pair to each trip. First, an initial feasible solution is constructed by means of a greedy heuristic similar to the well-known best-fit-decreasing strategy for the bin packing problem, using constraint programming techniques for domain reduction. Second, an improvement procedure based on local search embedded in a simulated annealing metaheuristic is performed.

Prescott-Gagnon et al. [21] study the problem of planning oil deliveries to customers by lorry. To solve the problem, the authors develop three metaheuristics, a tabu search (TS) algorithm, a large neighbourhood search (LNS) heuristic based on this TS algorithm and another LNS heuristic based on a column generation (CG) heuristic which uses the TS algorithm to generate columns. They use a greedy construction heuristic which sequentially builds up routes for driver-lorry pairs by inserting the temporally closest customer. In the destruction phase of the LNS algorithm, different heuristics similar to those presented in Røpke [22] are used to determine the vertices to be removed from the routes of the current solution. The reconstruction phase applies either TS or CG. Among other move types, the TS procedure employs a driver switch move which tries to switch a pair of drivers, that is, have one driver drive the other driver's route and vice versa.

Xiang et al. [28] describe a static dial-a-ride problem which involves the scheduling of heterogeneous vehicles and a group of drivers with different qualifications. The solution procedure is composed of a construction phase to obtain an initial solution, an improvement phase, and an intensification phase to fine-tune the solution. The important aspect of the procedure is that, initially, 'abstract' routes with a fixed schedule are determined, and only in the last stage, concrete vehicles and drivers are assigned to the routes.

Zäpfel and Bögl [29] consider an application of local letter mail distribution. Pickup routes and delivery routes (but no combined pickup-and-delivery routes) have to be planned within a planning horizon of one week. In pickup routes, outbound shipments are transported from local post offices to a letter mail distribution centre. Conversely, in delivery routes, shipments are transported from the distribution centre to post offices. Schedules are planned for both drivers and vehicles, taking into account European Union social legislation. The problem is solved heuristically, by decomposing it into a generalized VRP with time windows (GVRPTW) and a 'personnel assignment problem' (PAP). First, a feasible solution to the GVRPTW is computed via a modification of the I1 heuristic by Solomon [24]. Then, the PAP tries to find a feasible driver assignment for the GVRPTW solution. The assignment is achieved by creating a table with all feasible combinations of drivers and routes. Each table entry represents the costs resulting from the driver performing the route. A complete personal assignment is computed, using three different strategies, among them a greedy and a random procedure. After that, an improvement procedure embedded into a metaheuristic follows.

Synchronization en route. Kim et al. [15] study a combined vehicle routing and staff scheduling problem where a certain number of tasks has to be fulfilled in a fixed sequence at customers. Among the tasks, an end-to-start relationship is assumed. In order to fulfil the tasks, different teams of workers are available. Each team is qualified to perform one specific type of task. The teams cannot move by themselves; instead, a set of vehicles is used to transport the teams. There is no fixed assignment of a vehicle to a team, and each vehicle may carry at most one team at a time. The authors develop an astonishingly simple procedure, in which the vehicles, the teams, and the next tasks for each customer are stored in three lists, along with the relevant information on times and locations. In each iteration, a triplet (vehicle, team, task) is selected from the lists, using a best-fit criterion. Then, the lists are updated to reflect the situation resulting when the selected vehicle transports the selected team to the location of the selected task.

Hollis et al. [13] describe a simultaneous vehicle and crew routing and scheduling application for urban letter mail distribution at Australia Post. The application characteristics are similar to those of our SVCRSP. Hollis et al. are the first to consider a problem with multiple depots, where vehicles and drivers may be stationed and interchanged. The authors use a two-stage approach. In the first stage, they determine 'abstract' vehicle routes by solving a pickup-and-delivery problem with time windows, multiple depots, a heterogeneous fleet of vehicles as well as several working time restrictions for drivers. A path-based mixed-integer programming (MIP) model is presented and solved by heuristic column generation. In the second stage, concrete vehicle and crew schedules are determined taking an integrated vehicle and crew *scheduling* approach. This is again done by solving an MIP with heuristic column generation. In the second-stage MIP, the tasks to be performed correspond to the vehicle routes computed in the first stage.

It is noteworthy that, although the objects to be synchronized in the papers described are vehicles and drivers, the concrete application contexts are almost all different (ranging from limousine rental to mail distribution). As regards solution approaches, and, in particular, the consideration of the interdependence problem, [29] perform, in principle, a decomposition of the problem by object type, exploiting the fact that drivers and vehicles may join and separate at one location only. On the other hand, [18] and [21] take the opposite way and compute routes for predetermined driver-vehicle pairs. [28] and [13] determine abstract routes first, and assign concrete vehicles and drivers afterwards. [15] use a heuristic in which a direct selection of vehicles, teams, and tasks is performed.

A recurring idea in most solution procedures described above is to decompose the problem into several stages and, in early stages, partially take aspects into account which are important to be able to obtain feasible solutions in later stages. Our solution approach, described in Section 5, also follows this principle.

As described in Section 2, the problem under consideration here is an SVCRSP in which European Union social legislation on drivers' driving and working times (driver rules for short) are considered. The relevant rules are briefly described in Section 5. For detailed information on these regulations and procedures for considering them in VRP algorithms, the reader is referred to Goel [10], Drexl and Prescott-Gagnon [8], Goel [11], Kok et al. [17], and Prescott-Gagnon et al. [20]. Our SVCRSP algorithm uses the procedure described in the last reference.

The SVCRSP is a special case of the broader class of *vehicle routing problems with multiple synchronization constraints (VRPMSs)*. These are VRPs where the routes of different 'vehicles' are interdependent. For a classification of VRPMSs and a comprehensive overview of the existing literature on synchronization in vehicle routing, including detailed discussions of the papers cited above, the reader is referred to Drexl [7].

A related and well studied field of application is integrated vehicle and crew *scheduling*, for example in the airline business or in the public transport sector. Pertinent surveys are given in Klabjan [16], Caprara et al. [3], and Desaulniers and Hickman [5]. However, the models and solution approaches used for solving such problems are always based on the fundamental assumption of a *given timetable* for the requests/tasks. Since we assume here that the required transports do *not* have a given timetable, the SVCRSP requires different or at least modified solution methods.

4 Important features of the problem

The important and distinguishing features of the present problem compared to the ones presented in the references discussed above lie in the consideration of the following criteria:

- Practical and real-life assumptions:
  - Shipments/requests may originate and end anywhere, not only at lorry or driver depots.
  - A route segment between two relay stations may be driven by any driver.
  - A lorry need not be empty when a driver change is performed at a relay station.
  - Driver shuttle transports to and from relay stations are possible.
  - European driver rules are considered completely and correctly for a planning horizon spanning one week.
- Components of the solution approach:
  - The algorithm is similar to the one taken by [13], and is also based on a heuristic decomposition of the overall problem into two stages. However, compared to the highly sophisticated approaches taken in some of the above papers, our algorithm is rather simple and straightforward and primarily relies on an appropriate network representation of the problem.
  - Both stages can be solved with essentially the same algorithm.
  - The large neighbourhood search heuristic we use can very easily be replaced by any other metaheuristic.

5 Solution approach

As mentioned, the basic idea of our solution algorithm is to take a two-stage approach. In the first stage, routes for lorries are determined, taking into account some driver rules. In the second stage, routes for drivers are computed, based on the lorry routes from stage 1 and taking into account the remaining driver rules to ensure feasibility. We thus adopt some ideas from [13], but in our first stage, routes are computed for concrete lorries.
More precisely, the first stage consists in the solution of a pickup-and-delivery problem with time windows, relay stations, and additional constraints (PDPTWRS). The vehicles in the PDPTWRS correspond to the lorries of the underlying SVCRSP. There are multiple depots, namely, the home depots of the lorries. Each lorry performs at most one route, and the maximum route duration is set to the length of the planning horizon, which equals 5–6 days. The maximum time between two consecutive visits at a depot or a relay station is variable and was set to 37 and 48 hours in the computational experiments. For each lorry, driver rules are considered in a manner described in detail below. At the end of the planning horizon, each lorry must return empty to its home depot.
The second stage consists in the solution of a vehicle routing problem with time windows and multiple depots (VRPTWMD). In this problem, the partial lorry routes or segments starting and/or ending at a home depot or a relay station, as determined in the first stage, are the customers. The drivers act as the vehicles. Each driver drives at most one route, and the maximum route duration is again set to the length of the planning horizon. Figure 1 shows the relationship between the two stages based on an example instance.

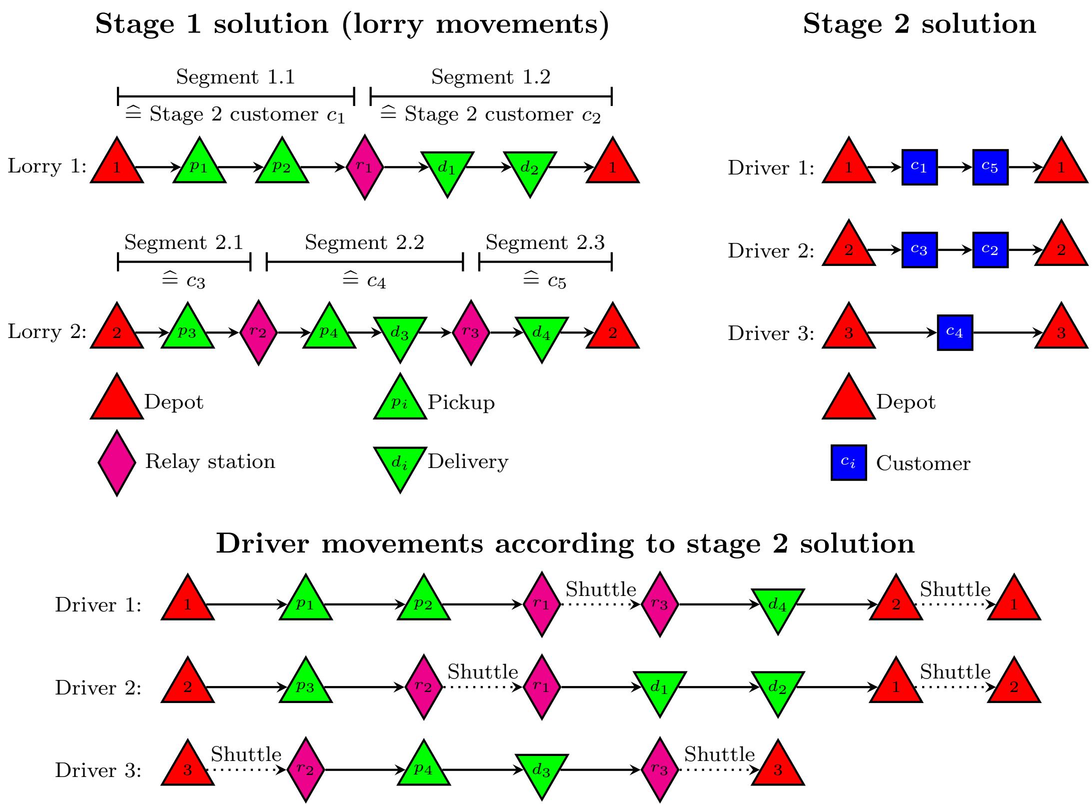


Figure 1: Relationship between stages 1 and 2

We consider an instance with three depots, three relay stations, and four pickup-and-delivery requests. A possible solution of stage 1, depicted on the left side of the upper part of the figure, consists of two lorry routes. The route of lorry 1 starts at the first depot, picks up the two requests 1 and 2, visits relay station r_1, delivers the two requests 1 and 2, and returns to the depot. Consequently, the route is comprised of two segments. The first one, segment 1.1, goes from depot 1 to relay station r_1, and yields customer c_1 for stage 2 of the solution procedure. The second segment goes from relay station r_1 back to the depot, and yields customer c_2 for stage 2. Similarly, the route of lorry 2 consists of three segments and, hence, yields three customers for stage 2. A possible solution of stage 2, depicted on the right side of the upper part of the figure, contains three driver routes. Driver 1 starts at depot 1, 'visits' the two 'customers' c_1 and c_5, and returns to the depot. Driver 2 starts at depot 2, visits customers c_3 and c_2, and returns. Finally, driver 3 starts at depot 3, visits customer c_4, and returns. Each shape on the left hand side corresponds to a physical location, and the depicted routes show the exact movements of the two lorries. On the right hand side, only the red triangles correspond to a physical location. The blue rectangles represent, in compact form, a movement in space. The lower part of the figure illustrates the exact movements of the drivers, that is, 'extracts' the route information hidden in the stage 2 customers. There, again, all shapes correspond to physical locations.

Note that the idea to model route segments as customers in a VRPTW was previously used in the already mentioned paper by Hollis et al. as well as in Fügenschuh [9]. In the latter paper, however, there is no simultaneous vehicle and driver routing and scheduling and thus no stage 1.

In both stages, we apply an implementation of a large neighbourhood search heuristic. Essentially, in such methods, elements of a solution are alternately removed (destruction step) and reinserted (reconstruction step) in order to improve a given solution. In the context of VRPs, given a complete route plan, a subset of customers/requests is removed from their respective routes in the destruction step and reinserted into the resulting partial routes in the reconstruction step. Our implementation is described in detail in Sigl [23]. It uses a large number of destruction heuristics, among them those described in Røpke [22]. The (re)construction heuristics used are different types of (parallel) insertion procedures. The implementation can easily be adapted, or, rather, instantiated, to solve PDPTWRSs as well as VRPTWMDs.

Stage 1: Determination of lorry routes

The determination of lorry routes in the course of the PDPTWRS in the first stage was based on an appropriate network representation. In addition, the consideration of driver rules and relay stations required modelling efforts.

Construction of the network

We consider a network $D_1 = (V_1, A_1)$, where V_1 is the set of vertices and A_1 is the set of arcs. The vertex set $V_1 = S \cup E \cup P \cup D \cup R$ consists of vertices for start and end depots in subsets S and E respectively, vertices for pickup and delivery in subsets P and D, and vertices for relay stations in subset R. ('$A \cup B$' denotes the union of the disjoint sets A and B.) This means that if a physical depot location may be used as a relay station, this location is represented by three different vertices in the network. The start of service at any vertex $i \in V$ must be within a prescribed time window $[a_i, b_i]$. Depot and relay station vertices have time windows starting Monday, 0:00 hours, and ending Saturday, 24:00 hours. For each request, a pickup and a delivery vertex with associated time windows and service times are given. The service must begin, but need not be finished, within the time window. The arc set A_1 is composed as follows (where l_i denotes the physical location corresponding to vertex $i \in V_1$):

(i) From each start depot vertex i, there is one arc to the corresponding end depot vertex (for unused lorries), to each relay station vertex $j \in R$ (since it may be sensible or necessary to visit a relay station before picking up the first request), and to each pickup vertex.

(ii) There is an arc between each pair of relay station vertices, from each relay station vertex $i \in R$ to every pickup and every delivery vertex, and to each end depot vertex $j \in E$.

(iii) Finally, there is an arc between each pair of request vertices (except for an arc from the delivery vertex of a request to the corresponding pickup vertex), an arc from each request vertex to each relay station vertex, and there is an arc from each delivery vertex to each end depot vertex.

Each arc (i, j) has an associated distance $d_{ij} \geqq 0$ and travel time $t_{ij} \geqq 0$. These weights correspond to the real-world distance and travel time by lorry between the respective locations and are assumed to be equal for each lorry type. The travel time indicates the pure driving time, disregarding any breaks or rests. Of course, only arcs (i, j) with $a_i + t_{ij} \leqq b_j$ are introduced.
An example network is depicted in Figure 2. The network represents an instance with one depot location, three relay station locations, and two pickup-and-delivery requests. Consequently, the network contains one start depot vertex, one end depot vertex, three relay station vertices, and two pickup and two delivery vertices. To keep the figure clear and concise, there is only one arc for each arc type described in items (i)–(iii). For example, the absence of an arc from vertex p_1 to vertex d_1 does not mean that there is no such arc in the network.

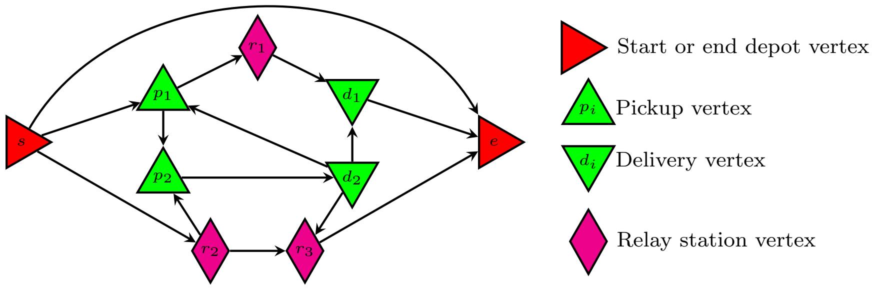


Figure 2: Example network for stage 1

Consideration of driver rules

In order to comply with the legislation on driver driving and working hours, the following rules must be considered:

- A break of at least 45 minutes is required
  - after 4.5 hours of driving, or
  - after 6 hours of work.
- A daily rest of at least 11 hours is required
  - after 9 hours of driving, or
  - after 9.6 hours of work (required by our project partner, not legally binding), or
  - 13 hours of wall-clock time after the end of the last daily rest.
- A weekly rest of at least 24 hours must be taken
  - after 56 hours of driving, or
  - after 48 hours of working (required by our project partner, not legally binding), or
  - 144 hours (6 days) of wall-clock time after the end of the last weekly rest.

'Work' encompasses driving as well as complementary activities such as loading or unloading, paperwork etc. Since driving time is always working time, but not vice versa, the weekly driving time is never restrictive. Note that the law makes no requirements with respect to the location where breaks or rests must be taken; these may be taken anywhere en route, such as on customer premises, at public parking places, or even at the roadside. All the references on driver rules cited in the literature review above assume that breaks or rests can be taken directly when scheduled and without incurring any detour to or from a suitable location. That is, it is essentially assumed that the driver simply stops at the roadside to take a break or rest if he is away from a depot or a customer location. By requiring that a daily rest must be taken upon visiting a relay station and specifying a maximum time period for two subsequent visits at a relay station, but otherwise allowing breaks or rests anywhere, we follow this established approach.
In the PDPTWRS, driver rules are considered as follows. The relevant resources are the time since the end of the last daily rest, the daily and interval driving time, and the daily and interval working time. At this, the daily driving (working) time is the total accumulated driving

(working) time between two daily rests and the interval driving (working) time is the total accumulated driving (working) time between two breaks. At the beginning of the planning horizon, each lorry is located at its home depot, and all driver time resources are completely reset. Then, routes are determined for the lorries by parallel insertion heuristics. The routes start at the depot vertices and visit pickup and delivery vertices of compatible requests, maintaining the capacity constraints of the lorries, the time windows of the requests, the driver rules resources, and the time period until the next visit at a relay station. An important feature of the approach is that the lorries need not be empty when visiting a relay station. Usually, at relay stations, a driver change is performed. For that reason, a lorry may leave a relay station after a short constant period of time after having reached the station, a so-called *relay time*. In the computational experiments, this time was set to 15 minutes. A visit at a relay station resets all driver resources, except for the time since the end of the last weekly rest and the weekly driving and working time; these resources are considered in the second stage of our solution procedure.

Within our insertion heuristics, a schedule considering the above-mentioned driver rules is computed so as to minimize overall route duration under the condition to arrive at the end depot vertex as early as possible. This is sensible when one and the same driver drives the entire route of a lorry. For each route, a complete scheduled sequence of activities (drive, wait, load, break, rest) is determined, indicating the earliest sensible beginning of each activity. This means that a concrete departure time from the start vertex of each segment is known.

Procedure for considering relay stations

The procedure for considering relay stations is a straightforward extension of the existing procedure for the PDPTW. In each iteration, after routes have been modified by adding, shifting, or removing requests in the (re)construction or destruction step, a concrete schedule is computed for all modified routes, taking into account driver rules. These routes are then checked for whether they violate the maximum time period between two visits at a relay station (but are otherwise feasible with respect to time windows, driver rules, and lorry capacity).

For a route/vertex sequence $(s, i_1, i_2, \ldots, i_m, i_{m+1}, \ldots, i_n, e)$, this check works as follows. The route is traversed backward from the end depot vertex $e \in E$. If, according to the route's schedule, the time period between the arrival at e and the departure from the last relay station visited (or the start depot, if no relay station is visited at all) is greater than allowed, a relay station is inserted. Assume, for example, that, in the above route, $i_m \in S \cup P \cup D \cup R$ is the first vertex encountered such that there is no relay station between i_m and e and that the time between the departure from i_m and the arrival at e exceeds the time period of 37 or 48 hours. Then a relay station vertex $r \in R$ is inserted between i_m and e. It is tried to insert r directly after i_m. If this is not possible (due to time window constraints), insertion is tried directly after i_{m+1} etc. The resulting route must be feasible with respect to time windows, driver rules *and* time period between the inserted relay station vertex r and the end depot vertex e. To verify this, it is necessary to compute a new schedule for the complete route. If the insertion was successful, the procedure is repeated, now moving backward from the last inserted relay station vertex r toward the start depot vertex s. These steps are iterated until a feasible route and a corresponding schedule are determined or the route is discarded for infeasibility.

If a request is removed from a route in the destruction step of the large neighbourhood search, all relay stations visited *after* the pickup vertex of the request are removed, too. This is motivated by the fact that it may be useful to reduce the number of relay stations, but to keep some relay stations in a partial route that was shown to be feasible. In the computational experiments, it was also tried to remove all relay stations from a route if at least one request was removed from this route. This increased the running times, but did not lead to better results.

An important issue is, of course, the selection of the relay station vertex to insert between two vertices. Here, we opted for the station causing the smallest detour. Seen globally, though, this might be suboptimal, since there might be no driver for changing at the relay station selected in this way. There might be more drivers available for continuing the lorry route at other, more distant, relay stations. Choosing such a station might help to avoid shuttle transports. Hence, storing the number of drivers available at different relay stations in different time intervals might be sensible. However, such a more sophisticated strategy is highly unlikely to be beneficial. Visiting a more distant relay station to find a suitable lorry/driver for continuing the route will obviously increase the lorry distance and the driving time compared to the visiting the closest station. This means that shuttle distance and travel time are replaced by lorry distance and travel time, but lorry kilometres are more expensive than shuttle kilometres, and lorry hours count as driving time for the driver, whereas shuttle time does not.

Stage 2: Determination of driver routes

In the second stage, a driver must be found for each lorry route segment determined in stage 1. Recall that the lorry routes consist of one or more route segments starting and ending at a depot or a relay station. Each segment must by assumption be driven by one and the same driver with one and the same lorry. To obtain feasible routes for drivers, the following points must be ensured (since they have not already been ensured in stage 1):

(a) The time between two consecutive daily rests on a driver route must be less than or equal to 13 hours.

(b) The overall duration of a driver route must be less than or equal to the maximum time between two weekly rests (144 hours).

(c) The overall working time on a driver route must be less than or equal to 48 hours.

To achieve this, an appropriate network representation is required.

Construction of the network

The network $D_2 = (V_2, A_2)$ for the VRPTWMD is set up as follows. V_2 consists of one start and one end depot vertex for each home depot of a driver, and of one customer vertex for each segment of a lorry route computed in stage 1. For a stage 2 customer vertex i, let $s_i \in S \cup R$ and $e_i \in R \cup E$ denote the start and end vertices of its corresponding stage 1 segment respectively. That is, the stage 1 segment corresponding to stage 2 vertex i starts at location l_{s_i} and ends at location l_{e_i}. All such l_{s_i} and l_{e_i} are either depots or relay stations. Again, the depot vertices have time windows starting Monday, 0:00 hours, and ending Saturday, 24:00 hours. The time window $[a_i, b_i]$ of each customer vertex i is equal to the start time window of the corresponding segment. The determination of these time windows is explained in detail below.
If no shuttle transports are allowed, the arc set A_2 is comprised of the following arcs:

(i) From each start depot vertex i, there is one arc to the corresponding end depot vertex (for unused drivers) and to each customer vertex j with $l_i = l_{s_j}$.

(ii) There is an arc between each pair (i, j) of customer vertices with $l_{e_i} = l_{s_j}$.

(iii) Finally, there is an arc from each customer vertex i to each end depot vertex j with $l_{e_i} = l_j$.

If shuttle transports are allowed, the $l_i = l_{s_j}$, $l_{e_i} = l_{s_j}$, and $l_{e_i} = l_j$ restrictions are abandoned. That is, if time windows allow, there are arcs from each start depot vertex to each customer vertex, between each pair of customer vertices, and from each customer vertex to each end depot vertex. Figure 3 depicts an example of how a network of a stage 2 instance is constructed from a stage 1 solution.
On the left hand side of Figure 3, a solution to stage 1 is described that contains two lorry routes. The first route starts and ends at depot 1 and consists of two segments. The second route starts and ends at depot 2 and involves one segment. The resulting stage 2 network is depicted on the right hand side. It is assumed that drivers are available at the same two depots where the lorries are stationed. Therefore, the stage 2 network possesses two start and two end depot vertices. Moreover, in accordance with the routes from stage 1, the network comprises three customer vertices, one for each route segment. The solid arcs represent possible movements of drivers between vertices corresponding to the same physical location, that is, where

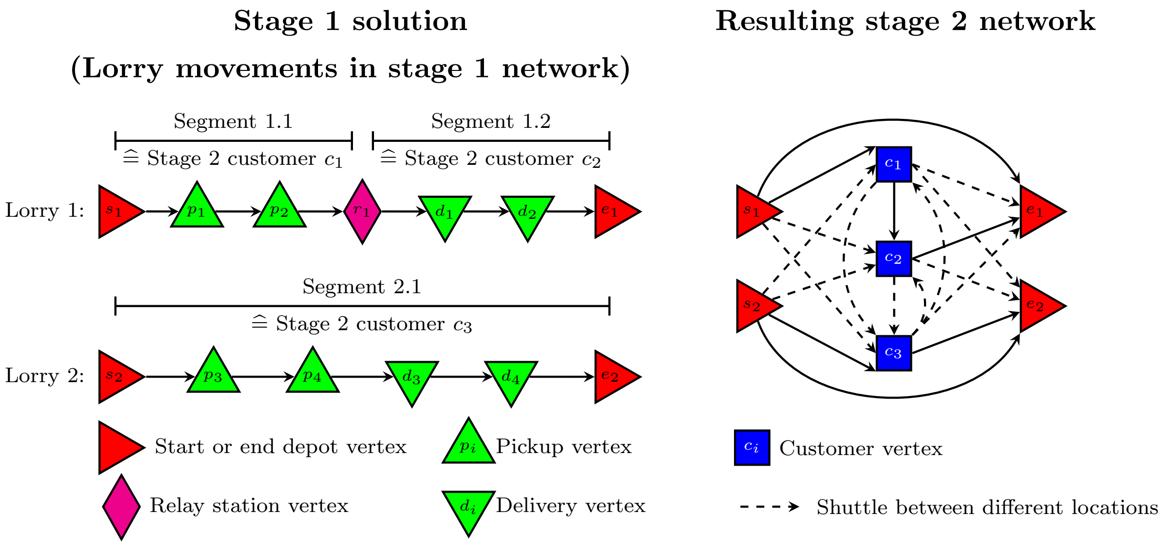


Figure 3: Example network for stage 2

no driver shuttle transports are necessary. For example, there is a solid arc from start depot vertex s_1 to customer vertex c_1, since c_1 corresponds to stage 1 route segment 1.1, which starts at vertex s_1. The dashed arcs represent driver movements between different physical locations, that is, movements requiring a shuttle transport. The depicted stage 2 network is complete in the sense that all possible arcs resulting from the stage 1 solution are shown. For example, there is no arc from vertex c_2 to c_1, because using such an arc would imply that segment 1.2, which corresponds to c_2, is performed before segment 1.1, which corresponds to c_1. This is impossible, since the requests delivered on segment 1.2 are picked up on segment 1.1. Moreover, depending on the stage 2 time windows (see below), also some of the depicted arcs may be impossible to use.

Each arc $(i, j) \in A_2$ has an associated distance $d_{ij}^{shuttle} \geqq 0$ and travel time $t_{ij}^{shuttle} \geqq 0$. In the stage 2 network, these weights correspond to the real-world distance and travel time *by shuttle van* between the respective locations. For arcs (i, j) linking vertices corresponding to the same physical location, $d_{ij}^{shuttle} := t_{ij}^{shuttle} := 0$. This means that, if no shuttle transports are allowed, both $d_{ij}^{shuttle}$ and $t_{ij}^{shuttle}$ are zero for all arcs, since in this case, there are only arcs linking vertices corresponding to the same physical location. If shuttle transports are allowed, the travel time indicates the pure driving time, disregarding any breaks or rests, because driver rules are not relevant for shuttle drivers. The weights $d_{ij}^{shuttle}$ and $t_{ij}^{shuttle}$ are necessary to determine the shuttle costs, which may be distance- and/or time-dependent. It is assumed that a driver never has to wait for a shuttle.

In order to meet the above-mentioned three requirements (a)–(c) concerning driver rules, two additional weights, $w_{ij} \geqq 0$ and $t_{ij}^{driver} \geqq 0$, are associated with each arc $(i, j) \in A_2$. The weight w_{ij} measures the working time accrued on the weekly working time account of the driver who uses arc (i, j). The weight t_{ij}^{driver} measures the wall-clock time that will elapse after arrival at vertex i or after the beginning of i's time window, whichever is greater, until a driver can reach j via the arc (i, j).

To illustrate this, the following explanations are appropriate: If a driver uses an arc (i, j) emanating from a customer vertex i and leading to a customer or end depot vertex j, this indicates that the driver drives the stage 1 segment corresponding to customer i. In other words, a 'visit' at a stage 2 customer induces a 'service time' at the customer vertex. This service time is equal to the time for executing the corresponding stage 1 segment. As mentioned above, this time is known from the unequivocal duration determined for each stage 1 route and comprises driving, service, waiting, break, and rest times along the segment. Driving and service time count as working time. Moreover, a change of vehicle incurs a relay time, which also counts as working time. (Entering the first vehicle at the beginning of a route and leaving the last vehicle at the end of a route also takes time, but this time is not counted as working time.) In addition, a driver is assumed to take a daily rest at a relay station, which increases overall route duration.

When shuttle transports are allowed, it must be considered that the time a lorry driver spends being driven in a shuttle van, although not counted as working time, must also not be counted as break or rest time. Therefore, when a driver reaches a relay station after a shuttle ride, the time since the end of his last daily rest will be positive. To ensure that such a driver can still perform the corresponding segment $i \in V_2$, we take into account the time t_i^{before} that elapses before the beginning of the first daily rest, and the time t_i^{after} that elapses after the end of the last daily rest on i. Both values are known from the computations performed in stage 1. If no daily rest is taken on a segment, we assume that a daily rest is necessary directly before and directly after the segment. Now, in order to ensure that two segments/customers i and j requiring a shuttle transport can be performed consecutively by any driver (at least as far as the 13-hour rule is concerned), we distinguish three cases:

(i) If $t_i^{after} + t_{ij}^{shuttle} \leqq 13$, the shuttle is carried out directly after performing the segment corresponding to i, and a daily rest is taken at l_{s_j}. (In this case, it is assumed, without loss of generality, that the daily rest lasts at least until the beginning of the time window at j.)

(ii) If $t_i^{after} + t_{ij}^{shuttle} > 13$ and $t_{ij}^{shuttle} + t_j^{before} \leqq 13$, a daily rest is taken at l_{e_i} directly after performing the segment corresponding to i; subsequently, the shuttle is executed and no daily rest is taken at l_{s_j}. (In this case, it is assumed that the daily rest lasts long enough so that there is no waiting time at j.)

(iii) If $t_i^{after} + t_{ij}^{shuttle} > 13$ and $t_{ij}^{shuttle} + t_j^{before} > 13$, a daily rest is taken directly before and directly after the shuttle transport. (Again, it is assumed that the daily rest at l_{s_j} lasts at least until the beginning of the time window at j.)

Similar considerations can be applied to potential arcs from a depot vertex to a customer vertex. In view of these elaborations and the described network structure, w_{ij} and t_{ij}^{driver} are set as follows (where $t_i^{service}$ denotes the overall time for execution of the segment corresponding to i, t^{daily_rest} denotes the time for the daily rest(s) between two vertices (11 or 22 hours, depending on which of the above cases (i)–(iii) holds), and t^{relay} denotes the time for handing over a lorry to another driver or being handed over a lorry from another driver):

- $w_{ij} := t_{ij}^{driver} := 0$, if i is a start depot vertex and j is a customer vertex with $l_i = l_{s_j}$.
- $w_{ij} := 0$, and $t_{ij}^{driver} := t_{ij}^{shuttle} + t^{daily\_rest}$, if i is a start depot vertex and j is a customer vertex with $l_i \neq l_{s_j}$.
- $w_{ij} := t_i^{service}$, and $t_{ij}^{driver} := t_i^{service} + t^{daily\_rest}$, if i and j are customer vertices and consecutive segments on one stage 1 route.
  In this case, no handover time is incurred, since the driver stays on the same vehicle.
- $w_{ij} := t_i^{service} + 2t^{relay}$, and $t_{ij}^{driver} := t_i^{service} + t^{daily\_rest} + 2t^{relay} + t_{ij}^{shuttle}$, if i and j are customer vertices and *not* consecutive segments on one stage 1 route.
  Here, the first relay time is incurred for handing over the lorry k to the driver who will perform the next segment on k's route. The second relay time is incurred for being handed over the lorry k' from the driver who has performed the preceding segment on k''s route.
- $w_{ij} := t_{ij}^{driver} := t_i^{service}$, if i is a customer vertex and j is an end depot vertex with $l_{e_i} = l_j$.
- $w_{ij} := t_i^{service}$, and $t_{ij}^{driver} := t_i^{service} + t^{daily\_rest} + t_{ij}^{shuttle}$, if i is a customer vertex and j is an end depot vertex with $l_{e_i} \neq l_j$.
  Note that the end depot vertex corresponds to the home depot location of the driver, not the lorry. The lorry is left at location l_{e_i}.

Using these values, only arcs (i, j) with $a_i + t_{ij}^{driver} \leqq b_j$ are introduced in order to satisfy the time window constraints.

Now, the observance of 13 hours at most between two daily rests is ensured by the fact that a driver always takes a daily rest between visiting two customers: at the relay station where the segment corresponding to the first customer ends, at the relay station where the segment corresponding to the second customer starts, or at both locations. Using an arc (i, j) increases the route duration by a wall-clock time of t_{ij}^{driver}. Depending on the actual arrival time at j, waiting time may occur. However, as explained above, it is assumed that this waiting time is added to the daily rest(s) taken between i and j.

The observance of the time since the end of the last weekly rest is ensured by the network structure. Each driver performs at most one route, he is assumed rested upon leaving his start depot vertex (that is, all driver rules resources are completely reset), and the time windows at the start and end depot vertices prohibit routes longer than 144 hours.

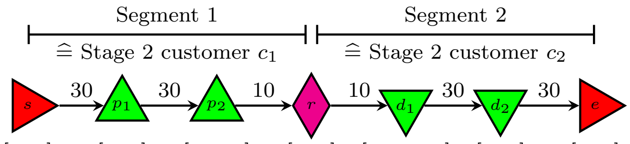
The only check concerning driver rules that has to be performed *within* the algorithm used to solve the stage 2 problem, no matter which VRPTWMD algorithm is actually used, is the check whether the accumulated working time, that is, the sum of the w_{ij} of all arcs on a driver's route, is 48 hours or less. This is essentially equivalent to checking the vehicle capacity constraint in a standard VRP (since the capacity consumption in a standard VRP, though usually associated with a customer vertex, can of course be associated with the arcs emanating from the vertex). For any insertion of a customer into a route, this check can be performed in constant time, independent of the number of customers on the route, cf. Irnich [14].

Considering time windows and dealing with the interdependence problem

Even after separating the determination of lorry and driver routes, the interdependence problem remains. Recall that in the stage 1 routine for determining a schedule considering driver rules, a schedule is computed specifying a concrete departure time from the start vertex of each segment. Now, depending on the request time windows, the driver rules, and the limited time between two visits at a relay station, it may on the one hand be possible to arrive at a relay station vertex sooner than the stage 1 schedule indicates, and it may on the other hand be possible to depart from a relay station vertex later than the stage 1 schedule denotes. In other words, there may be slack within a segment/route. Moreover, the amount of slack may differ between the segments of the same route. As an example, consider Figure 4 and assume, for simplicity, zero service and relay times and a maximum time between two visits at a relay station of 130 time units. (Driver rules are irrelevant in this example, because the overall route duration is so short that no breaks or rests are necessary or sensible; T denotes the length of the planning horizon.)

The route depicted in Figure 4 performs two pickup-and-delivery requests ($p_1 \to d_1$ and $p_2 \to d_2$) and consists of two segments. Segment 1 starts at a start depot, picks up the two requests, and ends at a relay station. Segment 2 starts at this relay station, delivers the two requests, and ends at an end depot. With respect to the time windows of the requests, we distinguish three cases. In each case, the line 'Schedule to minimize route duration' indicates the departure time at each vertex so that the overall route duration is minimized under the condition to arrive at the end depot vertex as early as possible. The line 'Overall route slack' specifies by how much the scheduled departure time from the start depot may be shifted without violating any time windows. The line 'Segment slacks' provides this information for each segment. The line 'Time windows for stage 2 customers' indicates the longest possible time windows for the stage 2 customers corresponding to the stage 1 segments, based on the minimum duration schedule and the segment slacks. The durations of the two segments, or, in other words, the service times for the stage 2 customers, are the same in all three cases: They are simply the sum of the segment driving times, that is, 70 time units (since there is no unavoidable waiting time, and since there are no stage 1 service or relay times).

In case 1, it is easy to see that the time windows for the customers in the stage 2 VRPTWMD induce no interdependencies: If customer c_1 is visited at time 30, that is, as late as possible, then customer c_2 can still be visited at time 100, that is, as early as possible. This is because it



	s	p_1	p_2	r	d_1	d_2	e
Driving time:		30	30	10	10	30	30
Case 1:							
Original time windows:	$[0, T]$	$[0, T]$	$[75, 90]$	$[0, T]$	$[110, 135]$	$[0, T]$	$[0, T]$
Schedule to minimize route duration:	30	60	90	100	110	140	170
Overall route slack:	0						
Segment slacks:		15			25		
Time windows for stage 2 customers:		$[15, 30]$			$[100, 125]$		
Case 2:							
Original time windows:	$[0, T]$	$[0, T]$	$[75, 90]$	$[0, T]$	$[100, 135]$	$[0, T]$	$[0, T]$
Schedule to minimize route duration:	20	50	80	90	100	130	160
Overall route slack:	10						
Segment slacks:		15			35		
Time windows for stage 2 customers:		$[15, 30]$			$[90, 125]$		
Case 3:							
Original time windows:	$[0, T]$	$[0, T]$	$[75, 120]$	$[0, T]$	$[100, 125]$	$[0, T]$	$[0, T]$
Schedule to minimize route duration:	20	50	80	90	100	130	160
Overall route slack:	25						
Segment slacks:		45			25		
Time windows for stage 2 customers:		$[15, 60]$			$[90, 115]$		

Figure 4: Example for stage 2 time windows

is ensured that the stage 1 segment corresponding to c_1 will be finished early enough ($30 + 70 = 100$), so that the lorry used to perform both segments will be at the relay station no later than at time 100.

In case 2, if customer c_1 is visited later than at time 20, customer c_2 is affected, and thus the stage 2 routes visiting the customers are interdependent. For example, let c_2 be on route ρ_2 and suppose that, due to subsequent customers, c_2 must be visited no later than at time 99. Further assume that the route visiting customer c_1, say, ρ_1, is changed so that c_1 can be visited no earlier than at time 30. Then, the lorry used to perform both segments will not have arrived at the relay station at time 99, and thus ρ_2 becomes infeasible. This, in turn, implies that in stage 2, route changes such as additions, removals, or repositionings of customers may have side effects on other routes, and these effects must be controlled in some way to be able to determine feasible solutions.

In case 3, the situation is still worse. Even if the time windows for the stage 2 customers are not reduced due to insertion into routes, any route visiting c_2 becomes infeasible if c_1 is visited later than at time 45.

To overcome this interdependence problem, there are several possibilities. The simplest option, of course, is to use the given concrete schedule for each route (that is, a time window of length zero for each stage 2 request) and essentially solve a pure vehicle *scheduling* problem. This greatly reduces the optimization potential for stage 2.

The other extreme is to use the slack as described above and, in stage 2, consider the route interdependencies and devise a specific algorithm for solving the VRPTWMD. This preserves the complete optimization potential (as given by the stage 1 solution). Considering the results obtained by Bock [1], it is doubtful, though, whether this second approach is promising for the large instances we try to solve.

We therefore chose a compromise. Using the flexibility given by a stage 1 solution, we set the time windows of the stage 2 requests in such a way that all requests, also those corresponding to segments belonging to the same stage 1 route, are independent of one another. More precisely, we define the slack for each segment as the maximal time by which the departure time from the start vertex of a segment can be deferred without violating any time window within the current segment *and* such that the actual departure of the subsequent segment is not delayed. In this way, no request interdependencies arise in stage 2, but still some flexibility with respect to the time windows of the requests is achieved. For the three cases in Figure 4, the procedure works as follows: In case 1, as stated above, we may set the time windows for the stage 2 requests to $[15, 30]$ and $[100, 125]$ respectively. In case 2, we may determine the time windows to $[15, 20]$ and $[90, 125]$, and, finally, in case 3, to $[15, 20]$ and $[90, 115]$.

Comments on the solution approach

It is not possible to reverse the two stages and compute driver routes in the first stage and lorry routes in the second, unless it is assumed that only empty lorries may reach relay stations. This is because when a driver route is computed in stage 1, it is unknown which locations the driver will have to visit after leaving a relay station, because it is unknown which lorry he will use and thus which requests he will have to deliver.
In tramp transportation in practice, the planning situation is often a dynamic one. Not all requests may be known in advance, and it may therefore be necessary to perform rolling horizon planning. In such a situation, only one or a few segments must be computed in advance for a lorry. The procedure in stage 1 remains the same, but the number of requests decreases. Stage 2 becomes the problem of assigning exactly one driver to each segment and at most one segment to each driver. Thus, the overall problem becomes easier.

6 Computational experiments

The present work was motivated by a research project studying the potential of an adoption of the *advanced truckload* business model to Germany and Europe. This business model originated in North America as a reaction to the deregulation of the road transport market and has provided a competitive edge for companies which successfully implemented it, the so-called advanced truckload firms (ATLFs). Details on the ATLF business model can be found in Walther [27]. One pillar of ATLFs is the increased temporal utilization of lorries obtained by allowing drivers to switch vehicles (commonly referred to as *slipseating*). Therefore, to assess the potential of slipseating for long-distance tramp transport in Germany by lorry, an extensive real-world data set was provided by our practice partner, a major German freight forwarder. The data set comprises 2,800 pickup-and-delivery requests between 1,975 locations and during a planning horizon of one calendar week, 1,645 lorries and drivers stationed at 43 depots, each of which can also function as relay station, and 157 additional relay station locations. All locations are dispersed over an area of approximately 350,000 km^2. The fleet consists of two types of lorry differing with respect to time- and distance-dependent costs, capacity, and ability to perform certain requests. Every driver is able to drive any lorry, and all drivers receive the same wage. An unlimited number of identical shuttle vans is available at any time at any location. The shuttle vans are assumed to incur distance-dependent costs only. The following table specifies some further indicators.

Indicator	**Min.**	**Avg.**	**Max.**
Length of pickup time window [hh:mm]	00:00	06:39	13:20
Length of delivery time window [hh:mm]	00:00	18:00	30:25
Time between earliest pickup and latest delivery [hh:mm]	02:15	28:10	52:00
Distance between pickup and delivery location [Lorry-kms]	1	391	1,117
Request size [Tons / Loading metres]	0.1 / 0.4	10.8 / 9.4	25.0 / 14.6
Vehicle capacity [Tons / Loading metres]	22.0 / 13.6	22.7 / 14.9	25.0 / 15.3

Table 1: Data set indicators

To test and evaluate our algorithm and to determine the best strategy for fulfilling the requests in our data set, we examined the following scenarios:

- A fixed lorry-driver assignment without the need to visit a relay station. (This was used as the baseline scenario.)

- A fixed lorry-driver assignment where visiting one of 43, 100, 200 relay stations is required for every other daily rest or once every two days (that is, every 37 or 48 hours).
- A free lorry-driver assignment where visiting one of 43, 100, 200 relay stations is required for every other daily rest or once every two days, and where the possibility of driver shuttle transports exists.

The objective was to minimize the sum of overall operating costs, which are comprised of four components:

- Fixed costs for each
  - lorry that leaves its home depot and
  - each lorry driver who leaves his home depot
- Distance-dependent costs for each
  - lorry-km and each
  - shuttle-km

For confidentiality, we cannot indicate the concrete values.

In order to perform the necessary computations, we implemented the different modules of our algorithm in Delphi/Object Pascal. The first scenario was computed with a basic variant of the stage 1 procedure, where no relay stations are considered. The other scenarios with fixed lorry-driver assignment were tackled using only the stage 1 part of our algorithm. The scenarios with free lorry-driver assignment were solved using both described stages. Tables 2 and 3 show the computational results, where 12,500 iterations of the large neighbourhood search algorithm, that is, 12,500 destruction and 12,500 reconstruction steps, were performed in each stage. In the tables, the line 'Relative overall costs' indicates the change in driver, lorry, and shuttle costs relative to the baseline scenario 1, which is why the latter has a value of 1.000 in this field.

Scenario	**1**	**2**	**3**	**4**	**5**	**6**	**7**
No. relay stations	–	43	43	100	100	200	200
Inter-visit time	–	37	48	37	48	37	48
No. lorry routes	729	767	746	768	752	756	758
No. driver routes	729	767	746	768	752	756	758
Lorry kms	1,697,000	1,785,000	1,752,000	1,778,000	1,754,000	1,752,000	1,755,000
Shuttle kms	–	–	–	–	–	–	–
Relative overall costs	1.000	1.052	1.027	1.051	1.032	1.035	1.038

Table 2: Computational results for fixed lorry-driver assignment

Scenario	**8**	**9**	**10**	**11**	**12**	**13**
No. relay stations	43	43	100	100	200	200
Inter-visit time	37	48	37	48	37	48
No. lorry routes	705	669	712	678	694	679
No. driver routes	968	962	990	970	977	961
Lorry kms	1,793,000	1,756,000	1,777,000	1,761,000	1,771,000	1,770,000
Shuttle kms	291,000	238,000	351,000	271,000	355,000	274,000
Relative overall costs	1.171	1.133	1.193	1.149	1.179	1.148

Table 3: Computational results for free lorry-driver assignment

The computational experiments showed the following:

- First and foremost, our algorithm is capable of solving large real-world instances and is able to achieve consistent and practically relevant results.
- Interestingly, for the given data set, a free lorry-driver assignment is not beneficial.
- For a fixed lorry-driver assignment, the number of routes and the overall distance travelled increase significantly when relay stations must be visited in regular intervals, compared to the baseline scenario where this is not required.
- An increased number of relay stations showed no effects. The potential benefit of reduced detours to reach one of a larger number of relay stations is compensated by the reduced

number of routes visiting the same relay station. Put differently, a network of 43 relay stations may already be considered sufficiently dense for the considered area.

- When, for free lorry-driver assignment, the time between two visits at a relay station is reduced from 48 to 37 hours, the number of routes and the overall distance travelled increase significantly.
- As previously mentioned, the stage 1 algorithm always inserts the relay station with the smallest detour between vertices of a lorry route. For free lorry-driver assignment, it might be promising to use a more sophisticated approach for choosing a relay station, where the chances are higher to find a rested driver who may readily be available to take over the lorry. The usefulness of such an approach, however, should have become visible comparing the scenarios with 200 and 43 relay stations, but the results showed no such effects.
- Increasing the temporal capacity and availability of drivers by extending their weekly working and driving time did only marginally improve the results for the scenarios with free lorry-driver assignment.
- Even if the shuttle costs are completely ignored, the costs of scenarios with free lorry-driver assignment are still higher than those of the corresponding scenarios with fixed assignment.
- With free lorry-driver assignment, no feasible solutions could be computed when no shuttle transports are allowed: In stage 2, only very few routes were found which ensured that drivers and lorries return to their home depots at the end of the planning horizon.
- When requiring that every daily rest be taken at a relay station (that is, when setting the maximum time between two visits at a relay station to 13 hours), a large part of the requests could not be fulfilled. For these requests, the time between earliest pickup and latest delivery is not sufficient to cover the distance between pickup and delivery location plus the detour to and from a relay station.

Overall, it was quite surprising that the scenarios with free lorry-driver assignment yielded no cost savings. Two reasons for this fact and the (seeming) contrast to the situation in North America are the following:

(i) The ratio of driver fixed costs to lorry fixed costs is much higher in the notoriously high-wage country Germany, so that it is hard to achieve savings by substituting drivers for lorries.

(ii) The requests do not possess a suitable structure with respect to their position in space and time. It is too seldom that a lorry is available at a relay station for a driver directly after the end of his daily rest, or, respectively, that a rested driver is available to take over a lorry directly after its arrival at a relay station. The main reason for this is that the requests in our testbed were not acquired in a systematic manner by a central department, as it is done in ATLF firms. The results obtained by Taylor et al. [25] support this conclusion: For North American long-distance road transport, the authors point out the importance of 'regularly scheduled delivery capacity in the form of delivery lanes, hubs and zones which regularize driver tours while providing performance benefits for the carrier.'

7 Summary and research outlook

This paper studied a simultaneous vehicle and crew (lorry and driver) routing and scheduling problem arising in long-distance road transport. A central aspect of the problem is that the usual assumption in the vehicle routing literature of a fixed assignment of a driver to a lorry is abandoned. Instead, lorry/driver changes are allowed at geographically dispersed relay stations, and driver shuttle transports between these relay stations are considered. This leads to interdependencies between lorries and drivers and requires the synchronization of their routes.

A heuristic solution algorithm based on large neighbourhood search has been described. The algorithm proceeds in two stages, computing routes for lorries in stage 1 and routes for drivers in stage 2, where the lorry routes from stage 1 constitute the customers/requests to perform in the

stage 2 problem. The interdependencies between the lorry and the driver routes are mitigated by specifying limited time windows for the lorry routes.
Extensive computational experiments have been performed with real-world data provided by a major freight forwarder. The results show the validity of our algorithm. Surprisingly, for typical request structures in non-timetabled long-distance transport in Central Europe, lorry/driver changes offer no savings potential. This is due to the difficulty of finding appropriate follow-up requests which are adequately situated in space and time. Under the given circumstances, a fixed lorry-driver assignment seems to be the right set-up. These results, though, are only valid for the considered business field. The developed algorithm for simultaneous vehicle and crew routing and scheduling may well lead to very different results with other data or in other application areas, so that its further study is justified.
An apparent improvement of the algorithm is the addition of a third stage, in which routes for shuttle vans are computed. Planning these transports can be done independently of stages 1 and 2, since the stage 2 solution can be used as an input for such a stage 3 procedure in the same manner as stage 1 yields the input for stage 2 in the presented algorithm.
Moreover, as discussed above, it would be highly interesting, though highly difficult, to fathom the potential offered by considering the complete temporal flexibility of the routes computed in stage 1 of the algorithm and develop a procedure capable of dealing with interdependent requests in stage 2.
An even more involved extension of our SVCRSP is to allow that a change of driver/lorry at a relay station is performed *without* the driver taking a daily rest before switching to another lorry. In this case, the interdependence of drivers and lorries becomes still closer, and the described decomposition approach no longer works, which leads to a yet more complicated situation. Bürckert et al. [2], Drexl [6], and Cheung et al. [4], in the context of long-distance road transport, short-distance collection traffic, and seaport container drayage respectively, consider such problems: Composite objects consisting of two or more types of elementary autonomous and/or non-autonomous object are required to fulfil tasks, and the elementary objects may join and separate on the fly at many different locations. In addition, whereas in our SVCRSP as well as in the references discussed in Section 3, it was always clear and fixed that two types of object must visit a location simultaneously, the latter three papers offer different options as to which combinations of object type are used to fulfil a task. A deeper study of such problems constitutes a challenging research area.

Acknowledgement. This research was funded by the Deutsche Forschungsgemeinschaft (DFG) under grant no. IR 122/5-1 and the Bundesministerium für Wirtschaft und Technologie (German Federal Ministry of Economics and Technology) under grants no. 19G7032A and no. 19G8036A.

References

[1] S. Bock.
Real-time control of freight forwarder transportation networks by integrating multimodal transport chains.
European Journal of Operational Research, 200:733–746, 2010.

[2] H. Bürckert, K. Fischer, and G. Vierke.
Holonic transport scheduling with teletruck.
Applied Artificial Intelligence, 14:697–725, 2000.

[3] A. Caprara, L. Kroon, M. Monaci, M. Peeters, and P. Toth.
Passenger railway optimization.
In C. Barnhart and G. Laporte, editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 129–187. Elsevier, Amsterdam, 2007.

[4] R. Cheung, N. Shi, W. Powell, and H. Simão.
An attribute-decision model for cross-border drayage problem.
Transportation Research Part E, 44:217–234, 2008.

[5] G. Desaulniers and M. Hickman.
Public transit.
In C. Barnhart and G. Laporte, editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 69–127. Elsevier, Amsterdam, 2007.

[6] M. Drexl.
On Some Generalized Routing Problems.
PhD thesis, Faculty of Business and Economics, RWTH Aachen University, 2007.

[7] M. Drexl.
Synchronization in vehicle routing—A survey of VRPs with multiple synchronization constraints.
Technical Report 1103, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, 2011.

[8] M. Drexl and E. Prescott-Gagnon.
Labelling algorithms for the elementary shortest path problem with resource constraints considering EU drivers' rules.
Logistics Research, 2:79–96, 2010.

[9] A. Fügenschuh.
The vehicle routing problem with coupled time windows.
Central European Journal of Operations Research, 14:157–176, 2006.

[10] A. Goel.
Vehicle scheduling and routing with drivers' working hours.
Transportation Science, 43:17–26, 2009.

[11] A. Goel.
Truck driver scheduling in the European Union.
Transportation Science, 44:429–441, 2010.

[12] B. Golden, S. Raghavan, and E. Wasil, editors.
The Vehicle Routing Problem: Latest Advances and New Challenges, volume 43 of *Operations Research/Computer Science Interfaces Series.*
Springer, Berlin, 2008.

[13] B. Hollis, M. Forbes, and B. Douglas.
Vehicle routing and crew scheduling for metropolitan mail distribution at Australia post.
European Journal of Operational Research, 173:133–150, 2006.

[14] S. Irnich.
Resource extension functions: Properties, inversion, and generalization to segments.
OR Spectrum, 30:113–148, 2008.

[15] B. Kim, J. Koo, and J. Park.
The combined manpower-vehicle routing problem for multi-staged services.
Expert Systems with Applications, 37:8424–8431, 2010.

[16] D. Klabjan.
Large-scale models in the airline industry.
In G. Desaulniers, J. Desrosiers, and M. Solomon, editors, *Column Generation*, pages 163–195. Springer, New York, 2005.

[17] A. Kok, C. Meyer, H. Kopfer, and J. Schutten.
A dynamic programming heuristic for the vehicle routing problem with time windows and European Community social legislation.
Transportation Science, 44:442–454, 2010.

[18] B. Laurent and J. Hao.
Simultaneous vehicle and driver scheduling: A case study in a limousine rental company.
Computers & Industrial Engineering, 53:542–558, 2007.

[19] J. Lenstra and A. Rinnooy Kan.
Complexity of vehicle routing and scheduling problems.
Networks, 11:221–227, 1981.

[20] E. Prescott-Gagnon, G. Desaulniers, M. Drexl, and L.-M. Rousseau.
European driver rules in vehicle routing with time windows.
Transportation Science, 44:455–473, 2010.

[21] E. Prescott-Gagnon, G. Desaulniers, and L.-M. Rousseau.
Heuristics for an oil delivery vehicle routing problem.
Technical Report G-2010-63, Les Cahiers du GERAD, 2010.

[22] S. Røpke.
Heuristic and Exact Algorithms for Vehicle Routing Problems.
PhD thesis, Department of Computer Science, University of Copenhagen, 2005.

[23] T. Sigl.
Modellbasierte Entscheidungsunterstützung am Beispiel des Verkehrsplanungsframeworks CG_{frame}.
In S. Müller, A. Roth, and N. Schmidt, editors, *Märkte, Anwendungsfelder und Technologien in der Logistik*, pages 361–378. Gabler, Wiesbaden, 2009.

[24] M. Solomon.
Algorithms for the vehicle routing and scheduling problems with time window constraints.
Operations Research, 35:254–265, 1987.

[25] G. Taylor, T. Meinert, R. Killian, and G. Whicker.
Development and analysis of alternative dispatching methods in truckload trucking.
Transportation Research Part E, 35:191–205, 1999.

[26] P. Toth and D. Vigo, editors.
The Vehicle Routing Problem.
SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2002.

[27] S. Walther.
Industrializing Transportation Networks.
Shaker, Aachen, 2010.

[28] Z. Xiang, C. Chu, and H. Chen.
A fast heuristic for solving a large-scale static dial-a-ride problem under complex constraints.
European Journal of Operational Research, 174:1117–1139, 2006.

[29] G. Zäpfel and M. Bögl.
Multi-period vehicle routing and crew scheduling with outsourcing options.
International Journal of Production Economics, 113:980–996, 2008.