CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Long-Haul Vehicle Routing and Scheduling with Working Hour Rules

Marie-Ève Rancourt
Jean-François Cordeau
Gilbert Laporte

October 2010

CIRRELT-2010-46

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université Laval · UQÀM · HEC Montréal · École Polytechnique Montréal · Université de Montréal

Long-Haul Vehicle Routing and Scheduling with Working Hour Rules

Marie-Ève Rancourt[1,2,3,*], Jean-François Cordeau[1,2], Gilbert Laporte[1,3]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Canada Research Chair in Logistics and Transportation, HEC Montréal, 3000 Côte-Sainte-Catherine, Montréal, Canada H3T 2A7

[3] Canada Research Chair in Distribution Management, HEC Montréal, 3000 Côte-Sainte-Catherine, Montréal, Canada H3T 2A7

Abstract. Long-haul carriers must comply with various safety rules which are rarely taken into account in models and algorithms for vehicle routing problems. In this paper, we consider the rules on truck driver safety during long-haul trips in North America. The problem under study has two dominant features: a routing component that consists in determining the sequence of customers visited by each vehicle, and a scheduling component that consists in planning the rest periods and the service time of each customer. We have developed different scheduling algorithms embedded within a tabu search heuristic. The overall solution methods were tested on modified Solomon instances, and the computational results confirm the benefits of using a sophisticated scheduling procedure when planning long-haul transportation.

Keywords. Vehicle routing, trip scheduling, multiple time windows, driver rules, HOS regulation, enumeration procedure, tabu search heuristic.

Acknowledgements. This work was partly supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grants 227837-09 and 39682-10. This support is gratefully acknowledged. We also thank Groupe Robert for their kind cooperation.

---

* Corresponding author: Marie-Ève.Rancourt@cirrelt.ca

Dépôt légal – Bibliothèque nationale du Québec,
Bibliothèque nationale du Canada, 2010

1 Introduction

In long-haul transportation, the distances traveled by truck drivers during a journey can be considerable and drivers often have to be on the road for several consecutive days. This is in contrast with classical vehicle routing problems (VRPs) in which all requirements are typically fulfilled within the same day. As a result, truck driver fatigue is often a contributing factor to serious accidents in long-haul transportation. To reduce fatigue and improve safety, several western countries have adopted legislations which regulate the amount of time long-haul truck drivers can drive without a rest over a given period. Trucking companies are monitored, and non-compliance with the legislation can result in substantial fines. Accordingly, legislative requirements should be considered when planning long-haul vehicle routes. However, these requirements are rarely considered in the vehicle routing literature or in routing and scheduling software.

The aim of this paper is to present, model and solve a rich vehicle routing problem arising in the less-than-truckload industry, and which takes into account the North American legislative requirements on work and rest hours. We solve the problem by means of scheduling algorithms embedded within a tabu search heuristic. Our study is motivated by the case of Groupe Robert Inc., one of the largest and best known Canadian third-party logistics providers, but our contribution is of general applicability. In addition to the North American legislation, our study also takes into account other constraints sometimes considered in VRPs, such as multiple time windows and heterogeneous fleet constraints. Multiple time windows are assigned to each customer since the problem extends over several days, but customers are visited only once.

The North American legislation stipulates for how long commercial vehicle drivers may drive without resting during long-haul trips. The legislation generally imposes three restrictions. First, a driver can only cumulate a certain number of driving hours between two consecutive rest periods. Second, a driver may not drive beyond a given number of consecutive hours after resuming duty following a rest. These two restrictions are meant to ensure that a driver does not work over a prescribed limit within a certain time interval. Third, a driver may only drive for a maximum cumulated time during a certain number of consecutive days after being off-duty for more than a prescribed time. We note that the American and Canadian regulations are similar in structure, but some of their parameters differ. For instance, it is forbidden to drive after 13 hours of cumulated driving time in Canada, while the limit is 11 hours in the United States (US).

Similar legislations are also enforced in several other countries. For example, the European Union (EU) has adopted regulation No 561/2006 (the EU regulation) which sets restrictions on driving hours

and limits the working hours of drivers. Although similar in objective, the provisions of the North American regulations are different from those of Europe. The main differences stem from the more restrictive nature of the EU regulation which requires additional breaks after specified working time periods. In fact, in addition to daily rest periods, the EU regulation stipulates that short breaks must be scheduled after certain driving time intervals. Other rules and alternative provisions also apply.

1.1 Literature review

Several solution approaches have been proposed to deal with working hour restrictions and driver break scheduling requirements within the VRP. Rochat and Semet [17] model rest breaks as fictitious customers, whereas Cordeau et al. [5] treat them as arcs in a multi-stage network. Desaulniers et al. [7] have shown that maximum working time constraints can be handled within constrained shortest path algorithms by means of resource constraints like those used for time and load variables. Campbell and Savelsbergh [3] have presented a modified insertion heuristic to handle maximum shift time limits for drivers. Savelsbergh and Sol [20] incorporate breaks and daily rests into a branch-and-price algorithm for a general pickup and delivery problem. Dealing with legislation on working hours cannot be treated as basic break scheduling because continuous movements on the road generate several dependent daily schedules over the planning horizon.

A number of authors have developed algorithms incorporating the EU or US regulations. Goel and Gruhn [11] consider the maximum driving time restriction imposed by the EU regulation within a VRP with time widows (VRPTW) and solve the problem by means of a large neighbourhood search algorithm. Goel [10] subsequently improved this methodology, although his research is focused on the basic provisions of the EU regulation and does not consider some of the alternative provisions of this regulation. The author has applied a large neighbourhood search algorithm to modified Solomon [21] test instances for the VRPTW. In their consideration of the EU regulation within a VRPTW, Kok et al. [13] present a basic break scheduling method embedded within a dynamic programming framework. Prescott-Gagnon, Drexl and Rousseau [16] have proposed a large neighbourhood search algorithm based on a column generation heuristic to solve the problem. This method relies on a tabu search for generating routes (columns) and a labeling algorithm to check their feasibility. Kok et al. [13] and Prescott-Gagnon, Drexl and Rousseau [16] have introduced in their algorithms the alternative EU provisions that were not considered by Goel [10]. The algorithm developed by Prescott-Gagnon, Drexl and Rousseau [16] outperforms those of Goel [10] and Kok et al. [13] on VRP benchmark instances. However, all these methods may not consider some routes within the optimization process because the truck driver scheduling algorithm they use is sometimes unable to find a feasible solution. Goel [9]

presents a procedure that always identifies a feasible schedule satisfying the EU regulation whenever one exists but, to our knowledge, this method has not yet been embedded within an overall algorithm for the combined VRP and truck driver scheduling problem.

Zäpfel and Bögl [23] and Bartodziej et al. [2] have worked on vehicle routing problems stemming from real case studies and incorporating rest constraints specified by the EU regulations. Zäpfel and Bögl [23] have presented a two-phase heuristic for a complex combined vehicle routing and personnel assignment problem, including outsourcing decisions. A VRP is solved during the first phase by a tabu search or by a genetic algorithm. An assignment problem is solved heuristically during the second phase. Bartodziej et al. [2] have studied a block planning problem that deals with time windows, rest regulations and a heterogeneous fleet. They have combined a column generation heuristic and a large neighbourhood search with two types of neighbourhoods.

To our knowledge, the only North Americain regulations considered in a routing context are those of the US Department of Transportation (DOT). These were first considered by Powell [15] who presented a hybrid model that takes into account forecast demands to perform dynamic routing and driver scheduling. The simulator used to test Powell's model integrated the version of the DOT regulations that were then in force. The first paper that explicitly integrates DOT regulation driver restrictions in a VRP is due to Xu et al. [22]. These authors have proposed a column generation algorithm in which DOT regulations are handled in the subproblem which is solved by means of a fast heuristic. More recently, Ceselli, Righini and Salani [4] have worked on a rich problem embodying several operational difficulties arising in real-world applications, among which they consider a working time limit before a rest. They have developed a column generation algorithm in which the pricing problem is a particular resource-constrained elementary shortest-path problem solved by a bounded bidirectional dynamic program. However, the scheme used to schedule rest periods is rather basic. A rest is inserted only when no more driving time is available, so that the possibility of resting before the allowable driving time is depleted is not considered. As shown by Archetti and Salvelsbergh [1], not using early rests means that some feasible customer sequences are considered to be infeasible.

More recently, some authors have focused their attention on the difficulty of constructing a feasible schedule for a given sequence of customers, in conformity with the US legislation. The two following studies take into account the Federal Motor Carrier Safety Administration's Hours-of-Service (HOS) regulations for commercial vehicle drivers, the latest DOT working hour rules, and concentrate on the trip scheduling problem rather than on the VRP. Archetti and Salvelsbergh [1] have considered the problem of determining how a sequence of full truckload transportation requests, each with a dispatch window at the origin, can be executed by a driver in conformity with the HOS regulations. They have

developed an algorithm, called SMARTRIP, to schedule the working and driving hours of a driver. It finds a feasible schedule in polynomial time, if one exists. Goel and Kok [12] have also studied a trip scheduling problem in which each location must be visited within one of several time windows, and have presented a scheduling method capable of finding a feasible schedule in polynomial time if one exists. They have shown that the complexity of their algorithm for the case of single time windows remains the same in the case of multiple time windows when there is a gap of at least ten hours between them.

1.2 Scientific contribution

The problem considered in this paper differs from those adressed by the above mentioned authors. First, the US regulations are different from those of the EU regulations. The EU regulations can be viewed as an extension of the North American regulations since they are more restrictive and several alternative provisions can be applied. This being said, in the problems considered by Goel [10], Kok et al. [13], and Prescott-Gagnon, Drexl and Rousseau [16], a single wide time window is associated to each customer and only one vehicle type is used. In the present paper, we consider multiple time windows for each customer and a heterogeneous fleet of vehicles. Second, we extend the ideas of the authors who have considered the HOS regulations, by embedding a trip scheduling module that can create intermediate infeasible solutions for the combined vehicle routing and driver scheduling problem. In the scheduling process, we also include the possibility for the driver to split a rest into two shorter periods spent in the sleeper berth, an alternative not yet treated in the papers that have dealt with the HOS regulations. Third, we integrate the labor costs of drivers in the routing and scheduling processes to better represent the cost of operating a vehicle in practice. In the scheduling process, we integrate optimization by minimizing the duration of the planned trips, rather than only trying to find a feasible schedule as was previously done. We also analyze the impact of considering the total schedule duration in the objective function as opposed to only concentrating on the total distance traveled.

The remainder of the paper is organized as follows. In Section 2, we provide a description of the constraints and objectives of the problem. The algorithms we have developed are described in Section 3. The computational results are presented in Section 4, followed by conclusions in Section 5.

2 Constraints and objectives

We consider a VRP with multiple time windows (VRPMTW) that combines scheduling rest periods for drivers in conformity with the North American regulations, as well as other typical VRP constraints.

We begin by describing the VRP and its constraints. We then elaborate on the work regulations in greater detail. Finally, we introduce two alternative objective functions for the problem.

2.1 Vehicle routing and scheduling constraints

We first describe the basic concepts of the VRPMTW. Given a set of vehicles based at a depot, the problem consists in determining a set of feasible routes to serve a set V of customers in order to minimize a given objective. The depot is denoted by 0 and we define $V_0 = V \cup \{0\}$. Let c_{ij} be the travel distance between i and j, where $i, j \in V_0$, and let d_{ij} be the driving time to reach j from i. The problem is solved over a planning horizon of length H. A unique time window $[a_{1,0}, b_{1,0}]$ of length H is associated with the depot, where $a_{1,0}$ and $b_{1,0}$ represent the earliest possible departure from the depot and the latest admissible arrival at the depot, respectively.

Many companies provide customized transportation services to meet their customer requirements, which may lead to the use of a heterogeneous fleet of vehicles suited to various functions (e.g. refrigerated vehicles, tank-trucks, livestock transportation vehicles, etc.). Consequently, we consider a heterogeneous vehicle fleet in this study. Every customer $i \in V_0$ has a non-negative demand q_i and can be served only by a subset of the vehicle fleet. Every vehicle type has a given load capacity which cannot be exceeded by the total demand it carries. Moreover, a restriction on on-duty time has to be imposed to prevent overloaded work schedules for drivers. A driver can be on-duty for a maximum of h^{work} hours during the planning horizon and this restriction translates directly into a duration constraint on the vehicle routes.

An ordered set of time windows $T_i = \{[a_{ti}, b_{ti}], t = 1, \ldots, \bar{t}_i\}$, with $b_{t-1,i} \leq b_{ti}$, is associated with each customer i to determine the time intervals within which delivery is allowed. Each customer $i \in V_0$ must be visited once by a vehicle during w_i time units, without service interruption. Consequently, a single time window has to be chosen for each customer delivery. If a vehicle arrives before the opening of this time window, it has to wait. It must also arrive before the closing of the selected time window, for otherwise the driver will have to wait until the opening of the next available time window, if one exists.

2.2 Working hour constraints

By law, every driver operating a commercial vehicle during long-haul trip in North America is required to record his duty status for each 24-hour period on a specific grid in a log book (Figure 1). A change of

duty status has to be registered as one of the following four possible states: driving, on-duty, off-duty, or sleeper berth. These states are described as follows.

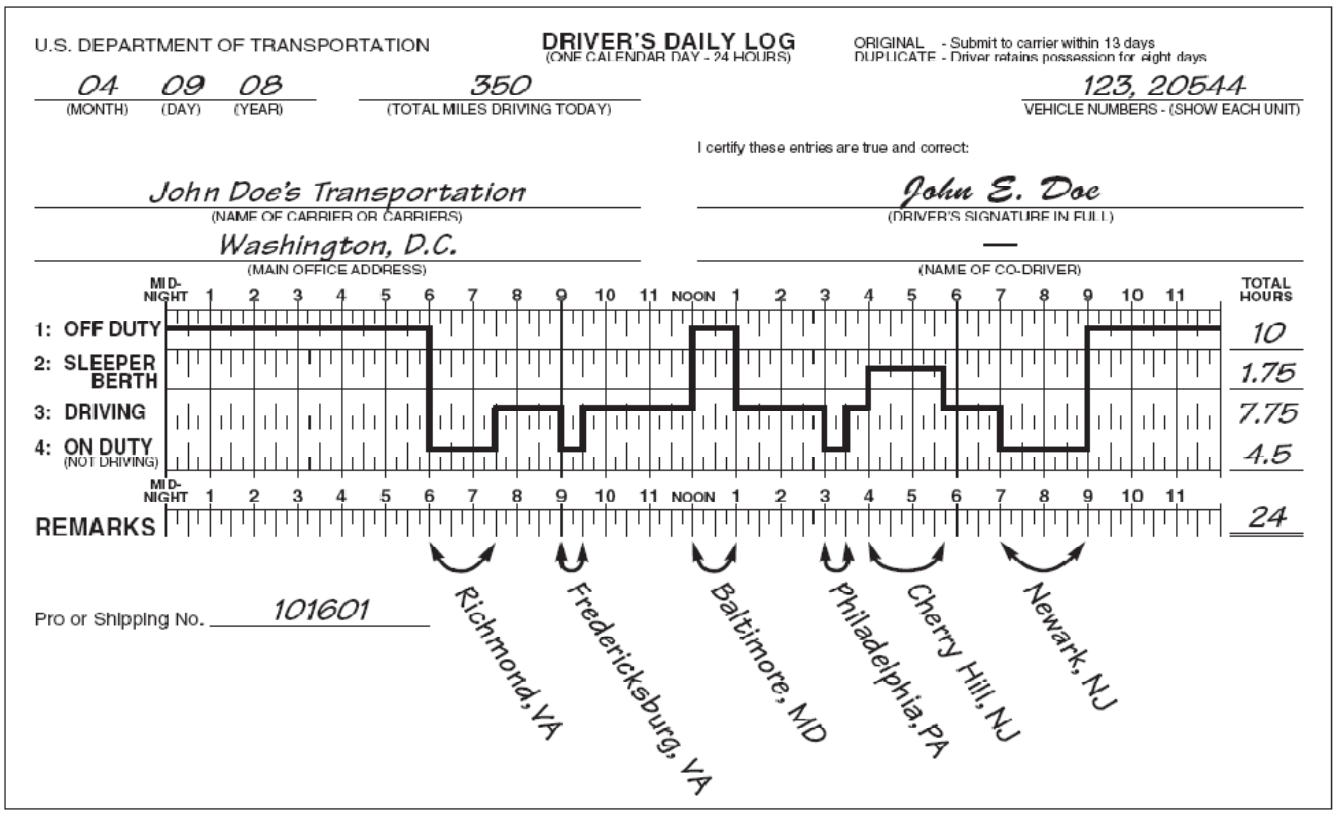

Figure 1: Illustration of a completed driver's log for one working day [8].

Driving: Driving time is the total time spent at the driving controls of a commercial vehicle in operation, even when the vehicle is stuck in a traffic jam.

On-duty: On-duty time means all the time a driver is performing work for any employer or is required to be available for work. Accordingly, this time is computed from the moment a driver begins work until he is relieved from work. On-duty time includes the following activities: 1) driving time; 2) time spent at a property of a carrier or a shipper, unless the driver has been relieved from duty; 3) time spent inspecting, servicing, or conditioning any commercial vehicle; 4) time spent in or upon any commercial vehicle, except time spent resting in the sleeper berth; 5) time spent loading, unloading, supervising or attending a commercial vehicle; 6) time spent handling work paper for shipments; 7) time spent repairing or remaining in attendance upon a disabled commercial vehicle; 8) time spent performing any compensated work for a person who is not a carrier.

Off-duty: When off-duty, drivers have no obligation to perform any work. They are free to pursue any activity and are allowed to leave the place where the vehicle is parked.

Sleeper berth: The time the driver is in the sleeper berth of a commercial vehicle in conformity with specific requirements.

In this paper, we consider the HOS regulations for commercial vehicle drivers (Part 395 of the Federal Carrier Safety Regulations) which have been in force in the US since January 2009. The HOS restrictions on driving periods are described in Table 1.

Table 1: Hour of service rules

70-hour on-duty limit	A driver cannot drive after 70 hours on-duty in eight consecutive days. He may restart an eight consecutive day period after 34 or more consecutive hours off-duty.
11-hour driving limit	A driver may only drive a maximum of 11 hours after 10 consecutive hours off-duty.
14-hour limit	A driver cannot drive beyond the 14^{th} consecutive hour after coming on-duty, following 10 consecutive hours off-duty. Off duty time does not expand the 14-hour period.

Instead of applying the 70-hour on-duty limit rule, a similar restriction with a limit of 60 working hours during a period of seven consecutive days can also be applied. In this case, we use the 70-hour on-duty limit to be consistent with the rule applied by Groupe Robert, but the idea remains the same for the 60-hour on-duty limit. A driver can also use the sleeper berth to extend the 14-hour limit. Any period of at least eight consecutive hours spent in a sleeper berth will not be included in the 14-hour horizon. This allows a driver to extend the time during which he can use the 11 hours of driving as long as the conditions described in Table 2 are respected.

In summary, a driver can drive at most h^{drive} hours and can be on-duty at most h^{on_duty} hours before a prescribed rest of at least h^{rest} hours has to be taken to gain the right to drive again. In fact, a rest period has to start at the latest when the driving limit or the on-duty limit is reached. Duty time consists primarily of driving, waiting and service time. When a driver makes use of the sleeper berth provision, he has to take a break of at least h^{long_break} consecutive hours and another one of at least h^{short_break} consecutive hours. In such a case, the driving limit and the on-duty limit remain. According to the HOS regulations, we can set the parameters as follows:

- $h^{work} = 70$ hours, the maximal cumulated on-duty hours during eight consecutive days;
- $h^{rest} = 10$ hours, the minimal duration of a rest period to regain driving time;

Table 2: Sleeper berth provision

Sleeper berth partial rest periods	Drivers using the sleeper berth provision must spend: 1) at least eight consecutive hours (but less than 10 consecutive hours) in the sleeper berth; 2) a separate block of at least two consecutive hours (but less than 10 consecutive hours) either in the sleeper berth, off-duty, or in any combination of the two.
Driving and duty limits with partial rest periods	After the second required rest period is completed, a new calculation point for the 14-hour limit, starting at the end of the previous rest period, will have to be considered to determine the available on-duty and driving hours. In this calculation, only the time block in the sleeper berth of at least eight consecutive hours will not be counted as part of the 14-hour limit; a block of less than eight hours will. The sleeper berth provision can be used continually until 10 consecutive hours off duty are taken. After 10 consecutive hours off duty, a driver has 11 hours of driving time and 14 hours of duty time available again.

- $h^{drive} = 11$ hours, the maximal cumulated driving hours between two rest periods;
- $h^{on\_duty} = 14$ hours, the maximal cumulated on-duty hours after which it is illegal to drive before resting;
- h^{long_break} = eight hours, the minimal duration of a long break period when a rest is split in conformity with the sleeper berth provision;
- h^{short_break} = two hours, the minimal duration of a short break period when a rest is split in conformity with the sleeper berth provision.

2.3 Objective functions

In studies in which driver working hours are considered (see Section 1.1), the primary objective is typically to minimize the number of vehicles in the solution, and the secondary objective is to minimize the total distance traveled. However, minimizing this objective without considering driver schedules may yield unacceptable solutions such as long routes with significant resting times. To avoid such situations we alternatively consider the total trip duration as a secondary objective. The problem is solved twice, using each of the two different objectives. The solutions obtained can then be compared and the selection of the best compromise solution is left to the decision maker.

To solve the problem of Groupe Robert, we minimize the number of vehicles used in the solution, and then the real routing cost. This cost is in fact a weighted sum of the total traveled distance and total duration.

2.4 Illustration

Figure 2 depicts a feasible solution for a VRPMTW with working rules. Each horizontal line represents either a time line for the depot (the top and the bottom lines) or for a customer. The demand of each customer is indicated on the left-hand side of their time line and the capacity of each vehicle is equal to 10 units. The square brackets and associated values on a time line represent the time windows associated with the corresponding customer. The numbers in the middle of the double arrows on the right-hand side show the driving times between locations. A vehicle trip is represented by a path that reads from the top left corner to the bottom right corner. More precisely, a path between the two depot time lines represents a driver's schedule. An inclined black segment means that the driver is driving, whereas a horizontal segment represents working time (dark dotted line) or resting time (light grey line). A dotted light grey line shows that the rest period is longer than the prescribed resting time h^{rest}. A customer is served by a driver when a dark dotted line appears in a time window. In Figure 2, a first vehicle serves customers 1, 2 and 3, and a second one serves customers 4 and 5.

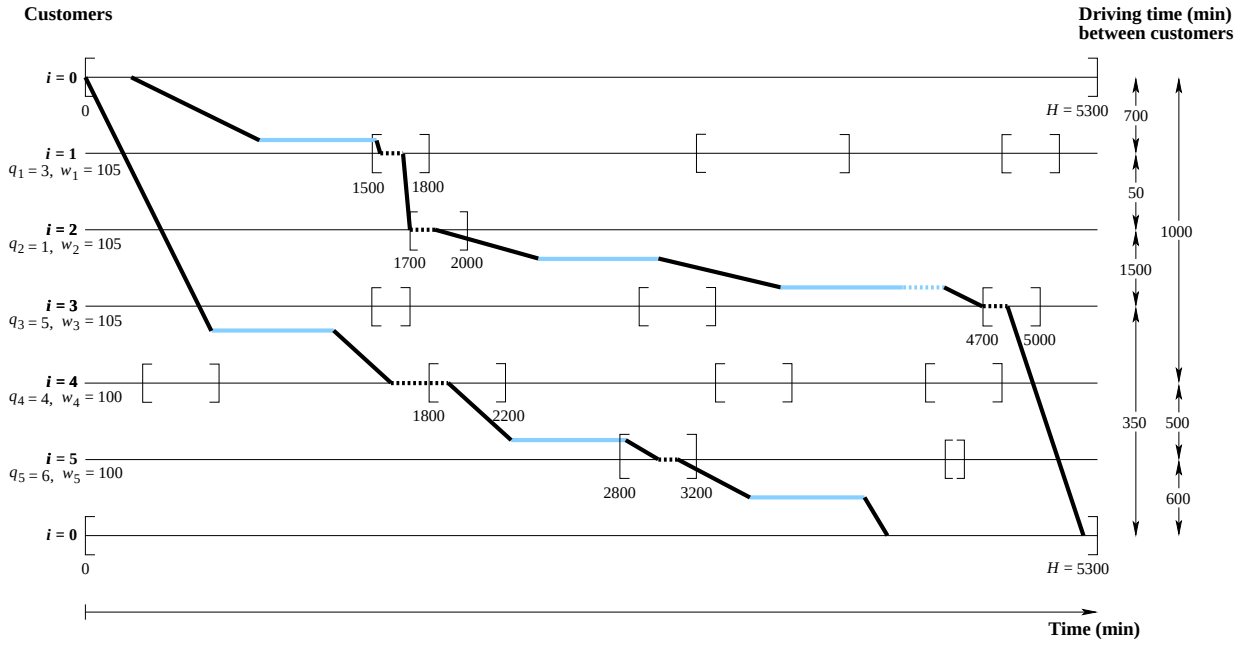

Figure 2: Illustration of a feasible solution for a VRPMTW with working hour rules. An inclined black segment represents driving time, whereas a horizontal segment represents working time (dark dotted line) or resting time (light grey line). A customer is served by a driver when a dark dotted line appears in a time window.

3 Tabu search heuristic

The problem under study contains two dominant features: a routing component consisting of determining the sequence of customers visited by each vehicle, and a scheduling component consisting of planning the rest periods and the service time of each customer. Therefore, we have developed several scheduling algorithms embedded within a routing heuristic. Our algorithm for the VRPTW with HOS regulations is based on an updated version of the Cordeau, Laporte and Mercier [6] unified tabu search algorithm (UTSA). This heuristic has been proven to be effective and flexible for a variety of VRPs. In the following, we will summarize the main features of the tabu search and detail the scheduling algorithms that can handle the HOS regulations.

3.1 Outline of the tabu search heuristic

The algorithm starts with an initial solution x_0, obtained with a simple constructive procedure, and moves at each iteration from a solution x of value $f(x)$ to another solution in the neighbourhood $N(x)$ of x. The neighbourhood $N(x)$ consists of all solutions that can be obtained by applying a given type of transformation to x. An attribute set $B(x) = \{(i,k) : i \in V, k = 1, ..., m\}$, is associated with solution x and indicates on which route k each customer i is visited. A neighbour solution is obtained by replacing a pair $(i,k) \in B(x)$ by another non-tabu pair $(i,k') \notin B(x)$. The attribute (i,k) is then declared tabu for a fixed number of iterations. However, an aspiration criterion allows the search process to accept a solution x containing a tabu attribute (i,k) if this solution is the best known solution with this attribute. The search can be broadened by accepting intermediate infeasible solutions. This is achieved through the use of a penalized objective function with self-adjusting penalties. Let $c(x)$ be the cost of the solution x, $q(x)$ the violation of capacity constraints, $d(x)$ the violation of duration constraints and $v(x)$ the violation of time window constraints. The global cost function is:

$$f(x) = c(x) + \alpha q(x) + \beta d(x) + \gamma v(x),$$

where the parameters α, β and γ are dynamically updated throughout the search. After each iteration, the values of these parameters are modified by a factor of $1 + \delta$, where $\delta > 0$: if the current solution is feasible with respect to a constraint, the value of its associated parameter is divided by $1 + \delta$, and it is multiplied by $1 + \delta$ otherwise. Diversification and intensification mechanisms are also used to improve the search. The goal of the diversification is to penalize solutions containing frequently encountered attributes, whereas intensification is a process aimed at deepening the search around good solutions.

In our problem, multiple time windows and the HOS regulations are integrated during route construction. To evaluate duration and time window violations of a route, a schedule complying with the

HOS regulations is first determined. Before calculating the cost of a solution $x' \in N(x)$, a rest scheduling algorithm generates a schedule for the given modified sequence of nodes in x' in order to assess the cost impact of the modification.

In addition, operators favouring the minimization of the number of vehicles in the solution have also been implemented within the tabu search. Every time a feasible solution with m' vehicles is found and this value is smaller than the least known number of vehicles in any feasible solutions, the upper bound on the number of vehicles available is fixed to $m' + 1$. This way, the possibility of moving customers to an empty trip remains, which ensures a certain flexibility in the search. A trip destruction operator has also been added to the tabu search procedure, but only when the secondary objective is to minimize the total distance. Every 100 iterations within the first 500 iterations, and every 500 iterations thereafter, the trip with the shortest duration is destroyed and its customers are moved sequentially into another route. This procedure is not applied when the secondary objective is to minimize the total duration because destroying the trip of shorter duration conflicts with this objective.

3.2 Adaptations of a procedure developed by Goel and Kok

Goel and Kok [12] have presented a search scheme for an algorithm that generates feasible driver schedules for an ordered sequence of λ customers. This $O(\lambda^2)$ time algorithm was not implemented but the authors suggested that such a procedure could be used in some simulation experiments and could be useful for carriers to avoid precarious delivery schedules. We have adapted this procedure to our problem and we have embedded it within our tabu search heuristic. The main differences between the suggestion made by Goel and Kok and our implementation is that we consider infeasible solutions during the search process as well as extended rest durations during the scheduling process. In the following sections, we will explain this modified procedure in more detail. The trip scheduling algorithms we have developed and embedded within the tabu search heuristic will then be described.

3.2.1 Truck driver scheduling problem

The driver scheduling problem consists in deciding when a driver will drive, work and rest in order to comply with the HOS regulations, in such a way that all customers are visited within one of their time windows. Consider a route $(i_0 = 0, i_1, ..., i_p, i_{p+1}, ..., i_\lambda = 0)$. The driving time from i_p to i_{p+1} is denoted by $d_{i_p,i_{p+1}}$, and the service duration at customer i_p is equal to w_{i_p}. We will designate by *drive* any period of continuous driving, by *rest* an off-duty period of at least h^{rest} hours, by *work* any period of

service time at a customer and by *idle* the waiting time at a customer. *Work* and *idle* time are on-duty periods not regarded as driving time.

A schedule $s = (a_0, a_1, ..., a_u, ..., a_\kappa)$ is a sequence of activities performed by the driver along the route. Let $A := \{a_u = (a_u^{type}, a_u^{length}) | a_u^{type} \in \{drive, work, rest, idle\} \text{ and } a_u^{length} \geq 0\}$ denote the set of driver activities. A partial schedule from a_i to a_j is denoted by s_{ij}.

To present the algorithm that constructs schedules in compliance with HOS regulations, we define the following values:

- starting time of schedule s:

$$l_s^{start} := \begin{cases} a_{1,0} + a_0^{length} & \text{, if } a_0^{type} = idle \text{ ;} \\ a_{1,0} & \text{, otherwise ;} \end{cases}$$

- completion time of schedule s:

$$l_s^{end} := \sum_{1 \leq u \leq \kappa} a_u^{length} \text{ ;}$$

- index of the last full rest period activity in schedule s:

$$u^r := \max \{u \mid a_u^{type} = rest \text{ or } u = 0\} \text{ ;}$$

- cumulated driving time since the last rest period in schedule s:

$$l_s^{drive} := \sum_{\substack{u^r \leq u \leq \kappa \\ a_u^{type} = drive}} a_u^{length} \text{ ;}$$

- completion time of the last rest period in schedule s:

$$l_s^{last\_rest} := \max \{l_s^{start}, l_{s_{0,u^r}}^{end}\} \text{ ;}$$

- cumulated slack time of partial schedule s_{ij}:

$$l_{s_{ij}}^{slack} := \sum_{\substack{i \leq u \leq j \\ a_u^{type} = idle}} a_u^{length} \text{ .}$$

In the classical VRPTW, in order to reduce a partial route duration and unnecessary waiting time, it may be advantageous to delay departure from the depot and the beginning of service at a customer. Similarly, in the context of long-haul transportation, it may also be beneficial to expand the length of a rest period in order to reduce waiting time and to increase driver flexibility by delaying the end of the

14-hour limit. Goel and Kok [12] apply these ideas by taking into account the time by which the last rest period may be postponed in a schedule. This time is denoted by $l_s^{postpone}$. We have adapted the computation of this value since our problem is characterized by multiple time widows, and infeasible solutions are allowed during the search.

The forward time slack is defined by Savelsbergh [19] as the largest margin by which one can postpone the beginning of service at a customer without causing any time window violations. For a given route, this value is recursively evaluated starting from the depot up to the last customer. In our problem, which deals with several rests and allows temporary time window violations, we compute the "forward time slack since the last rest", which represents the latest time at which service can begin at a customer without increasing time window violations and is recursively evaluated starting from the previous rest to the following rest. In this evaluation, for every customer i of a route, we consider the time window $[a_{t^*i}, b_{t^*i}]$ that allows the earliest possible service or generates the least delay when there is no admissible time window. Let $\sigma(u)$, with $a_u^{type} = work$, be the index of the customer served during activity u. Thus, the value t^* that determines the selected time window for a customer i is the value of t yielding $\min\{b_{ti} | l^{end}_{s_{u^r,\kappa-1}} \leq b_{ti} \text{ or } b_{ti} = b_{\bar{t}_i i}\}$, with $\sigma(u) = i$. The forward time slack since last rest in schedule s is defined by

$$F_s := \min_{\substack{u^r \leq u \leq \kappa \\ a_u^{type} = work}} \{ l^{slack}_{s_{u^r,u}} + (b_{t^*\sigma(u)} - \max\{l^{end}_{s_{0,u-1}}, a_{t^*\sigma(u)}\})^+ \} \ ,$$

where $(y)^+ = \max\{0, y\}$. Then, the time by which the end of the last rest period in the work plan s may be postponed corresponds to

$$l_s^{postpone} := \min\{F_s, l^{slack}_{s_{u^r,\kappa}}\} \ .$$

Moreover, if two *rest* periods have to be included to reach the following customer starting from the last served customer in s, then the last *rest* period cannot be postponed:

$$l_s^{postpone} = 0 \ .$$

A rest postponement is implemented whenever a new rest is inserted and when the depot is reached at the end of a route. To this end, the departure from the depot will be delayed when the newly inserted rest is the first one of the schedule, and the previous rest is lengthened otherwise. The value by which the departure from the depot or the last rest (activity a_{u^r}) will be brought forward or extended is equal to $l_s^{postpone}$. In fact, $a_{u^r}^{length}$ will be increased by a value equal to $l_s^{postpone}$ whenever a new rest is inserted.

3.2.2 Trip scheduling procedure

This section presents our extension of the method developed by Goel and Kok [12] for determining driver schedules. The main idea of this procedure is to take a partial schedule and complete it by sequentially adding activities, such as driving and rest periods, until the following customer is reached. Given a partial route in which the last customer is i_p, the remaining driving time required to reach the next customer i_{p+1} in the trip is denoted by

$$\delta_s := d_{i_p,i_{p+1}} - \sum_{\substack{u^w \leq u \leq \kappa \\ a_u^{type} = drive}} a_u^{length} ,$$

with $u^w = \max\{u | a_u^{type} = work\}$. From a partial schedule s, the trip scheduling method illustrated in Algorithm 1 can be used to determine the sequence of activities to be performed by a driver from customer i_p to customer i_{p+1}.

First, the trip scheduling method computes Δ, the maximal legal driving time until either the next customer is reached or it is necessary to plan a rest period. A driving activity of length Δ is then added to the current schedule. At this point, if the next customer is not yet reached, a full rest period is included after extending the last rest period or by delaying the departure from the depot. This process continues until the next customer i_{p+1} is reached, at which point two time windows are determined: $\left[a_{t^{\bar{r}},i_{p+1}}, b_{t^{\bar{r}},i_{p+1}}\right]$ allows the earliest direct service and $\left[a_{t^r,i_{p+1}}, b_{t^r,i_{p+1}}\right]$ allows the earliest service after a full rest period. A first potential schedule $s^{\bar{r}}$ is then obtained by directly serving the customer with a minimum idle time, and another schedule s^r is defined by resting before serving the customer. The set $S_{i_{p+1}}$ of all identified schedules that comply with HOS regulations is then augmented.

By applying this process sequentially from the depot to the succeeding nodes in a route, multiple schedules are generated to reach each customer. The best schedule is the one that completes a trip whitin the shortest time. To reduce the number of possibilities and to speed up the process, dominated schedules are pruned during the enumeration. The dominance rule works as follows. Consider two possible schedules s and s' obtained after serving the last customer of a sequence. The remaining driving time before resting in schedule s corresponds to $\Delta_s = \min\{h^{drive} - l_s^{drive},\ l_s^{last\_rest} + l_s^{postpone} + h^{on\_duty} - l_s^{end}\}$. When schedules of shorter duration are preferable, schedule s dominates schedule s' if

$$l_s^{end} - l_s^{start} \leq l_{s'}^{end} - l_{s'}^{start} \text{ and } \Delta_s \geq \Delta_{s'} .$$

These conditions mean that in schedule s the driver spends less time to perform the same duties as in schedule s' and still has more allowable driving time. Moreover, schedule s dominates schedule s' if $l_s^{end} + h^{rest} \leq l_{s'}^{end}$. This is the case since the driver in schedule s can take a complete rest, enabling a maximum driving time, while still allowing the completion of all activities earlier than in schedule s'.

Algorithm 1 Trip scheduling procedure

$\delta \leftarrow d_{i_p,i_{p+1}}$

if $i_p = 0$ **then**

$a_0 = (idle, 0)$

end if

1. Schedule driving and rest periods on the route from customer i_p to i_{p+1}:

while $\delta > 0$ **do**

$\Delta \leftarrow \min\left\{\delta, h^{drive} - l_s^{drive}, l_s^{last\_rest} + l_s^{postpone} + h^{on-duty} - l_s^{end}\right\}$

$\delta \leftarrow \delta - \Delta$

$s \leftarrow (s, (drive, \Delta))$

if $\delta > 0$ **then**

Apply foward time slack to the last rest:

$a_{u^r}^{length} \leftarrow a_{u^r}^{length} + l_s^{postpone}$

Include a rest:

$s \leftarrow (s, (rest, h^{rest}))$

end if

end while

2. Schedule service without resting:

Determine the index $t^{\bar{r}}$ of the time window allowing the earliest service at customer i_{p+1} without resting:

$t^{\bar{r}}$ the value of t yielding $\min\left\{b_{t,i_{p+1}} \mid l_{s_{0\kappa}}^{end} \leq b_{t,i_{p+1}} \text{ or } b_{t,i_{p+1}} = b_{\bar{t}_{i_{p+1}},i_{p+1}}\right\}$.

Concatenate the current schedule to create $s^{\bar{r}}$:

if $l_s^{end} \geq a_{t^{\bar{r}},i_{p+1}}$ **then**

$s^{\bar{r}} = \left(s, (work, w_{i_{p+1}})\right)$

else

$s^{\bar{r}} = \left(s, (idle, a_{t^{\bar{r}},i_{p+1}} - l_s^{end}), (work, w_{i_{p+1}})\right)$

end if

3. Schedule service with a previous rest:

Determine the index t^r of the time window allowing the earliest service at customer i_{p+1} including a previous rest:

t^r the value of t yielding $\min\left\{b_{t,i_{p+1}} \mid l_{s_{0\kappa}}^{end} + h^{rest} \leq b_{t,i_{p+1}} \text{ or } b_{t,i_{p+1}} = b_{\bar{t}_{i_{p+1}},i_{p+1}}\right\}$.

Concatenate the current schedule to create s^r:

if $l_s^{end} + h^{rest} \geq a_{t^r,i_{p+1}}$ **then**

$s^r = \left(s, (rest, h^{rest}), (work, w_{i_{p+1}})\right)$

else

$s^r = \left(s, (rest, h^{rest}), (idle, a_{t^r,i_{p+1}} - l_s^{end} - h^{rest}), (work, w_{i_{p+1}})\right\}$

end if

4. Update set of possible schedules to reach customer i_{p+1}:

$S_{i_{p+1}} \leftarrow S_{i_{p+1}} \cup \{s^r, s^{\bar{r}}\}$

To effectively implement the enumeration procedure, we have used a breadth-first search algorithm which can briefly be described as follows. Consider the route $(i_0 = 0, i_1, ..., i_p, i_{p+1}, ..., i_\lambda = 0)$ and let S_{i_p} be a set of feasible schedules obtained at node i_p. To determine the set $S_{i_{p+1}}$, the schedules in S_{i_p} that are completed the earliest are successively considered and extended within the trip scheduling algorithm. Then, prior to extending the next schedule in S_{i_p}, all dominated schedules in $S_{i_{p+1}}$ are removed. When all schedules of S_{i_p} have been generated, the same steps are repeated for $S_{i_{p+1}}$. We can determine a set of feasible schedules for the route starting at i_0 and apply the same process up to i_λ. In our case, the feasible schedule retained for the trip, $s^* \in S_{i_\lambda}$, will always be the one with $l_{s^*}^{end} \leq l_s^{end}$ for all $s \in S_{i_\lambda}$.

3.2.3 Inclusion of the sleeper berth provision in the trip scheduling procedure

We now describe how to include the possibility of splitting a rest into two periods in compliance with the sleeper berth provision. We first establish a new driver state, called *sleeper_berth*, which represents a period spent in the sleeper berth in accordance with the provision described in Table 2. When rests are split in partial rest periods, the computation of the cumulated driving time and the 14-hour limit are modified. Accordingly, some values linked to a schedule have to be defined or adjusted:

- index of the rest activity which generates the last *calculation point* of the 14-hour and driving limits in schedule s:

$$u^{cp} := \max\{u \mid a_u^{type} = \textit{sleeper\_berth} \text{ and } \exists u' > u : a_{u'}^{type} = \textit{sleeper\_berth} \text{ or } a_u^{type} = \textit{rest} \text{ or } u = 0\} ;$$

- index of the last activity of type *sleeper_berth* in schedule s:

$$u^{sb} := \begin{cases} \max\{u \mid a_u^{type} = \textit{sleeper\_berth}\} & \text{, if } \exists u \leq \kappa : a_u^{type} = \textit{sleeper\_berth} \\ -1 & \text{, otherwise ;} \end{cases}$$

- index of the last rest period in schedule s:

$$u^{rp} := \max\{u^{cp}, u^{sb}\} ;$$

- cumulated driving time since the last calculation point in schedule s:

$$l_s^{drive} := \sum_{\substack{u^{cp} \leq u \leq \kappa \\ a_u^{type} = drive}} a_u^{length} ;$$

- time past in the sleeper berth during the last rest period of schedule s:

$$l_s^{sleeper\_berth} := \begin{cases} a_{u^{sb}}^{length} & \text{, if } u^{sb} \neq -1 \\ 0 & \text{, otherwise ;} \end{cases}$$

- completion time of the rest period where the last calculation point starts in schedule s:

$$l_s^{last\_rest} := \max\{l_s^{start}, l_{s_{0u^{cp}}}^{end}\} ;$$

- minimal resting time required to regain the right to drive when the on-duty limit or the driving limit will be reached after the end of schedule s:

$$l_s^{rest\_required} := \begin{cases} h^{long\_break} & \text{, if } l_s^{sleeper\_berth} > 0 \text{ and } h^{short\_break} \leq a_{u^{sb}}^{length} < h^{long\_break} \\ h^{short\_break} & \text{, if } l_s^{sleeper\_berth} > 0 \text{ and } a_{u^{sb}}^{length} \geq h^{long\_break} \\ h^{rest} & \text{, otherwise ;} \end{cases}$$

- number of partial rest periods during the last use of the sleeper berth provision in schedule s:

$$l_s^{split\_length} := \begin{cases} \left|\left\{a_u | a_u^{type} = sleeper\_berth \text{ and } u^r \leq u \leq u^{sb}\right\}\right| & \text{, if } l_s^{sleeper\_berth} > 0 \\ 0 & \text{, otherwise ;} \end{cases}$$

- cumulated slack time of partial schedule s_{ij}:

$$l_s^{postpone} := \begin{cases} \min\{F_s, l_{s_{u^{rp},\kappa}}^{slack}, h^{rest} - l^{rest\_required}\} & \text{, if } l_s^{sleeper\_berth} > 0 \\ \min\{F_s, l_{s_{u^{rp},\kappa}}^{slack}\} & \text{, otherwise .} \end{cases}$$

Before describing our second scheduling algorithm, the trip scheduling procedure that exploits the sleeper berth provision, we present the algorithms used to generate the schedules representing the four strategies that could be adopted before serving a customer. The first possibility, described by Procedure 1, is to execute the service directly without resting. The second possibility is to take a full rest before the service, which is described by Procedure 2. Procedure 3 and Procedure 4 outline the two possibilities of partial rest periods according to the sleeper berth provision: taking a long break or a short break. In each of these procedures, the time window selected is the one that allows the earliest service. The idle time is then added to the schedule if necessary.

Algorithm 2 is similar to Algorithm 1, but more rest period possibilities are considered as it makes use of the sleeper berth provision. First, the maximum legal driving time Δ is determined. Then, a driving activity of length Δ is added to the current schedule. If the next customer is not yet reached, a rest period will be included after extending the previous rest, or by delaying the departure from the

Procedure 1 Schedule service without resting

Determine the index $t^{\bar{r}}$ of the time window allowing the earliest service at customer i_{p+1} without resting:

$t^{\bar{r}}$ the value of t yielding $\min \left\{ b_{t,i_{p+1}} \mid l^{end}_{s0_\kappa} \leq b_{t,i_{p+1}} \text{ or } b_{t,i_{p+1}} = b_{\bar{t}_{i_{p+1}},i_{p+1}} \right\}$.

Concatenate the current schedule to create $s^{\bar{r}}$:

if $l^{end}_s \geq a_{t^{\bar{r}},i_{p+1}}$ **then**

$s^{\bar{r}} = \big(s, (work, w_{i_{p+1}})\big)$

else

$s^{\bar{r}} = \Big(s, (idle, a_{t^{\bar{r}},i_{p+1}} - l^{end}_s), (work, w_{i_{p+1}})\Big)$

end if

Procedure 2 Schedule a full rest before service

Determine the index t^{r} of the time window allowing the earliest service at customer i_{p+1} with a previous rest:

t^{r} the value of t yielding $\min \left\{ b_{t,i_{p+1}} \mid l^{end}_{s0_\kappa} + h^{rest} \leq b_{t,i_{p+1}} \text{ or } b_{t,i_{p+1}} = b_{\bar{t}_{i_{p+1}},i_{p+1}} \right\}$.

Concatenate the current schedule to create s^{r}:

if $l^{end}_s + h^{rest} \geq a_{t^{r},i_{p+1}}$ **then**

$s^{r} = \big(s, (rest, h^{rest}), (work, w_{i_{p+1}})\big)$

else

$s^{r} = \big(s, (rest, h^{rest}), (idle, a_{t^{r},i_{p+1}} - l^{end}_s - h^{rest}), (work, w_{i_{p+1}})\big\}$

end if

Procedure 3 Schedule a long break before service

Determine the index t^{lb} of the time window allowing the earliest service at customer i_{p+1} with a previous long break:

t^{lb} the value of t yielding $\min \left\{ b_{t,i_{p+1}} \mid l^{end}_{s0_\kappa} + h^{long\_break} \leq b_{t,i_{p+1}} \text{ or } b_{t,i_{p+1}} = b_{\bar{t}_{i_{p+1}},i_{p+1}} \right\}$.

Concatenate the current schedule to create s^{lb}:

if $l^{end}_s + h^{long\_break} \geq a_{t^{r},i_{p+1}}$ **then**

$s^{r} = \big(s, (rest, h^{long\_break}), (work, w_{i_{p+1}})\big)$

else

$s^{r} = \big(s, (rest, h^{long\_break}), (idle, a_{t^{r},i_{p+1}} - l^{end}_s - h^{long\_break}), (work, w_{i_{p+1}})\big\}$

end if

Procedure 4 Schedule a short break before service

Determine the index t^{sb} of the time window allowing the earliest service at customer i_{p+1} with a previous short break:

t^{sb} the value of t yielding $\min \left\{ b_{t,i_{p+1}} \mid l^{end}_{s0_\kappa} + h^{short\_break} \leq b_{t,i_{p+1}} \text{ or } b_{t,i_{p+1}} = b_{\bar{t}_{i_{p+1}},i_{p+1}} \right\}$.

Concatenate the current schedule to create s^{sb}:

if $l^{end}_s + h^{short\_break} \geq a_{t^{r},i_{p+1}}$ **then**

$s^{sb} = \big(s, (rest, h^{short\_break}), (work, w_{i_{p+1}})\big)$

else

$s^{sb} = \big(s, (rest, h^{short\_break}), (idle, a_{t^{r},i_{p+1}} - l^{end}_s - h^{short\_break}), (work, w_{i_{p+1}})\big\}$

end if

depot. Depending on the state of the driver, a rest split in the sleeper berth will be continued only if a complete rest cannot be included. This process is repeated until the next customer is reached, at which point the possibility of a direct service is considered in a first schedule and all eligible rest periods are then considered in different schedules. Accordingly, if the driver is already making use of the sleeper berth provision, the schedule with the appropriate break time will be created. Another schedule that ends the rest split in the sleeper berth (including a complete rest) will also be created if it is allowed. When the driver is not making use of the sleeper berth provision, then all the rest possibilities are considered by creating the corresponding schedules. Finally, the set $S_{i_{p+1}}$ of schedules that comply with the regulations is augmented. In the tabu search heuristic, this procedure is implemented in the same way as Algorithm 1, but the dominance criterion is modified. In this case, schedule s dominates schedule s' if

$$l_s^{end} - l_s^{start} \leq l_{s'}^{end} - l_{s'}^{start} \ , \ \Delta_s \geq \Delta_{s'} \text{ and } \ l_s^{rest\_required} \leq l_{s'}^{rest\_required} \ .$$

The first two conditions mean that in schedule s the driver has spent less time to perform the same duties of schedule s' and still has more allowable driving time to continue his journey without resting. The last condition means that the resting time required to regain the right to drive, when the on-duty limit or the driving limit will be reached, is not more in schedule s than in schedule s'. Moreover, schedule s dominates schedule s' if the driver in schedule s is not making use of the sleeper berth provision or the use of the sleeper berth provision can be ended by including a complete rest ($l_s^{split\_length} = 0 \mod 2$), and $l_s^{end} + h^{rest} \leq l_{s'}^{end}$.

3.2.4 Basic trip scheduling procedure

To evaluate the benefits of a sophisticated trip scheduling procedure, we compare in Section 4.3.1 the Algorithm 1 with the basic trip scheduling procedure used by Xu et al. [22] and Ceselli, Righini and Salani [4] to solve rich VRPs by column generation. In this procedure, a rest is always inserted as late as possible, i.e. when the driving or the on-duty limit has been reached. The departure time from the depot corresponds to the earliest possible time that will not create any waiting time before serving the first customer in a route, while departing as early as possible. Algorithm 3 determines the sequence of activities to be fulfilled by a driver from customer i_p to customer i_{p+1} according to the basic trip scheduling procedure. In this procedure, the current schedule s is sequentially extended so that a unique schedule is established for any sequence of customers.

Algorithm 2 Trip scheduling procedure including sleeper berth provision

$\delta \leftarrow d_{i_p,i_{p+1}}$
if $i_p = 0$ **then**
  $a_0 = (idle, 0)$
end if
1. Schedule driving and rest periods on the route from customer i_p to i_{p+1}
while $\delta > 0$ **do**
  if $l_s^{sleeper\_berth} > 0$ and $a_{u^{sb}}^{length} \geq h^{long\_break}$ **then**
    $\Delta \leftarrow \min\left\{\delta, h^{drive} - l_s^{drive}, l_s^{last\_rest} + l_s^{postpone} + h^{on-duty} + a_{u^{sb}}^{length} - l_s^{end}\right\}$
  else
    $\Delta \leftarrow \min\left\{\delta\ ,\ h^{drive} - l_s^{drive}\ ,\ l_s^{last\_rest} + l_s^{postpone} + h^{on-duty} - l_s^{end}\right\}$
  end if
  $\delta \leftarrow \delta - \Delta$
  $s \leftarrow (s, (drive\ ,\ \Delta))$
  if $\delta > 0$ **then**
    Apply foward time slack to the last rest:
    $a_{u^{rp}}^{length} \leftarrow a_{u^{rp}}^{length} + l_s^{postpone}$
    Include a rest or a break period:
    if $l_s^{sleeper\_berth} > 0$ and $l_s^{split\_length} \neq 0 \mod 2$ **then**
      $s \leftarrow \left(s, (sleeper\_berth\ ,\ l_s^{rest\_required})\right)$
      $l_s^{rest\_required} \leftarrow h^{rest} - l_s^{rest\_required}$
    else
      $s \leftarrow \left(s, (rest\ ,\ l_s^{rest\_required})\right)$
      $l_s^{rest\_required} \leftarrow h^{rest}$
    end if
  end if
end while
2. Schedule service without resting: Execute Procedure 1
3. Schedule rest periods before service
if $l_s^{sleeper\_berth} > 0$ **then**
  if $l_s^{rest\_required} = h^{long\_break}$ **then**
    Schedule a long break before service: Execute Procedure 3
    $s^{sb} \leftarrow \emptyset$
  else
    Schedule a short break before service: Execute Procedure 4
    $s^{lb} \leftarrow \emptyset$
  end if
  if $l_s^{split\_length} \neq 0 \mod 2$ **then**
    $s^{r} \leftarrow \emptyset$
  else
    Schedule a complete rest before service: Execute Procedure 2
  end if
else
  Schedule a complete rest before service: Execute Procedure 2
  Schedule a long break before service: Execute Procedure 3
  Schedule a short break before service: Execute Procedure 4
end if
4. Update set of possible schedules to reach customer i_{p+1}
  $S_{i_{p+1}} \leftarrow S_{i_{p+1}} \cup \left\{s^{\bar{r}}, s^{r}, s^{lb}, s^{sb}\right\}$

Algorithm 3 Basic trip scheduling procedure

$\delta \leftarrow d_{i_p,i_{p+1}}$

if $i_p = 0$ **then**

$a_0 = (idle, 0)$

end if

1. Schedule driving and rest periods on the route from customer i_p to i_{p+1}:

while $\delta > 0$ **do**

$\Delta \leftarrow \min\left\{\delta, h^{drive} - l_s^{drive}, l_s^{last\_rest} + h^{on-duty} - l_s^{end}\right\}$

$\delta \leftarrow \delta - \Delta$

$s \leftarrow (s, (drive, \Delta))$

if $\delta > 0$ **then**

Include a rest:

$s \leftarrow (s, (rest, h^{rest}))$

end if

end while

2. Schedule service:

Determine the index $t^{\bar{r}}$ of the time window allowing the earliest service at customer i_{p+1}:

$t^{\bar{r}}$ the value of t yielding $\min\left\{b_{t,i_{p+1}} \mid l_{s_{0\kappa}}^{end} \leq b_{t,i_{p+1}} \text{ or } b_{t,i_{p+1}} = b_{\bar{t}_{i_{p+1}},i_{p+1}}\right\}$.

Concatenate the current schedule to create $s^{\bar{r}}$:

if $i_p = 0$ and $l_s^{end} > a_{t^{\bar{r}},i_{p+1}}$ **then**

$a_0 \leftarrow (idle, a_{t^{\bar{r}},i_{p+1}} - l_s^{end})$

end if

if $l_s^{end} \geq a_{t^{\bar{r}},i_{p+1}}$ **then**

$s \leftarrow \left(s, (work, w_{i_{p+1}})\right)$

else

$s \leftarrow \left(s, (idle, a_{t^{\bar{r}},i_{p+1}} - l_s^{end}), (work, w_{i_{p+1}})\right)$

end if

4 Computational experiments

Because our problem is new, no benchmark instances are available for it. We have first created test instances from known VRPTW benchmark problems and we have used different objective functions to compare their impact on the solution. We have also solved a real instance provided by Groupe Robert for a typical week. We first describe the test instances and we then present our computational results.

4.1 Artificial instances

We have first adapted the benchmark instances of Solomon [21] for the VRPTW. These instances contain 100 customers each and are divided into six classes that differ by the geographical distribution of the customers and their time window tightness. The customers are clustered in the C1 and C2 instances, and uniformly distributed in the R1 and R2 instances. In the RC1 and RC2 instances some customers are clustered while others are uniformly distributed. The C2, R2 and RC2 instances have wider time windows and a larger load capacity per vehicle than the C1, R1 and RC1 instances, which makes them harder to solve since the number of customers per route increases considerably. For each instance, the travel time matrix and the time windows associated with each customer were modified to better suit our context. The other parameters, i.e. the geographic coordinates of customers and the depot, the distance matrices, as well as the vehicle capacities remain the same. The time intervals during which the vehicles are available in the Solomon instances were first interpreted as periods of 24 hours. The time windows of each customer have been scaled accordingly and then replicated for eight days. The available time of all vehicles was set to 192 hours ($H = 8$ days). As proposed by Goel [10] for the EU problem, the travel time matrices have been adjusted so that the traveling speed of a vehicle is set to five distance units per hour, instead of 60 as in the original Solomon instances.

To illustrate, consider a customer i that has to be visited by a vehicle during the interval $[0, 15]$ and with $[3, 9]$ as the associated time window in the original Solomon instance. In our case, the depot time window will be $[0, 192]$ to represent the eight-day planning horizon and the set of time windows for that customer will be $T_i = \left\{ \left[3\left(\frac{24}{15-0}\right) + 24(t-1), 9\left(\frac{24}{15-0}\right) + 24(t-1) \right], t = 1, ..., 8 \right\}$. Furthermore, in order to adjust the travel time matrix of the customers visited during an eight-day horizon, each distance value is multiplied by 12.

4.2 Real-life Groupe Robert instance

We now describe the instance of Groupe Robert which concerns the distribution of goods in the US during a typical week. The geographical distribution of the 162 customers is depicted in Figure 3, which

was created using the MapPoint commercial software. A set of time windows are also associated with each customer depending on its delivery schedule and a service quality level guaranteed by the carrier. There are three types of goods with mutual incompatibilities. To deal with this constraint we consider two vehicle types, and a set of compatible vehicles is associated to each customer. The Groupe Robert instance is comparable to the RC1 instances since some customers are clustered while others are more scattered, and not many customers (between one and seven) can be served in the same route. Moreover, this is an open vehicle routing problem (Sariklis and Powell [18]) since the route does not include the trip back to the depot so as to include further on-line requests. Accordingly, all distances and traveling times from any customer to the depot are set to zero (see Pisinger and Ropke [14]).

Figure 3: Customer locations in the Groupe Robert instance.

4.3 Results and analysis

We have implemented our tabu search heuristic in C++ and have run all experiments on a Dual Core AMD Opteron 275, 2.19 GHz CPU with 8.0 GB of RAM. We have first tested the tabu search heuristic incorporating the trip scheduling algorithms on the modified Solomon intances just described and then on the Groupe Robert case. We compare the basic trip scheduling procedure with Algorithm 1, and then Algorithm 1 with Algorithm 2. We also report the most relevant results obtained on the real-life instance.

4.3.1 Comparison of the basic trip scheduling procedure with Algorithm 1 on the artificial instances

We compare the results obtained with the basic trip scheduling procedure (Xu et al. [22] and Ceselli, Righini and Salani [4]) and with Algorithm 1 embedded within the tabu search heuristic on the six sets of modified Solomon instances. Tables 3 to 6 provide a summary of the computational experiments performed when minimizing the number of vehicles as the primary objective, and the total distance or the total duration as a secondary objective. We provide the best solution values obtained after 50,000 iterations of a single run of the algorithm for each set of instances. The first colum gives the instance class, each of the next two groups of three columns compares the results obtained for each objective, and the last two colums provide the CPU time in minutes for each algorithm. The detailed results are presented in Appendix A.

Tables 3 and 4 show that Algorithm 1 clearly outperforms the basic trip scheduling procedure. When the total distance is minimized, Algorithm 1 finds solutions that require an average of 7.13% fewer vehicles and whose distance is on average 4.06% shorter. When the total duration is minimized, Algorithm 1 finds solutions that require on average 2.8% fewer vehicles and reduce the total duration by 2.74% on average. For most sets of instances, Algorithm 1 finds solutions that improve the secondary objective. When the secondary objective does not improve (C2 and R2 in table 4), fewer vehicles are used. We also note that better results are obtained when there are more customers per vehicle (instances C2, R2 and RC2). In these cases, Algorithm 1 identifies solutions with far fewer vehicles.

The basic trip scheduling procedure, which allows a rest only when there is no more available driving time, is not the best procedure for trip scheduling. However, the scheduling policies applied in this procedure are often adopted in practice and in some VRP studies. Our results clearly demonstrate the benefits of using a sophisticated scheduling procedure that includes the possibility of scheduling rest periods before the allowable driving time is depleted.

Table 3: Minimizing the number of vehicles, and then the total distance

	Number of vehicles			Total distance			CPU (min)	
	Basic	Algorithm 1	Improvement (%)	Basic	Algorithm 1	Improvement (%)	Basic	Algorithm 1
C1	10	10	0	851.63	824.32	3.17	16.62	69.16
C2	7.75	6.63	13.5	1043.90	949.08	8.83	20.14	113.6
R1	8.08	8	0.93	899.14	871.07	2.98	21.59	100.05
R2	5.91	5	14.78	833.68	831.60	0.31	32.76	185.74
RC1	9	9	0	1047.52	1012.44	3.26	18.05	79.99
RC2	6.38	5.38	14.66	940.74	863.08	7.84	27.4	151.36
Average	7.8	7.30	**7.13**	926.47	886.00	**4.06**	23.1	118.32

Table 4: Minimizing the number of vehicles, and then the total duration

	Number of vehicles			Total duration (min)			CPU (min)	
	Basic	Algorithm 1	Improvement (%)	Basic	Algorithm 1	Improvement (%)	Basic	Algorithm 1
C1	10	10	0	33188.86	31494.20	5.16	12.58	46.86
C2	6.88	6.38	6.82	40356.24	40965.04	–1.69	21.65	115.88
R1	8	8	0	34340.11	33947.95	1.31	18.10	73.38
R2	5.45	5.18	4.33	32098.76	32312.69	–0.62	31.02	135.00
RC1	9	9	0	38456.06	35835.96	6.92	16.41	65.33
RC2	6	5.63	5.65	35329.28	33398.28	5.39	27.02	120.83
Average	7.52	7.36	**2.8**	35303.57	34659.02	**2.74**	21.29	92.88

4.3.2 Comparison of Algorithm 1 and 2 on the artificial instances

We report in Tables 5 and 6 the results obtained with Algorithm 1 and Algorithm 2 on the artificial instances. When the total distance is minimized, Algorithm 2 finds solutions that require on average 2.11% fewer vehicles and 3.48% less distance. When the total duration is minimized, Algorithm 2 finds solutions that require on average 2.36% fewer vehicles and 4.02% less travel duration. These results illustrate the benefits of incorporating the sleeper berth provision in the driver scheduling process, especially when the instance involves a larger number of customers per route. However, the computing time increases considerabely since more schedule possibilities have to be considered during the solution process.

Table 5: Minimizing the number of vehicles, and then the total distance

	Number of vehicles			Total distance			CPU (min)	
	Algorithm 1	Algorithm 2	Improvement (%)	Algorithm 1	Algorithm 2	Improvement (%)	Algorithm 1	Algorithm 2
C1	10	10	0	824.32	822.14	0.26	69.16	184.46
C2	6.63	6.25	5.26	949.08	871.47	7.45	113.6	356.04
R1	8	8	0	871.07	859.38	1.22	100.05	324.38
R2	5	4.91	1.82	831.60	799.61	3.77	185.74	748.48
RC1	9	9	0	1012.44	999.84	1.24	79.99	263.4
RC2	5.38	5	6.99	863.08	792.25	8.39	151.36	554.41
Average	7.30	7.18	**2.11**	886.00	853.86	**3.48**	118.32	413.87

Table 6: Minimizing the number of vehicles, and then the total duration

	Number of vehicles			Total duration (min)			CPU (min)	
	Algorithm 1	Algorithm 2	Improvement (%)	Algorithm 1	Algorithm 2	Improvement (%)	Algorithm 1	Algorithm 2
C1	10	10	0	31494.20	30905.02	1.85	46.86	132.76
C2	6.38	6.25	1.37	40965.04	40256.45	2.37	115.88	318.44
R1	8	8	0	33947.95	32165.01	5.27	73.38	250.74
R2	5.18	5.09	1.82	32312.69	30401.78	5.65	135.00	565.15
RC1	9	9	0	35835.96	35491.8	0.62	65.33	215.58
RC2	5.63	5	10.95	33398.28	30617.73	8.36	120.83	477.14
Average	7.36	7.22	**2.36**	34659.02	33306.3	**4.02**	92.88	326.64

4.3.3 Results obtained on the Groupe Robert real-life instance

To generate a data set compatible with the format used by the algorithms proposed in this paper, we have performed some manipulations on the Groupe Robert data. We have used the MapPoint commercial software to compute the distance and travel time matrices. We have then compared the solution used by Groupe Robert to those generated by our heuristics. Having at our disposal the list of customers visited by each vehicle, we have determined a realistic route for each of them by solving a traveling salesman problem using the basic trip scheduling procedure. We consider this solution to be realistic and probably even better than the one actually used by the company.

We summarize the results of our experiments in Table 7. We have first solved this instance by means of Algorithm 1 embedded within the tabu search heuristic and compared our solution with that of Groupe Robert. We have minimized the number of vehicles as the primary objective, and the total routing cost as the secondary objective. To speed up the solution process and to ensure reasonable computational times, we have first generated an initial solution by applying the basic trip scheduling procedure for 50,000 iterations. We have then performed 100,000 iterations of the tabu search using Algorithm 1 to improve this solution. The resulting solution requires three vehicles fewer without

increasing the routing cost considerably. Indeed, the primary objective is improved by 7.5% and the routing cost goes up by approximately 3%.

We have also tested Algorithm 2 by means of the same solution process. As for Algorithm 1, the solution of Algorithm 2 requires 37 vehicles. This value has been proven to be optimal given the total demand and the vehicle capacity. Allowing rest splits in the slepper berth reduces the routing cost by 0.15%. These results are consistent with those obtained on the RC1 instances. Indeed, experiments conducted on these instances have shown that using the sleeper berth provision yields a smaller improvement in this context. Nevertheless, we can observe benefits in using a sophiticated routing and scheduling procedure as opposed to the current dispatching process.

Table 7: Summary of the results obtained on the real-life Groupe Robert instance

Algorithm	Vehicles	Routing cost	CPU (min)
Groupe Robert	40	108139.12	–
Algorithm 1	37	111417.15	70.92
Algorithm 2	37	111362.38	121.01

5 Conclusions

We have developed and compared several scheduling algorithms for a VRPMTW with working hour rules. This routing and scheduling problem is encountered by several carriers providing long-haul transportation services. The presence of multiple time windows and the working hour constraints considerably complicate the standard VRP. We have proposed a tabu search heuristic that embeds various scheduling algorithms. The algorithm was tested on the modified Solomon instances and on a real-life instance. The computational study performed on the six Solomon instance sets shows that better solutions are obtained when one considers the possibility of resting before the allowable driving time is depleted. The quality of solutions can also be improved by allowing the algorithm to split complete rests into partial rests according to the sleeper berth provision. Our tests also reveal that sophiticated scheduling procedures tend to yield better results when the vehicle routes contain more customers. Our results on the Groupe Robert data indicate that much better solutions can be obtained in terms of the number of vehicles at the expense of a modest increase in routing cost. Gains vary with respect to the complexity of the scheduling algorithm. Since short response times are often crucial in the trucking industry, dispatchers may choose to run the tabu search heuristic with the basic trip

scheduling procedure until a feasible solution is found and then use a more sophisticated scheduling algorithm to improve the solution.

Acknowledgements

This work was partly funded by the Canadian Natural Sciences and Engineering Research Council under grants 227837-09 and 39682-10. This support is gratefully acklowledged. We also thank Groupe Robert for their kind cooperation.

References

[1] C. Archetti and M. W. P. Savelsbergh. The trip scheduling problem. *Transportation Science*, 43:417–431, 2009.

[2] P. Bartodziej, U. Derigs, D. Malcherek, and U. Vogel. Models and algorithms for solving combined vehicle and crew scheduling problems with rest constraints : an application to road feeder service planning in air cargo transportation. *OR Spectrum*, 31:405–429, 2009.

[3] A. M. Campbell and M. W. P. Savelsbergh. Efficient insertion heuristics for vehicle routing and scheduling problems. *Transportation Science*, 38:369–378, 2004.

[4] A. Ceselli, R. Righini, and M. Salani. A column generation algorithm for a rich vehicle-routing problem. *Transportation Science*, 43:56–69, 2009.

[5] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M. M. Solomon, and F. Soumis. The VRP with time windows. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, pages 157–193, Philadelphia. SIAM Monographs on Discrete Mathematics and Applications, 2002.

[6] J.-F. Cordeau, G. Laporte, and A. Mercier. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society*, 52:928–936, 2001.

[7] G. Desaulniers, J. Desrosiers, A. Lasry, and M. M. Solomon. Crew pairing for a regional carrier. In N. H. M. Wilson, editor, *Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems 471*, pages 19–41. Berlin, Springer, 1999.

[8] Federal Motor Carrier Safety Administration. *Interstate Truck Driver's Guide to Hours of Service*, 2010.

[9] A. Goel. Truck driver scheduling in the European Union. To appear in *Transportation Science*.

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

[11] A. Goel and V. Gruhn. Drivers' working hours in vehicle routing and scheduling. *Proceedings of the 9th IEEE International Conference on Intelligent Transportation Systems*, Toronto, Canada, 1280-1285, 2006.

[12] A. Goel and A. L. Kok. Truck driver scheduling and U.S. hours of service regulations. Working paper, Department of Computer Science, University of Leipzig, 2010.

[13] A. L. Kok, C. M. Meyer, H. Kopfer, and J. M. J. Schutten. Dynamic programming algorithm for the vehicle routing problem with time windows and EC social legislation. To appear in *Transportation Science*.

[14] D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34:2403–2435, 2007.

[15] W. B. Powell. A stochastic formulation of the dynamic assignment problem, with an application to truckload motor carriers. *Transportation Science*, 30:195–219, 1996.

[16] E. Prescott-Gagnon, M. Drexl, and L.-M. Rousseau. European driver rules in vehicle routing with time windows. 2009. Cahier du GERAD G-2009-68, HEC Montréal, Canada.

[17] Y. Rochat and F. Semet. A tabu search approach for delivering pet food and flour in Switzerland. *Journal of the Operational Research Society*, 45:1233–1246, 1994.

[18] D. Sariklis and S. Powell. A heuristic method for the open vehicle routing problem. *Journal of the Operational Research Society*, 51:564–573, 2000.

[19] M. W. P. Savelsbergh. The vehicle routing problem with time windows: minimizing route duration. *ORSA Journal of Computing*, 4:146–154, 1992.

[20] M. W. P. Savelsbergh and M. Sol. Drive: Dynamic routing of independent vehicles. *Operations Research*, 46:474–490, 1998.

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

[22] H. Xu, Z.-L. Chen, S. Rajagopal, and S. Arunapuram. Solving a practical pickup and delivery problem. *Transportation Science*, 37:347–364, 2003.

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

Appendix A: Detailed computational results

Table 8: Minimizing the number of vehicles, and then the total distance by using the basic trip scheduling procedure

Instance	Vehicles	Distance	Working time (min)	Rest time (min)	Total duration (min)	CPU (min)
C101	10	875.74	33256.79	17400	50656.79	16.08
C102	10	840.78	35122.52	20400	55522.52	16.54
C103	10	844.84	27645.09	13200	40845.09	16.57
C104	10	824.89	29497.1	12600	42097.1	17.13
C105	10	862.18	36240.97	19200	55440.97	16.5
C106	10	853.88	34189.74	17400	51589.74	16.7
C107	10	875.87	36567.07	19800	56367.07	16.46
C108	10	846.2	36612.14	21600	58212.14	16.92
C109	10	840.27	31674.46	16800	48474.46	16.7
C201	10	1331.22	38919.71	24000	62919.71	11.59
C202	9	1121.01	36421.07	20400	56821.07	15.75
C203	7	973.22	28840.03	16200	45040.03	22.24
C204	5	743.38	19839.12	10200	30039.12	38
C205	9	1076.82	36020.27	21000	57020.27	15.74
C206	7	1158.6	28097.94	16800	44897.94	18.18
C207	8	1032.07	30956.8	18000	48956.8	18.53
C208	7	914.86	28493.7	16200	44693.7	21.11
R101	9	1127.9	35817.73	20400	56217.73	14.92
R102	8	982.72	32717.07	17400	50117.07	17.94
R103	8	871.42	30597.82	15600	46197.82	22.25
R104	8	840.61	26699.36	13800	40499.36	23.7
R105	8	951.33	30414.4	18600	49014.4	19.55
R106	8	889.62	31198.14	18000	49198.14	22.3
R107	8	853.73	24882.77	12600	37482.77	22.36
R108	8	839.58	25545.28	13200	38745.28	23.96
R109	8	878.83	28668.45	16800	45468.45	22.13
R110	8	850.07	29864.64	16200	46064.64	23.15
R111	8	867.63	29780.51	15000	44780.51	23.06
R112	8	836.27	24745.26	12600	37345.26	23.8
R201	8	957.75	32437.65	18600	51037.65	18.5
R202	7	896.17	27979.22	15600	43579.22	23.64
R203	6	815.94	24168.14	13200	37368.14	30.73
R204	5	760.54	20017.42	12000	32017.42	43.2
R205	6	947.7	23782.85	14400	38182.85	24.71
R206	6	823.56	22527.98	12600	35127.98	32.32
R207	5	784.97	19742.63	11400	31142.63	38.74
R208	5	746.17	19108.01	10800	29908.01	48.17
R209	6	838.98	24012.62	13800	37812.62	30.73
R210	6	821.19	24217.82	13800	38017.82	29.73
R211	5	777.51	19498.19	12000	31498.19	39.93
RC101	9	1134.61	32890.62	19200	52090.62	15.88
RC102	9	1075.12	32743.46	18000	50743.46	17.37
RC103	9	1015.62	32432.49	18000	50432.49	18.67
RC104	9	996.69	25409.81	13200	38609.81	19.3
RC105	9	1080.61	33443.47	19800	53243.47	17.35
RC106	9	1041.53	30565.33	18000	48565.33	17.86
RC107	9	1037.26	28789.14	16200	44989.14	18.67
RC108	9	998.75	27117.44	15600	42717.44	19.29
RC201	8	1103.13	30382.7	17400	47782.7	17.73
RC202	7	1015.4	28030.82	15000	43030.82	22.11
RC203	6	900.4	23661.13	13800	37461.13	28.73
RC204	5	813.93	19273.23	11400	30673.23	39.78
RC205	7	1024.11	28079.96	16200	44279.96	20.5
RC206	7	961.34	27050.29	15600	42650.29	23.63
RC207	6	901.47	24003.23	14400	38403.23	29.47
RC208	5	806.12	19326.53	11400	30726.53	37.25

Table 9: Minimizing the number of vehicles, and then the total duration by using the basic trip scheduling procedure

Instance	Vehicles	Distance	Working time (min)	Resting time (min)	Total duration (min)	CPU (min)
C101	10	1189.07	22524.44	12000	34524.44	12.45
C102	10	1214.47	23685.06	12000	35685.06	12.47
C103	10	1266.95	22502.98	11400	33902.98	12.44
C104	10	1073.59	19269.47	10200	29469.47	12.14
C105	10	1166.27	22035.03	12000	34035.03	13.49
C106	10	1210.64	21877.13	12000	33877.13	12.55
C107	10	1203.73	21362.67	11400	32762.67	12.83
C108	10	1273.4	21962.48	12000	33962.48	12.32
C109	10	1112.24	19680.5	10800	30480.5	12.56
C201	9	1867.23	33962.81	20400	54362.81	12.89
C202	7	1472.82	26862.07	16200	43062.07	18.08
C203	6	1298.12	22426.13	13800	36226.13	25.27
C204	5	1008.8	18347.61	10800	29147.61	34.43
C205	8	1505.36	27491.37	15600	43091.37	17.94
C206	7	1472.5	25310.5	15000	40310.5	19.97
C207	7	1427.53	25417.51	15600	41017.51	21.04
C208	6	1230.89	22431.94	13200	35631.94	23.54
R101	8	1630.42	29195.68	16800	45995.68	15.96
R102	8	1472.84	26332.62	15000	41332.62	17.8
R103	8	1295.39	22409.74	12000	34409.74	18.65
R104	8	1114.03	20200.43	9600	29800.43	17.88
R105	8	1379.36	25214.4	14400	39614.4	18.66
R106	8	1347.42	23001.85	12600	35601.85	18.68
R107	8	1255.17	21713.23	11400	33113.23	18.17
R108	8	1028.26	18558.04	9000	27558.04	17.86
R109	8	1265.71	21963.06	11400	33363.06	18.84
R110	8	1145.12	20054.46	10200	30254.46	19.02
R111	8	1219.41	21090.7	11400	32490.7	17.93
R112	8	1045.05	18947.11	9600	28547.11	17.74
R201	7	1486.05	25904.3	15000	40904.3	19.9
R202	6	1343.24	23045.42	13800	36845.42	24.44
R203	5	1163.75	20368.81	12000	32368.81	28.43
R204	5	962.66	17735.69	10200	27935.69	39.98
R205	6	1296.89	22579.36	13800	36379.36	24.88
R206	6	1160.03	20343.54	12000	32343.54	27
R207	5	1009.02	18408.46	10800	29208.46	39.14
R208	4	806.52	15723.28	9000	24723.28	39.75
R209	5	1099.19	19479.51	12000	31479.51	30.23
R210	6	1133.17	20127	11400	31527	27.64
R211	5	1039.99	18570.95	10800	29370.95	39.85
RC101	9	1599.74	28279.24	16800	45079.24	15.82
RC102	9	1445.33	24955.82	13800	38755.82	16.32
RC103	9	1504.73	25399.86	13800	39199.86	16.53
RC104	9	1211.15	21024.36	11400	32424.36	18.18
RC105	9	1601.97	26566.9	15600	42166.9	16.2
RC106	9	1463.51	25182.76	14400	39582.76	15.92
RC107	9	1462.98	24120.24	13800	37920.24	15.31
RC108	9	1238.33	21119.32	11400	32519.32	17.02
RC201	7	1560.97	27765.77	16200	43965.77	17.59
RC202	6	1380.23	23659.35	13800	37459.35	21.88
RC203	6	1233.27	22237.88	12600	34837.88	26.82
RC204	5	978.2	18001.37	10200	28201.37	39.24
RC205	7	1388	23941.29	14400	38341.29	21.41
RC206	6	1415.78	23945.66	14400	38345.66	22.41
RC207	6	1181.45	20428.59	12000	32428.59	28.94
RC208	5	1017.95	18254.35	10800	29054.35	37.86

Table 10: Minimizing the number of vehicles, and then the total distance by using Algorithm 1

Instance	Vehicles	Distance	Working time (min)	Resting time (min)	Total duration (min)	CPU (min)
C101	10	821.36	16955.56	22617	39572.56	62.16
C102	10	825.32	19596.14	24708.17	44304.3	69.39
C103	10	822.82	17799.26	23712.48	41511.74	74.05
C104	10	823.79	17515.61	19470.78	36986.39	78.88
C105	10	821.64	16990.54	22771.78	39762.33	65.13
C106	10	823.08	17016.2	26731.5	43747.7	65.64
C107	10	833.26	19018.31	25165.38	44183.69	66.51
C108	10	827.63	16600.34	36980.34	53580.68	69.6
C109	10	819.97	16265.58	29211.58	45477.16	71.1
C201	9	1223.29	22468.26	39634.56	62102.82	60.41
C202	7	1118.2	20144.13	31980.2	52124.34	93.51
C203	6	837.59	16605.62	31355.32	47960.94	133.68
C204	5	709.69	14811.57	21030.01	35841.58	206.58
C205	7	1002.27	18499.76	30291.47	48791.24	82.61
C206	7	892.12	17060.88	31167.27	48228.15	99.78
C207	6	938.11	17703.28	23528.24	41231.52	116.89
C208	6	871.37	16537.65	26728.05	43265.7	115.31
R101	8	1061.08	22610.31	37180.08	59790.39	77.61
R102	8	928.65	20653.48	33556.98	54210.45	95.56
R103	8	850.41	20989.1	30575.22	51564.33	103.6
R104	8	837.87	18121.1	17246.43	35367.54	109.45
R105	8	900.98	18683.81	35905.51	54589.31	90.37
R106	8	845.81	18919.43	28827.79	47747.22	99.83
R107	8	842.82	20880.42	24212.4	45092.82	106.75
R108	8	832.02	17852.79	18935.35	36788.13	110.46
R109	8	842.97	16663.15	30319.11	46982.26	96.83
R110	8	840	18346.19	27287.28	45633.47	103.01
R111	8	836.62	16513.61	27065.33	43578.95	103.3
R112	8	833.62	16003.43	18518.33	34521.76	103.78
R201	6	954.91	19810.96	29333.97	49144.93	111.58
R202	6	859.56	18026.28	25431.11	43457.39	134.78
R203	5	828.09	17645.66	21073.37	38719.02	177.39
R204	5	751.61	15986.53	14721.21	30707.74	229.79
R205	5	1004.68	18253.37	25148.93	43402.3	148.83
R206	5	800.76	16953.91	20791.56	37745.47	179.85
R207	4	790.27	15725.89	13445.33	29171.21	219.53
R208	4	749.85	15178.93	13001.43	28180.36	272.41
R209	5	830.07	16700.3	23593.88	40294.17	176.09
R210	5	821.96	16532.49	21454.38	37986.86	175.71
R211	5	755.88	15513.11	18910.22	34423.33	217.2
RC101	9	1063.84	22291.48	35096.12	57387.59	71.5
RC102	9	1004.74	20073.94	34267.98	54341.92	79.22
RC103	9	1007.52	20246.47	28715.56	48962.04	84.36
RC104	9	997.99	18427.88	18170.67	36598.56	87.34
RC105	9	1039.17	20271.02	33709.41	53980.42	77.7
RC106	9	997.72	19138.54	34666.88	53805.42	77.86
RC107	9	991.86	20115.81	27389.47	47505.28	80.73
RC108	9	996.65	17959.86	17645.57	35605.43	81.23
RC201	7	972.9	21294.78	30523.82	51818.6	96.07
RC202	6	950.4	20307.72	25690.2	45997.92	120.58
RC203	5	926.36	17832.44	21977.81	39810.26	162.47
RC204	5	788.48	16468.78	16357.39	32826.16	219.31
RC205	4	795.39	14487.98	18599.56	33087.55	107.45
RC206	6	856.33	18581.87	25994.07	44575.94	136.74
RC207	5	865.94	16606.51	23129.56	39736.07	158.76
RC208	5	748.87	15047.59	20392.75	35440.33	209.45

Table 11: Minimizing the number of vehicles, and then the total duration by using Algorithm 1

Instance	Vehicles	Distance	Working time (min)	Resting time (min)	Total duration (min)	CPU (min)
C101	10	1183.49	21194.23	11820.05	33014.28	43.6
C102	10	1159.36	21118.27	12591.8	33710.07	45.56
C103	10	1167.16	20601.95	11320.61	31922.56	50.15
C104	10	1037.29	18554.45	8755.06	27309.51	60.55
C105	10	1170.72	20746.95	12228.16	32975.11	42.15
C106	10	1231.29	21016.49	11371.24	32387.73	45.74
C107	10	1111.05	20180.32	11158.2	31338.52	41.92
C108	10	1182.13	20562.62	11848.59	32411.21	42.95
C109	10	1086.23	19036.46	9342.39	28378.85	49.1
C201	8	1621.28	26290.27	25953.9	52244.17	90.21
C202	7	1448.48	24726.81	20160.12	44886.93	114.81
C203	5	1140.79	20003.59	18061.64	38065.22	231.19
C204	5	1006.02	18316.17	10347.23	28663.41	173.9
C205	7	1332.89	22343	22681.17	45024.16	61.36
C206	7	1472.42	23683.34	17627.62	41310.96	80.11
C207	6	1356.11	22333.06	17704.71	40037.77	89.71
C208	6	1267.28	21291.38	16196.36	37487.73	85.78
R101	8	1523.01	26606.99	17875.99	44482.98	63.81
R102	8	1430.43	24986.17	17835.54	42821.7	74.54
R103	8	1289.96	22134.38	13532.47	35666.86	75.22
R104	8	1082.76	19097.21	9783.29	28880.5	81.08
R105	8	1441.56	23918.68	16032.06	39950.74	66.4
R106	8	1263.75	21663.54	14107.23	35770.77	70.44
R107	8	1171.33	20177.61	10626.9	30804.51	74.69
R108	8	1019.36	18232.33	8515.65	26747.97	83.09
R109	8	1301.29	21684.8	12099.23	33784.04	69.69
R110	8	1164.45	19982.9	11092.18	31075.08	72.06
R111	8	1174.33	20138.57	10564.55	30703.12	73.31
R112	8	1021.85	18262.17	8424.99	26687.17	76.23
R201	6	1362.98	23460.13	18680.52	42140.66	81.92
R202	6	1304.94	22574.09	15805.76	38379.84	90.38
R203	5	1165.21	20095.47	14275.89	34371.36	130.36
R204	5	948.12	17863.37	10351.87	28215.24	183.41
R205	6	1219.41	21584.22	13871.68	35455.9	111.66
R206	5	1099.33	19191.91	11807	30998.91	124.9
R207	5	1014.34	18330.18	10709.33	29039.52	170.02
R208	4	851.65	16219.8	9627.78	25847.58	163.95
R209	5	1121.53	19514.42	11528.9	31043.31	128.32
R210	5	1079.16	19053.05	13362.45	32415.51	128.62
R211	5	993.21	17918.58	9613.17	27531.75	171.44
RC101	9	1474.68	24250.94	18003.6	42254.54	55.6
RC102	9	1420.46	23298.17	15302.28	38600.45	64.19
RC103	9	1367.03	22501.26	11804.52	34305.79	66.96
RC104	9	1180.26	20183.85	10238.25	30422.1	74.28
RC105	9	1577.86	25436.58	17314.42	42751	61.56
RC106	9	1352.85	22321.02	13299.51	35620.52	63.82
RC107	9	1276.9	21322.8	11426	32748.8	65.38
RC108	9	1152.37	19828.51	10155.94	29984.45	70.83
RC201	7	1421.73	24459.42	21639.4	46098.82	63.07
RC202	6	1286.16	21916.1	16752.9	38669	89.92
RC203	6	1155.73	20467.95	12457.29	32925.24	125.08
RC204	5	941.63	17321.07	9789.39	27110.46	191.8
RC205	5	929.2	15756.48	10788.9	26545.38	92.33
RC206	6	1297.12	21670.72	13207.19	34877.91	116.7
RC207	5	1198.02	20376.28	12728.11	33104.39	122.56
RC208	5	959.37	17512.39	10342.67	27855.06	165.18

Table 12: Minimizing the number of vehicles, and then the total distance by using Algorithm 2

Instance	Vehicles	Distance	Working time (min)	Resting time (min)	Total duration (min)	CPU (min)
C101	10	820.62	18051.73	18919.93	36971.65	131.19
C102	10	824.78	18075.47	20119.21	38194.68	168.95
C103	10	821.37	21890.5	23959.67	45850.16	224.85
C104	10	821.65	17569.45	19343.4	36912.85	326.07
C105	10	824.7	19299.38	22350.97	41650.35	143.19
C106	10	821.15	17880.72	23441.62	41322.33	148.09
C107	10	824.54	18911.22	23103.22	42014.44	150.87
C108	10	820.92	21887.07	23936.15	45823.22	167.3
C109	10	819.56	19783.91	26351.88	46135.79	199.64
C201	8	1065.66	22404.02	40558.13	62962.15	145.97
C202	7	984.21	20075.77	33901.68	53977.45	238.68
C203	6	861.46	18541.25	27450.81	45992.06	379.9
C204	5	713.29	17007.27	17671.44	34678.71	850.44
C205	7	905.63	21452.67	29884.75	51337.41	189.23
C206	6	796.91	16585.76	31102.78	47688.55	238.9
C207	6	793.71	17966.46	30009.65	47976.12	295.02
C208	5	850.92	17082.07	23172.97	40255.04	510.16
R101	8	1002.47	22649.25	35863.91	58513.16	168.81
R102	8	892.58	20656.27	34929.48	55585.75	233.95
R103	8	837.32	20729.24	25769.9	46499.15	309.28
R104	8	838.88	18225.39	17357.55	35582.94	432.78
R105	8	876.06	22816.06	34249.71	57065.78	212.6
R106	8	845.66	21320.33	31998.15	53318.49	267.62
R107	8	838.69	19789.93	25870.59	45660.52	355.1
R108	8	834.31	17679.68	16341.12	34020.8	468.3
R109	8	841.25	20793.92	29499.23	50293.15	263.02
R110	8	837.88	19335.32	19744.94	39080.27	345.01
R111	8	834.55	20497.8	23006.16	43503.97	341.24
R112	8	832.88	16780.46	13702.88	30483.34	494.9
R201	6	943.27	18944.1	25109.12	44053.22	263.9
R202	6	838.27	20231.4	25687.44	45918.84	365.77
R203	5	801.12	17368.64	19164.38	36533.02	798.42
R204	4	784.86	15796.3	10440	26236.3	1667.61
R205	5	936.02	18295.85	19886.18	38182.03	601.55
R206	5	810.23	16878.87	18984.53	35863.4	528.1
R207	4	594.54	13638.4	14217.82	27856.22	832.57
R208	4	762.23	15471.21	12198.52	27669.73	1246.19
R209	5	790.65	16829.81	18117.93	34947.75	543.34
R210	5	796.03	16957.36	17853.24	34810.6	538.25
R211	5	738.53	15811.01	15852.79	31663.8	847.58
RC101	9	1060.46	24351.44	40468.96	64820.4	175.9
RC102	9	1002.28	22986.83	33810.55	56797.39	220.8
RC103	9	990.47	21365.24	25223.52	46588.75	288.04
RC104	9	977.72	19797.51	19257.6	39055.11	370.73
RC105	9	1001.33	23256.37	28352.25	51608.62	206.24
RC106	9	985.91	22393.29	32434.6	54827.89	213.48
RC107	9	994.89	21145.95	20950.88	42096.83	271.55
RC108	9	985.63	18937.79	17540.44	36478.23	360.46
RC201	6	927.33	20331.23	27303.94	47635.18	251.8
RC202	6	874.89	19389.74	20205.72	39595.45	348.17
RC203	5	797.37	17464.29	18360.74	35825.03	549.68
RC204	5	759.38	15852.88	14555.2	30408.08	1044.27
RC205	3	406.95	8411.26	10721.95	19133.21	245.96
RC206	5	974.95	18746.77	17390.5	36137.27	611.99
RC207	5	840.51	17281.48	14420.73	31702.21	506.44
RC208	5	756.65	16113.42	16788.51	32901.93	876.94

Table 13: Minimizing the number of vehicles, and then the total duration by using Algorithm 2

Instance	Vehicles	Distance	Working time (min)	Resting time (min)	Total duration (min)	CPU (min)
C101	10	1128.89	19977.91	11992	31969.91	94.77
C102	10	1177.24	20620.26	11790.98	32411.25	118.75
C103	10	1107.56	19374.62	11667.89	31042.51	158.84
C104	10	992.05	17973.24	8444.98	26418.21	251.26
C105	10	1072.59	19133.83	12171.69	31305.53	100.18
C106	10	1133.65	20058.49	14052.5	34110.99	100.26
C107	10	1149.16	20129.45	11664.02	31793.47	106.95
C108	10	1121.22	19470.87	11356.82	30827.69	117.33
C109	10	1051.89	18749.02	9516.58	28265.6	146.54
C201	8	1779.24	28518.41	29314.37	57832.78	115.05
C202	7	1419.91	24085.560	20956.43	45041.99	214.89
C203	6	1267.97	21406.9	13572.59	34979.49	360
C204	5	956.19	17497.57	10452.87	27950.44	777.19
C205	7	1368.53	23269.02	21607.8	44876.82	160
C206	6	1256.14	21730.58	17676.08	39406.66	186.35
C207	6	1233.27	20998.2	17050.21	38048.41	236.77
C208	5	997.1	18407.47	15507.53	33915.01	497.32
R101	8	1546.42	26289.86	19148.78	45438.64	152.22
R102	8	1455.31	24319.59	16434.3	40753.89	199.93
R103	8	1286.29	21695.09	12466.17	34161.26	238.28
R104	8	1071.21	18985.6	9233.36	28218.96	356.36
R105	8	1271.65	21599.81	13805.76	35405.57	164.34
R106	8	1190.13	20389.58	11742.26	32131.84	205.5
R107	8	1146.57	19946.77	10544.87	30491.64	272.78
R108	8	1010.01	18120.14	7996.44	26116.58	352.95
R109	8	1186.19	20234.23	10278.36	30512.59	203.32
R110	8	1086.83	19041.91	9476.49	28518.39	263.7
R111	8	1107.03	19313.35	10182.35	29495.7	254.98
R112	8	957.96	17495.54	7239.54	24735.08	344.47
R201	6	1245.53	21445.55	18557.36	40002.92	354.09
R202	6	1235.8	21329.53	14857.37	36186.9	278.41
R203	4	896.11	15740.83	10968.58	26709.41	457.96
R204	5	996.84	17979.96	9699.85	27679.81	1020.92
R205	6	1188.55	20393.4	13200.22	33593.62	332.74
R206	5	1030.97	18371.65	11349.09	29720.73	449.44
R207	5	924.28	17099.2	9886.1	26985.3	722.58
R208	4	824.97	15899.64	9039.1	24938.73	852.86
R209	5	1054.36	18697.84	11121.79	29819.63	379.65
R210	5	1086.86	19078.23	11667.12	30745.35	679.44
R211	5	981.73	17780.72	10256.51	28037.23	688.56
RC101	9	1585.69	25502.36	16708.41	42210.77	148.11
RC102	9	1341.98	22336.45	13613.48	35949.94	177.37
RC103	9	1313.81	21765.76	11968.91	33734.66	225.17
RC104	9	1229.64	20761.73	10377.69	31139.41	298.35
RC105	9	1556.39	25074.83	15270.14	40344.97	170.83
RC106	9	1350.22	22604.52	13680.19	36284.71	181.53
RC107	9	1227.54	20910.26	11907.16	32817.41	228.03
RC108	9	1261.19	21134.29	10318.26	31452.55	295.22
RC201	6	1287.33	22237.33	16587.83	38825.16	208.19
RC202	6	1235.86	21420.09	15399.76	36819.86	284.17
RC203	5	1140.18	19860.45	14561.76	34422.2	465.83
RC204	5	952.77	17555.24	10136.59	27691.82	937.77
RC205	3	566.3	9898.3	6979.46	16877.76	212.26
RC206	5	1096.69	19316.3	13503.36	32819.66	583.04
RC207	5	1044.59	18535.11	11406.82	29941.93	385.62
RC208	5	989.27	17871.27	9672.19	27543.46	740.26