TRANSPORTATION SCIENCE
Vol. 44, No. 4, November 2010, pp. 429–441
ISSN 0041-1655 | EISSN 1526-5447 | 10 | 4404 | 0429

informs

DOI 10.1287/trsc.1100.0330
© 2010 INFORMS

Truck Driver Scheduling in the European Union

Asvin Goel
MIT-Zaragoza International Logistics Program, Zaragoza Logistics Center, 50197 Zaragoza, Spain, and
Applied Telematics/e-Business Group, Department of Computer Science, University of Leipzig,
04109 Leipzig, Germany, asvin@mit.edu, asvin.goel@uni-leipzig.de

Since April 2007 working hours of truck drivers in the European Union are controlled by regulation (EC) No. 561/2006. According to the new regulation, road transport undertakings must organise the work of drivers in a way that drivers are able to comply with the regulations and can be made liable for infringements committed by the drivers. Although of particular importance in long-distance haulage, regulations on working hours of truck drivers have received very little attention in the scheduling literature. This paper presents a method for scheduling driving and working hours of truck drivers with respect to regulation (EC) No. 561/2006. Given a sequence of locations to be visited within specified time windows, the approach is guaranteed to find a schedule complying with the regulation if such a schedule exists.

Key words: vehicle scheduling; regulation (EC) No. 561/2006; drivers' working hours
History: Received: April 2009; revisions received: February 2010, March 2010; accepted: March 2010. Published online in *Articles in Advance* August 19, 2010.

1. Introduction

Driver fatigue is a significant factor in approximately 20% of commercial road transport crashes and one out of two long haul drivers has fallen asleep while driving (see European Transport Safety Council 2001). Under previous legislation in the European Union it had been legally possible for truck drivers to drive almost nine hours with a break period of only 15 minutes. To improve road safety and working conditions of drivers, the European Parliament and the Council of the European Union have adopted regulation (EC) No. 561/2006 laying down provisions concerning driving and working hours of drivers in road transport. This regulation went into force in April 2007 and applies to drivers of vehicles with a total mass of at least 3.5 tons and vehicles constructed to carry more than nine persons. According to the new regulation, road transport undertakings can be made liable for infringements committed by drivers. Neglecting regulatory constraints when scheduling driving and working hours of drivers may lead to infringements and/or delayed arrival times because of required breaks and rest periods that have not been scheduled. Consequently, road transport undertakings must ensure that truck driver schedules comply with regulation (EC) No. 561/2006.

Scheduling problems for mobile staff in the transportation sector have been studied since the 1960s. An early survey of airline crew scheduling is presented by Arabeyre et al. (1969). Kohl and Karisch (2004) describe typical rules and regulations arising in airline crew rostering. Traditionally, airline crew scheduling is performed after a flight timetable has been constructed and aircrafts have been assigned to each flight. More recently, several approaches integrating aircraft routing and crew scheduling have been proposed, e.g., by Cordeau et al. (2001), Mercier, Cordeau, and Soumis (2005), and Sandhu and Klabjan (2007). Further approaches regarding simultaneous vehicle and driver scheduling are presented for mass transit systems by Haase, Desaulniers, and Desrosiers (2001), Valouxis and Housos (2002), Freling, Huisman, and Wagelmans (2003), Huisman, Freling, and Wagelmans (2005) and for limousine rental by Laurent and Hao (2007). Ernst et al. (2004) provide a comprehensive annotated bibliography on personnel scheduling that covers crew and driver scheduling problems for airlines, railways, and mass transit systems.

Interestingly, scheduling driving and working hours of truck drivers has attracted very little interest in the literature. Driver scheduling in road freight transportation differs from airline crew scheduling and driver scheduling in rail transport or mass transit systems because it is usually possible to interrupt transportation services to take breaks and rest periods. Furthermore, time constraints in road freight transport are usually not as strict, and departure and arrival times can often be scheduled with some degree of freedom. Savelsbergh and Sol (1998) consider a dynamic and generalised pickup and delivery problem in which lunch breaks and night breaks must be taken within fixed time intervals. Xu et al. (2003) study a rich pickup and delivery problem and

present a column generation approach for combined routing and scheduling. They conjecture that determining a minimal cost truck driver schedule for a given sequence of customer locations is NP-hard in the presence of multiple time windows, restrictions on drivers' working hours imposed by the U.S. Department of Transportation, as well as different costs for time spent waiting idle and time spent taking rest periods. Archetti and Savelsbergh (2009) present an algorithm for truck driver scheduling problems in the United States. They prove that their algorithm finds a feasible truck driver schedule in cubic time, if one exists. The restrictions imposed by the U.S. Department of Transportation differ significantly from regulation (EC) No. 561/2006, which requires that, in addition to daily rest periods in which drivers can sleep, shorter break periods for recuperation must be scheduled after a certain amount of driving.

The only approaches known to the author considering regulation (EC) No. 561/2006 in scheduling driving and working hours of truck drivers are presented by Goel (2009) and Kok et al. (2009). Both papers study a combined vehicle routing and truck driver scheduling problem with time windows. Goel (2009) presents two scheduling approaches embedded into a large neighbourhood search algorithm. The *naïve* method is a greedy approach that determines a single label for each location to be visited. Each label corresponds to the driver's state at the respective location. The *multilabel* method is a breadth-first search method that iteratively determines multiple labels corresponding to different possible driver states. Both approaches only consider a subset of the provisions imposed by regulation (EC) No. 561/2006. Kok et al. (2009) present a dynamic programming approach that extends the naïve method considering provisions of the regulation ignored in Goel (2009).

The approaches by Goel (2009) and Kok et al. (2009) have the shortcoming that, for a given tour, they cannot guarantee finding a schedule complying with the regulation, if such a schedule does, indeed, exist. This paper presents a method for scheduling driving and working hours of truck drivers that is guaranteed to find a schedule complying with the regulation, if such a schedule exists.

Being capable of determining whether a given sequence of locations can be visited in compliance with legislatory constraints is of particular importance for several reasons. Scheduling driving and working hours of truck drivers is a subproblem of combined vehicle routing and truck driver scheduling problems arising in many real-life applications. Many successful approaches for solving rich vehicle routing problems are local search-based optimisation methods, which generate a sequence of feasible tours for each vehicle until a near-optimal solution is found. Without a sophisticated method for scheduling driving and working hours of truck drivers, many tours will never be considered within the optimisation process because simple scheduling algorithms fail in finding a feasible truck driver schedule for the tour. Thus, some parts of the search space may never be explored. This may result in reduced overall performance of the optimisation process.

Another reason is that many real-life applications require interaction with human decision makers. Powell et al. (2002) report experiences on implementing an interactive dispatching system for a truckload motor carrier. They found that only 60%–80% of all dispatching decisions made by the software system were accepted by the human decision maker. In all other cases dispatching decisions were made manually. If a scheduling algorithm cannot find a schedule complying with the regulation although such a schedule exists, experienced human decision makers may find it easy to manually determine solutions superior to the solutions determined by the computer system. Furthermore, the computer system may fail in processing manually generated solutions. As a consequence, the scheduling algorithm and potentially the entire dispatching system may be questioned by the human decision maker. An interactive dispatching system that is not fully approved by the human decision maker will hardly be successful in any operational setting.

This paper presents a method for scheduling driving and working hours of truck drivers that is guaranteed to find a feasible truck driver schedule for a given tour, if such a schedule exists. This method can be embedded into optimisation methods for combined vehicle routing and truck driver scheduling problems. The remainder of this paper is organised as follows. Regulation (EC) No. 561/2006 is summarised in §2 and discussed in §3. In §4 a formal model of the European Union Truck Driver Scheduling Problem (EU-TDSP) is given. Section 5 introduces conditions for pseudofeasibility which relax the conditions for feasibility presented in §4. This relaxation allows us to myopically construct truck driver schedules without considering future driver activities and constraints imposed on them. Furthermore, §5 gives dominance criteria, which help us in reducing the number of partial schedules that must be explored to solve the EU-TDSP. Normality conditions are presented that guide us when solving the EU-TDSP. Section 6 presents a scheduling method that solves the EU-TDSP by constructing pseudofeasible schedules satisfying these normality conditions. Computational experiments presented in §7 show that this method significantly outperforms previous approaches in terms of the number of tours for which a feasible truck driver schedule can be found.

2. Regulation (EC) No. 561/2006

Regulation (EC) No. 561/2006 distinguishes between four driver activities: rest periods, breaks, driving time, and other work. Rest periods are longer periods during which a driver may freely dispose of her or his time. Breaks are short periods exclusively used for recuperation during which a driver may not carry out any work. Driving time refers to the time during which a driver is operating a vehicle and includes any time during which the vehicle is temporarily stationary because of reasons related to driving, e.g., traffic jams. Other work refers to any work except for driving and includes time spent for loading or unloading, cleaning and technical maintenance, customs, etc.

The regulation places constraints on the minimum duration and frequency in which rest periods must be taken. A weekly rest period of at least 45 hours shall start no later than 144 hours after the end of the previous weekly rest period. Alternatively, a reduced weekly rest period of at least 24 hours may be taken. However, before the end of the third calendar week (i.e., the week starting Monday 0.00 hours and ending Sunday 23.59 hours) after the reduced weekly rest period has been taken, the reduction shall be compensated by an equivalent period of rest attached to another rest period of at least nine hours. In any two consecutive calendar weeks at least two weekly rest periods must be taken, and the duration of at least one of them must be of 45 hours or more. Weekly rest periods falling in two calendar weeks may be counted in either week but not in both.

Regular daily rest periods of at least 11 hours must be completed within each period of 24 hours after the end of the previous daily rest period or weekly rest period. This regular daily rest period may be replaced by a reduced daily rest period of at least nine hours. However, a driver may have at most three reduced daily rest periods between any two weekly rest periods. Alternatively, a regular daily rest period may be replaced by two periods, the first of which must be a rest period of at least three hours and the second a rest period of at least nine hours. A daily rest period may be extended beyond 24 hours after the end of the previous daily rest period or weekly rest period. Depending on the amount of extension the rest period may be interpreted as a weekly rest period.

The accumulated driving time between the end of a daily or weekly rest period and the beginning of the next daily or weekly rest period must not exceed nine hours. However, twice during a calendar week the daily driving time may be extended up to at most ten hours. The accumulated working time during a calendar week must not exceed 60 hours. The accumulated driving time during a calendar week must not exceed 56 hours. The total accumulated driving time during any two consecutive calendar weeks must not exceed 90 hours. In any period of four months, the average working time during a calendar week must not exceed 48 hours.

A full break is completed after an uninterrupted break of at least 45 minutes or after a break of at least 30 minutes, which is preceded by a break of at least 15 minutes. A driver may drive for at most $4\frac{1}{2}$ hours without taking a rest period or a full break.

3. Discussion

The regulation imposes constraints on driving and working hours between weekly rest periods and on driving and working hours ranging across several weeks. For most real-life operations detailed work plans for truck drivers are only prepared for a relatively short time frame, e.g., one week. If we only schedule driving and working hours of truck drivers focussing on the working period of one week, we do not need to explicitly consider the provisions of the regulation for longer time periods. Instead, we can implicitly consider these regulations by modifying the limits on the amount of driving and working within the week of concern, which are determined by the amount of working and driving in the previous weeks. As most truck drivers take weekly rest periods on the weekend, we will assume throughout this paper, that schedules are only generated for a planning horizon of at most six days (144 hours), which does not range across the night from Sunday to Monday. Furthermore, we assume that, at the beginning of the planning horizon, the driver returns from a weekly rest period. Thus, the accumulated driving and working time within the planning horizon is limited to 56 hours and 60 hours, or less, if the maximum amount of biweekly driving would be otherwise exceeded. It must be noted, that without the above assumptions a driver could legally drive for up to 67 hours within a period of six days. This is a result of an inconsistency of the regulation, which imposes some constraints on the time period between Monday 0.00 hours and Sunday 23.59 hours, and others on the period between two consecutive weekly rest periods. Thus, the weekly limits on the accumulated driving and working time are not binding if a driver works from Friday 0.00 hours to Wednesday 23.59 hours.

According to the regulation, break and rest periods may be taken in two parts. Figure 1 illustrates how a short break of 15 minutes followed by a break of 30 minutes can be scheduled to complete a full break after $4\frac{1}{2}$ hours of driving. Regulation (EC) No. 561/2006 does not specify a minimum amount of driving that shall be conducted before taking the first part of a break taken in two parts. As the amount of driving before the first part may be arbitrarily small, we can conclude that de facto a driver may



Figure 1 Break Taken in Two Parts

Notes. If a short break of 15 minutes is scheduled after a driving period of duration $t \in [0\text{ h}, 4\frac{1}{2}\text{ h}]$, a break of 30 minutes is sufficient to begin another driving period of $4\frac{1}{2}$ hours.

take such a short break even before starting to drive. Analogously, the first part of a rest period taken in two parts may be scheduled before the driver starts driving.

The regulation allows extending the daily driving time from 9 to 10 hours and reducing the length of a daily rest period from 11 to 9 hours. Although it may be desirable to maximise the productive time of the vehicle, making use of the respective provisions also bears some risks. Schedules in which the daily driving time is near its limit are very prone to large delays caused by prolonged handling operations or traffic congestion. According to European Commission (2003), every day 7,500 kilometres of European highways are blocked by traffic jams. Although heavy traffic can be predicted based on historic traffic flow data, traffic jams caused by accidents can occur anywhere at any time. If robustness issues are ignored when generating truck driver schedules, small delays caused by such traffic jams may result in the need for an additional daily rest period, and the complete schedule may become void. On the other hand, if schedules are generated without extending daily driving times and reducing daily rest periods, a certain level of robustness is guaranteed for all schedules. Figure 2 shows an example in which a delay of 15 minutes during handling operations and another delay of one hour because of traffic congestion can be compensated for by increasing the daily driving time and reducing the length of a daily rest period. Without this compensation a delay of more than 11 hours would be the result of those delays. To guarantee a certain level of robustness, we will assume in the remainder of this paper that daily driving times must not be extended and daily rest periods must not be reduced when determining truck driver schedules.

4. The Truck Driver Scheduling Problem

In this section we describe the truck driver scheduling problem for truck drivers in the European Union. Let us consider a sequence of locations denoted by $n_1, n_2, \ldots, n_\lambda$ that shall be visited by a truck driver. At each location n_μ some stationary work of duration w_μ shall be conducted. This work shall begin within a given time window denoted by T_μ. The (positive) driving time required for moving from node n_μ to the $n_{\mu+1}$ shall be denoted by $\delta_{\mu,\mu+1}$. The *European Union truck driver scheduling problem* (EU-TDSP) is the problem of determining whether driving and working hours of a truck driver can be scheduled in such a way that all locations are visited within the given time windows and that regulation (EC) No. 561/2006 is complied with.

To give a formal model of the problem, let us denote by DRIVE any period during which the driver is driving; by WORK any period of working time in which the driver is not driving; by REST any period of rest after which a daily rest period is completed; by REST1 any period of rest corresponding to the first part of a daily rest period taken in two parts; by BREAK any break period after which a full break is completed; by BREAK1 any period corresponding to the first part of a break period taken in two parts; and by IDLE any period in which drivers are neither working nor driving and that is not regarded as a break or rest period.

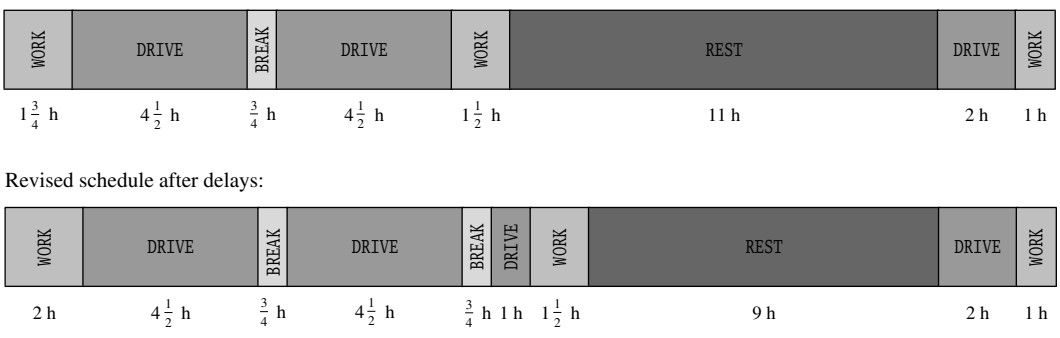


Figure 2 Extended Daily Driving Times and Reduced Daily Rest Periods May Be Used to Account for Delays

Note that a period denoted by REST may be a regular rest period of at least 11 hours or the second part of a daily rest period taken in two parts. Similarly, a period denoted by BREAK may be a full break of at least 45 minutes or the second part of a break period taken in two parts.

A truck driver schedule can be specified by a sequence of activities to be performed by the drivers. Let $\mathcal{A} := \{a = (a^{\text{type}}, a^{\text{length}}) \mid a^{\text{type}} \in \{\texttt{DRIVE}, \texttt{WORK}, \texttt{REST}, \texttt{REST1}, \texttt{BREAK}, \texttt{BREAK1}, \texttt{IDLE}\}, a^{\text{length}} > 0\}$ denote the set of driver activities to be scheduled. Let $\langle\langle . \rangle\rangle$ be an operator that concatenates different activities. Thus, $a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ denotes a schedule in which for each $i \in \{1, 2, \ldots, k-1\}$ activity a_{i+1} is performed immediately after activity a_i. For a given schedule $s := a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ and $1 \le i \le k$, let $s_{1,i} := a_1 \,.\, a_2 \,.\, \ldots \,.\, a_i$ denote the partial schedule composed of activities a_1 to a_i. Recall that we assumed that at the beginning of the planning horizon, the driver returns from a weekly rest period. For simplicity we will thus only consider schedules $s := a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ that begin with a rest period, i.e., $a_1^{\text{type}} = \texttt{REST}$.

Table 1 gives an overview of the parameters imposed by the regulation. We use the following notation for determining whether a schedule complies with the regulation. For each schedule $s := a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ with $a_1^{\text{type}} = \texttt{REST}$ we denote the completion time by l_s^{end}, the time of completion of the last daily rest period by $l_s^{\text{last_rest}}$, the accumulated driving time since completion of the last rest period by l_s^{daily}, the accumulated driving time since completion of the last full break or rest period by l_s^{nonstop},

Table 1 Parameters Imposed by Regulation (EC) No. 561/2006

Notation	Value	Description
t^{rest}	11 hours	The minimum duration of a daily rest period taken en-block
t^{rest1}	3 hours	The minimum duration of the first part of a daily rest period taken in two parts
t^{rest2}	9 hours	The minimum duration of the second part of a daily rest period taken in two parts
t^{break}	$\frac{3}{4}$ hours	The minimum duration of a full break taken en-block
t^{break1}	$\frac{1}{4}$ hour	The minimum duration of the first part of a break taken in two parts
t^{break2}	$\frac{1}{2}$ hour	The minimum duration of the second part of a break taken in two parts
t^{daily}	9 hours	The maximum accumulated driving time between a daily or weekly rest period and the next daily or weekly rest period
t^{nonstop}	$4\frac{1}{2}$ hours	The maximum driving time without a stationary period representing a rest period or full break
t^{day}	24 hours	The maximum time after the end of the previous daily or weekly rest period after which a new daily rest period must be completed

the minimum time required to complete a new daily rest period by l_s^{rest}, and the minimum time required to complete a new full break by l_s^{break}. All of these values can be recursively computed during schedule generation by

$$l_{s_{1,1}}^{\text{end}} := a_1^{\text{length}} \quad \text{and} \quad l_{s.a}^{\text{end}} := l_s^{\text{end}} + a^{\text{length}},$$

$$l_{s_{1,1}}^{\text{last_rest}} := l_{s_{1,1}}^{\text{end}} \quad \text{and} \quad l_{s.a}^{\text{last_rest}} := \begin{cases} l_{s.a}^{\text{end}} & \text{if } a^{\text{type}} = \texttt{REST}, \\ l_s^{\text{last_rest}} & \text{else}, \end{cases}$$

$$l_{s_{1,1}}^{\text{daily}} := 0 \quad \text{and}$$

$$l_{s.a}^{\text{daily}} := \begin{cases} 0 & \text{if } a^{\text{type}} = \texttt{REST}, \\ l_s^{\text{daily}} + a^{\text{length}} & \text{else if } a^{\text{type}} = \texttt{DRIVE}, \\ l_s^{\text{daily}} & \text{else}, \end{cases}$$

$$l_{s_{1,1}}^{\text{nonstop}} := 0 \quad \text{and}$$

$$l_{s.a}^{\text{nonstop}} := \begin{cases} 0 & \text{if } a^{\text{type}} \in \{\texttt{REST}, \texttt{REST1}, \texttt{BREAK}\}, \\ l_s^{\text{nonstop}} + a^{\text{length}} & \text{else if } a^{\text{type}} = \texttt{DRIVE}, \\ l_s^{\text{nonstop}} & \text{else}, \end{cases}$$

$$l_{s_{1,1}}^{\text{rest}} := t^{\text{rest}} \quad \text{and} \quad l_{s.a}^{\text{rest}} := \begin{cases} t^{\text{rest}} & \text{if } a^{\text{type}} = \texttt{REST}, \\ t^{\text{rest2}} & \text{else if } a^{\text{type}} = \texttt{REST1}, \\ l_s^{\text{rest}} & \text{else}, \end{cases}$$

$$l_{s_{1,1}}^{\text{break}} := t^{\text{break}} \quad \text{and}$$

$$l_{s.a}^{\text{break}} := \begin{cases} t^{\text{break}} & \text{if } a^{\text{type}} \in \{\texttt{REST}, \texttt{REST1}, \texttt{BREAK}\}, \\ t^{\text{break2}} & \text{else if } a^{\text{type}} = \texttt{BREAK1}, \\ l_s^{\text{break}} & \text{else}. \end{cases}$$

For a given sequence of locations $n_1, n_2, \ldots, n_\lambda$ and a work plan $s = a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ with $a_1^{\text{type}} = \texttt{REST}$, let us denote by $i(\mu)$ the index corresponding to the μth stationary work period; i.e., $a_{i(\mu)}$ corresponds to the work performed at location n_μ. With this notation we can now give a formal model of the problem.

The European Union truck driver scheduling problem (EU-TDSP) is the problem of determining for a given sequence of locations, $n_1, n_2, \ldots, n_\lambda$, whether a schedule $s := a_1 \,.\, a_2 \,.\, \ldots \,.\, , a_k$ with $a_1^{\text{type}} = \texttt{REST}$ exists that contains exactly λ work activities and satisfies

$$\sum_{\substack{i(1) \le j \le i(\lambda) \\ a_j^{\text{type}} = \texttt{DRIVE}}} a_j^{\text{length}} = \sum_{\substack{1 \le j \le k \\ a_j^{\text{type}} = \texttt{DRIVE}}} a_j^{\text{length}}, \tag{1}$$

$$a_{i(\mu)}^{\text{length}} = w_\mu \quad \text{for each } \mu \in \{1, 2, \ldots, \lambda\}, \tag{2}$$

$$l_{s_{1,i(\mu)-1}}^{\text{end}} \in T_\mu \quad \text{for each } \mu \in \{1, 2, \ldots, \lambda\}, \tag{3}$$

$$\sum_{\substack{i(\mu)\le j\le i(\mu+1)\\ a_j^{\text{type}}=\texttt{DRIVE}}} a_j^{\text{length}} = \delta_{\mu,\mu+1} \quad \text{for each } \mu \in \{1,2,\ldots,\lambda-1\}, \tag{4}$$

$$l_{s_{1,i}}^{\text{daily}} \le t^{\text{daily}} \quad \text{for each } 1 < i \le k, \tag{5}$$

$$l_{s_{1,i}}^{\text{nonstop}} \le t^{\text{nonstop}} \quad \text{for each } 1 < i \le k, \tag{6}$$

$$a_i^{\text{length}} \ge l_{s_{1,i-1}}^{\text{break}} \quad \text{for each } 1 < i \le k \text{ with } a_i^{\text{type}} = \texttt{BREAK}, \tag{7}$$

$$a_i^{\text{length}} \ge l_{s_{1,i-1}}^{\text{rest}} \quad \text{for each } 1 < i \le k \text{ with } a_i^{\text{type}} = \texttt{REST}, \tag{8}$$

$$a_i^{\text{length}} \ge t^{\text{break1}} \quad \text{for each } 1 < i \le k \text{ with } a_i^{\text{type}} = \texttt{BREAK1}, \tag{9}$$

$$a_i^{\text{length}} \ge t^{\text{rest1}} \quad \text{for each } 1 < i \le k \text{ with } a_i^{\text{type}} = \texttt{REST1}, \tag{10}$$

$$l_{s_{1,i}}^{\text{end}} + l_{s_{1,i}}^{\text{rest}} \le l_{s_{1,i}}^{\text{last_rest}} + t^{\text{day}} \quad \text{for each } 1 < i \le k \text{ with } a_i^{\text{type}} \ne \texttt{REST}. \tag{11}$$

Condition (1) demands that all driving is conducted between the first and the last work activity. Condition (2) demands that the duration of the μth work activity matches the specified work duration at location n_μ. Condition (3) demands that each work activity begins within the corresponding time window. Condition (4) demands that the accumulated driving time between two work activities matches the driving time required to move from one location to the other. Conditions (5) and (6) demand that the maximum amount of daily or nonstop driving does not exceed the limit given by the regulation. Conditions (7) to (10) demand that each break or rest period has the minimum duration required by the regulation. Condition (11) demands that a new daily rest period can be completed within 24 hours since the end of the last rest period. In the remainder of this paper, we will say that a schedule $s := a_1 . a_2 . \ldots . a_k$ with $a_1^{\text{type}} = \texttt{REST}$ is *feasible* if and only if it satisfies conditions (1) to (11).

5. Pseudofeasibility and Normality

An obvious approach in trying to solve the EU-TDSP is to iteratively append activities to a partial schedule until a feasible schedule is found. One of the difficulties in solving the EU-TDSP by such an approach is that it is not clear how to determine the best duration of rest periods without looking at all future activities to be scheduled. Obviously, we do not want to schedule unnecessarily long rest periods, because this may reduce the amount of time available until the end of one of the time windows. On the other hand, if a truck arrives at a location before the opening of the time window, the driver may have to wait idle. This idle time may unnecessarily increase the amount of time since completion of the last rest period. As a new daily rest period must be completed within 24 hours after the end of the previous rest period, any rest period should be long enough to avoid unnecessary idle time.

Kok et al. (2009) tackled this problem by using a heuristic approach which tries to extend rest periods in such a way that no idle time at the locations is required. Whenever a rest period is scheduled on a trip from location n_μ to location $n_{\mu+1}$, the method by Kok et al. (2009) sets the duration of the rest period to a value that eliminates all idle time on this trip. Thus, idle time at location $n_{\mu+1}$ is only scheduled when no rest period is scheduled during the trip from n_μ to $n_{\mu+1}$. A different approach, which ignores constraint (11) during schedule construction, is presented by Goel (2009). Achieving compliance with (11) is attempted by using a repair method in a postprocessing step. In some cases, however, time window constraints prevent the repair method from sufficiently increasing the duration of some of the rest periods and compliance with condition (11) cannot be achieved.

In the following, we will see that we can restrict our search to schedules in which each rest period appended to a partial schedule s has a duration of exactly l_s^{rest}. For this, we relax constraint (11) of the EU-TDSP in such a way that the repair method presented by Goel (2009) is guaranteed to succeed. The repair method extends rest periods and reduces subsequent idle time in such a way that the completion time of the schedule remains unchanged. For any work plan s let us denote the accumulated slack time since completion of the last rest period by l_s^{slack}. This value constrains the amount by which the duration of the last rest period can be extended without increasing the completion time of the schedule. It can be recursively computed by

$$l_{s_{1,1}}^{\text{slack}} := 0 \quad \text{and}$$

$$l_{s.a}^{\text{slack}} := \begin{cases} 0 & \text{if } a^{\text{type}} = \texttt{REST}, \\ l_s^{\text{slack}} + a^{\text{length}} & \text{else if } a^{\text{type}} = \texttt{IDLE}, \\ l_s^{\text{slack}} & \text{else.} \end{cases}$$

By extending the duration of the last rest period in the schedule, the start time of some subsequent work activities may be pushed to a later point in time. Thus, we need to know the maximum amount by which the duration of the rest period can be increased without violating any time windows or exceeding the closing time of the time window corresponding to the next working period to be scheduled. Let us denote this value by l_s^{push}. Let us furthermore denote by $\tau_s^{\max}$ the closing time of the time window corresponding to the next work location to be visited. For every partial

schedule s, the last rest period in the schedule can be increased by at most $l_s^{\text{slack}} + \tau_s^{\max} - l_s^{\text{end}}$ without exceeding the closing time of the next time window. Thus, l_s^{push} can be recursively computed by

$$l_{s_{1,1}}^{\text{push}} := \tau_{s_{1,1}}^{\max} - l_{s_{1,1}}^{\text{end}} \quad \text{and}$$

$$l_{s.a}^{\text{push}} := \begin{cases} \tau_{s.a}^{\max} - l_{s.a}^{\text{end}} & \text{if } a^{\text{type}} = \texttt{REST}, \\ \min\{l_s^{\text{push}}, l_s^{\text{slack}} + \tau_s^{\max} - l_s^{\text{end}}\} & \\ & \text{else if } a^{\text{type}} = \texttt{WORK}, \\ l_s^{\text{push}} & \text{else.} \end{cases}$$

Hence, the maximum amount by which the duration of the last rest period in schedule s may be extended without violating time window constraints or increasing the completion time is

$$l_s^{\text{extend}} := \min\{l_s^{\text{slack}}, l_s^{\text{push}}\}.$$

Now, let us replace condition (11) of the EU-TDSP by condition

$$l_{s_{1,i}}^{\text{end}} + l_{s_{1,i}}^{\text{rest}} \le l_{s_{1,i}}^{\text{last_rest}} + l_{s_{1,i}}^{\text{extend}} + t^{\text{day}} \quad \text{for each} \quad 1 < i \le k \text{ with } a_i^{\text{type}} \ne \texttt{REST}, \quad (11')$$

and let us say that a schedule $s := a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ with $a_1^{\text{type}} = \texttt{REST}$ is *pseudofeasible* if and only if it satisfies conditions (1) to (10) and (11′).

As $l_s^{\text{extend}} \ge 0$ for any feasible schedule s, we know that each feasible schedule is pseudofeasible. Figure 3 illustrates the idea behind replacing condition (11) by condition (11′). The first partial schedule in Figure 3 shows a situation in which a new rest period would be required before the opening of the time window of the next work location. However, it is not possible to append a new rest period to the schedule, because the time window of the location closes before a new rest period can be completed. The second partial schedule shows a schedule with $l_s^{\text{end}} > l_s^{\text{last_rest}} + t^{\text{day}} - l_s^{\text{rest}}$ and $l_s^{\text{end}} \le l_s^{\text{last_rest}} + l_s^{\text{extend}} + t^{\text{day}} - l_s^{\text{rest}}$. Thus, the schedule violates condition (11) and satisfies condition (11′). However, we can achieve compliance with condition (11) because the duration of the last rest period can be extended by as much as l_s^{extend} without violating time windows or increasing the completion time. At the bottom of Figure 3, we see that the schedule, which is obtained by increasing the duration of the last rest period by l_s^{extend} (and reducing the duration of subsequent idle periods), is feasible. As we can similarly transform every pseudofeasible schedule into a feasible schedule, the EU-TDSP is equivalent to the problem of determining whether a pseudofeasible schedule $s := a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ with $a_1^{\text{type}} = \texttt{REST}$ exists.

The advantage of searching for pseudofeasible schedules is that we do not need to know the best duration of rest periods during schedule generation. Instead, we can start by scheduling rest periods which are as short as possible. In a postprocessing step, the duration of these rest periods can be extended to the required length.

In general, the set of all pseudofeasible schedules is too large to be enumerated. Let us now define criteria indicating which schedules can safely be ignored when solving the EU-TDSP.

Definition 1. A schedule s' *dominates* another schedule s'' if some work plan $\tilde{s}'$ exists such that $s' \,.\, \tilde{s}' \,.\, \tilde{s}''$ is pseudofeasible for all $\tilde{s}''$ for which $s'' \,.\, \tilde{s}''$ is pseudofeasible.

Thus, if the schedule s'' is dominated by schedule s' and a pseudofeasible schedule for tour $n_1, n_2, \ldots, n_\lambda$ exists that begins with the activities in schedule s'', then there also exists a pseudofeasible schedule for

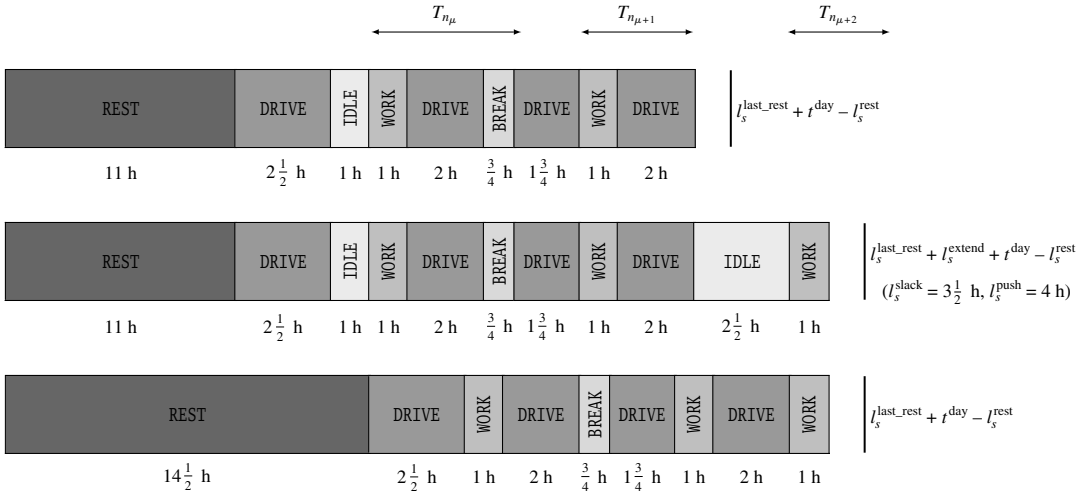


Figure 3 Partial Schedules and the Time When the Next Rest Period Must Be Scheduled w.r.t. Condition (11) or (11')

tour $n_1, n_2, \ldots, n_\lambda$ that begins with the activities in schedule s'. Therefore, s'' is not required for solving the EU-TDSP. Criteria for dominance are given in the following two lemmata.

LEMMA 1. *Let s', s'' be pseudofeasible schedules for the partial tour $n_1, \ldots, n_\mu$ with $1 \le \mu < \lambda$. Schedule s' dominates s'' if*

$$l_{s'}^{end} \le l_{s''}^{end}, \quad l_{s'}^{daily} \le l_{s''}^{daily}, \quad l_{s'}^{nonstop} \le l_{s''}^{nonstop},$$
$$l_{s'}^{break} \le l_{s''}^{break}, \quad l_{s'}^{rest} \le l_{s''}^{rest},$$
$$l_{s'}^{last_rest} + l_{s'}^{slack} \ge l_{s''}^{last_rest} + l_{s''}^{slack}, \quad \textit{and}$$
$$l_{s'}^{last_rest} + l_{s'}^{push} \ge l_{s''}^{last_rest} + l_{s''}^{push}.$$

PROOF. If $l_{s'}^{\text{end}} = l_{s''}^{\text{end}}$, any activity a that can be appended to the schedule s'' without violating constraints (1) to (10) and (11′) can be appended to the schedule s' without violating constraints (1) to (10) and (11′). After appending a, the conditions of the lemma equally hold for $s'.a$ and $s''.a$. Thus, any sequence of activities that can be appended to s'' can be appended to s' without violating the conditions for pseudofeasibility. If $l_{s'}^{\text{end}} < l_{s''}^{\text{end}}$ we can set $\tilde{s}' := (\texttt{IDLE}, l_{s''}^{\text{end}} - l_{s'}^{\text{end}})$. We have $l_{s'.\tilde{s}'}^{\text{end}} = l_{s''}^{\text{end}}$ and the conditions of the lemma hold for $s'.\tilde{s}'$ and s''. Thus, any sequence of activities that can be appended to s'' can be appended to $s'.\tilde{s}'$. □

LEMMA 2. *Let s', s'' pseudofeasible schedules for the partial tour $n_1, \ldots, n_\mu$ with $1 \le \mu < \lambda$. Schedule s' dominates s'' if*

$$l_{s'}^{end} + l_{s'}^{rest} + t^{rest1} + t^{break1} \le l_{s''}^{end}.$$

PROOF. We can set $\tilde{s}'$ to a combination of activities of type REST, REST1, BREAK1, and/or IDLE in such a way that $s'.\tilde{s}'$ dominates s'' with respect to the conditions of Lemma 1. □

To be able to efficiently solve the EU-TDSP by subsequently constructing pseudofeasible schedules for partial tours $n_1, \ldots, n_\mu$, we now define several normality conditions. For notational reasons, we can require that each pseudofeasible schedule $s = a_1 . a_2 . \ldots . a_k$ satisfies the following condition:

$$\text{for each } 1 < i \le k\text{: } a_{i-1}^{\text{type}} \ne a_i^{\text{type}}. \tag{N1}$$

If we have a pseudofeasible schedule violating (N1), we can simply transform the schedule into a pseudofeasible schedule satisfying (N1) by merging all subsequent activities of the same type.

We can demand that the duration of each break or rest period is as short as possible, i.e., that any schedule $s = a_1 . a_2 . \ldots . a_k$ satisfy the following:

for each $1 < i \le k$ with $a_i^{\text{type}} = \texttt{BREAK}$:

$$a_i^{\text{length}} = l_{s_{1,i-1}}^{\text{break}}, \tag{N2}$$

for each $1 < i \le k$ with $a_i^{\text{type}} = \texttt{REST}$:

$$a_i^{\text{length}} = l_{s_{1,i-1}}^{\text{rest}}, \tag{N3}$$

for each $1 < i \le k$ with $a_i^{\text{type}} = \texttt{BREAK1}$:

$$a_i^{\text{length}} = t^{\text{break1}}, \tag{N4}$$

and

for each $1 < i \le k$ with $a_i^{\text{type}} = \texttt{REST1}$:

$$a_i^{\text{length}} = t^{\text{rest1}}. \tag{N5}$$

Any schedule violating (N2), (N3), (N4), or (N5) can be transformed by shortening the break or rest period and inserting an idle period of appropriate length.

The first part of a break or rest is only beneficial if it reduces the time required to complete a full break or daily rest period. Therefore, we can require that each schedule $s = a_1 . a_2 . \ldots . a_k$ satisfy the following:

for each $1 < i \le k$ with $a_i^{\text{type}} = \texttt{BREAK1}$:

$$l_{s_{1,i-1}}^{\text{break}} > t^{\text{break2}}, \tag{N6}$$

and

for each $1 < i \le k$ with $a_i^{\text{type}} = \texttt{REST1}$:

$$l_{s_{1,i-1}}^{\text{rest}} > t^{\text{rest2}}. \tag{N7}$$

If a pseudofeasible schedule does not satisfy condition (N6), the short break does not bring any benefit and can be replaced by an idle period of the same length. If a pseudofeasible schedule does not satisfy condition (N7), the short rest does not bring any benefit compared to a break period and can be replaced by a break period of the same length.

As a break period preceding a rest period or preceded by a rest period does not bring any benefit, we can furthermore require that each schedule $s = a_1 . a_2 . \ldots . a_k$ satisfies the following:

for each $1 < i \le k$ with $a_{i-1}^{\text{type}} = \texttt{BREAK}$:

$$a_i^{\text{type}} \notin \{\texttt{REST1}, \texttt{REST}\}, \tag{N8}$$

and

for each $1 < i \le k$ with $a_{i-1}^{\text{type}} \in \{\texttt{REST1}, \texttt{REST}\}$:

$$a_i^{\text{type}} \ne \texttt{BREAK}. \tag{N9}$$

If a schedule does not satisfy condition (N8) or (N9), we can replace the break period by an idle period of the same length.

Taking break or rest periods in two parts does not generate any benefit if the first part is taken immediately before the second part. Thus, we can require that each schedule $s = a_1 . a_2 . \ldots . a_k$ satisfies the following:

for each $1 < i \le k$ with $a_{i-1}^{\text{type}} = \texttt{BREAK1}$:

$$a_i^{\text{type}} \ne \texttt{BREAK}, \tag{N10}$$

and

for each $1 < i \le k$ with $a_{i-1}^{\text{type}} = \texttt{REST1}$:

$$a_i^{\text{type}} \neq \texttt{REST}. \tag{N11}$$

If a pseudofeasible schedule does not satisfy condition (N10) or (N11), we can delete the first part of the break or rest and increase the length of the second part.

As idle activities represent unproductive periods, they should only be scheduled if they are required due to an early arrival at a work location. We can demand that idle periods only be scheduled immediately before work periods, i.e., that each schedule $s = a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ satisfy the following:

for each $1 < i \le k$ with $a_{i-1}^{\text{type}} = \texttt{IDLE}$:

$$a_i^{\text{type}} = \texttt{WORK}. \tag{N12}$$

If a pseudofeasible schedule does not satisfy condition (N12), we can remove the idle activity and insert it immediately before the next work period.

For any schedule s let us denote by $\tau_s^{\min}$ the opening time of the time window corresponding to the next work location to be visited. We can require that each idle period be as short as possible, i.e., that each schedule $s = a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ satisfy the following:

for each $1 < i < k$ with $a_i^{\text{type}} = \texttt{IDLE}$:

$$a_i^{\text{length}} = \tau_{s_{1,\,i-1}}^{\min} - l_{s_{1,\,i-1}}^{\text{end}}. \tag{N13}$$

If a pseudofeasible schedule does not satisfy condition (N13), we can reduce the length of the idle activity period and insert an idle period of appropriate length after the next work period.

Taking break periods in two parts is only beneficial if this helps us reduce the amount of idle time. We can demand that activities of type BREAK1 only be scheduled before idle periods or work periods, i.e., that each schedule $s = a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ satisfy the following:

for each $1 < i \le k$ with $a_i^{\text{type}} = \texttt{BREAK1}$:

$$a_{i+1}^{\text{type}} \in \{\texttt{WORK}, \texttt{IDLE}\}. \tag{N14}$$

If a pseudofeasible schedule does not satisfy condition (N14), we can remove the activity of type BREAK1 and insert it immediately before the next break, idle, or work period. If, after this transformation, the activity of type BREAK1 is followed by an activity of type BREAK, we can do the same as in the case when (N10) is violated.

Eventually, we can require that all driving time be conducted as early as possible, i.e., that each schedule $s = a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ satisfy the following:

for each $1 < i < k$ with $a_i^{\text{type}} \in \{\texttt{BREAK}, \texttt{REST1}\}$ and $a_{i+1}^{\text{type}} = \texttt{DRIVE}$:

$$l_{s_{1,\,i-1}}^{\text{nonstop}} = t^{\text{nonstop}}, \tag{N15}$$

and

for each $1 < i < k$ with $a_i^{\text{type}} = \texttt{REST}$ and $a_{i+1}^{\text{type}} = \texttt{DRIVE}$:

$$l_{s_{1,\,i-1}}^{\text{daily}} = t^{\text{daily}} \quad \text{or} \quad l_{s_{1,\,i-1}}^{\text{nonstop}} = t^{\text{nonstop}} \quad \text{or}$$
$$l_{s_{1,\,i-1}}^{\text{end}} + l_{s_{1,\,i-1}}^{\text{rest}} = l_{s_{1,\,i-1}}^{\text{last_rest}} + l_{s_{1,\,i-1}}^{\text{extend}} + t^{\text{day}}. \tag{N16}$$

If a pseudofeasible schedule does not satisfy condition (N15) or (N16), we can move some of the driving time to just before the break or rest.

All schedules obtained by modifying a schedule as described above are pseudofeasible and the modified schedule dominates the original schedule with respect to the conditions of Lemma 1. Obviously, some of the modifications may create a violation of another normality condition. However, by iteratively transforming a pseudofeasible schedule, we can achieve a pseudofeasible schedule satisfying conditions (N1) to (N16). We will say that a schedule satisfying (N1) to (N16) is in *normal form*. As we can transform any pseudofeasible schedule into a pseudofeasible schedule in normal form, the EU-TDSP is equivalent to the problem of determining whether a pseudofeasible schedule $s := a_1 \,.\, a_2 \,.\, \ldots \,.\, a_k$ with $a_1^{\text{type}} = \texttt{REST}$ exists that is in normal form.

In the next section we will present a scheduling method that constructs pseudofeasible schedules in normal form until the EU-TDSP is solved.

6. Scheduling Method

This section presents a scheduling method that iteratively takes a set of pseudofeasible schedules in normal form for a partial tour $n_1, n_2, \ldots, n_\mu$ and extends each schedule to construct pseudofeasible schedules in normal form for tour $n_1, n_2, \ldots, n_\mu, n_{\mu+1}$. This process is repeated until the EU-TDSP for tour $n_1, n_2, \ldots, n_\lambda$ is solved.

Because we assume that at the beginning of the planning horizon, the driver returns from a weekly rest period, we only consider schedules which begin with a weekly rest period with a duration of at least $t^{\text{weekly}} := 45$ hours. Then, the scheduling method illustrated in Figure 4 is initially invoked with $\mu = 1$ and

$$\begin{aligned}\mathcal{S}_1 = \big\{&\big(\texttt{REST}, \max\{t^{\text{weekly}}, \min T_1\}\big)\cdot(\texttt{WORK}, w_1),\\ &\big(\texttt{REST}, \max\{t^{\text{weekly}}, \min T_1 - t^{\text{break1}}\}\big)\cdot(\texttt{BREAK1}, t^{\text{break1}})\\ &\cdot(\texttt{WORK}, w_1), \big(\texttt{REST}, \max\{t^{\text{weekly}}, \min T_1 - t^{\text{rest1}}\}\big)\\ &\cdot(\texttt{REST1}, t^{\text{rest1}})\cdot(\texttt{WORK}, w_1),\\ &\big(\texttt{REST}, \max\{t^{\text{weekly}}, \min T_1 - t^{\text{rest1}} - t^{\text{break1}}\}\big)\\ &\cdot(\texttt{REST1}, t^{\text{rest1}})(\texttt{BREAK1}, t^{\text{break1}})\cdot(\texttt{WORK}, w_1)\big\}.\end{aligned}$$

Note that any other pseudofeasible schedule in normal form for $\mu = 1$ is dominated by one of the schedules in $\mathcal{S}_1$. Within the scheduling method δ_s denotes

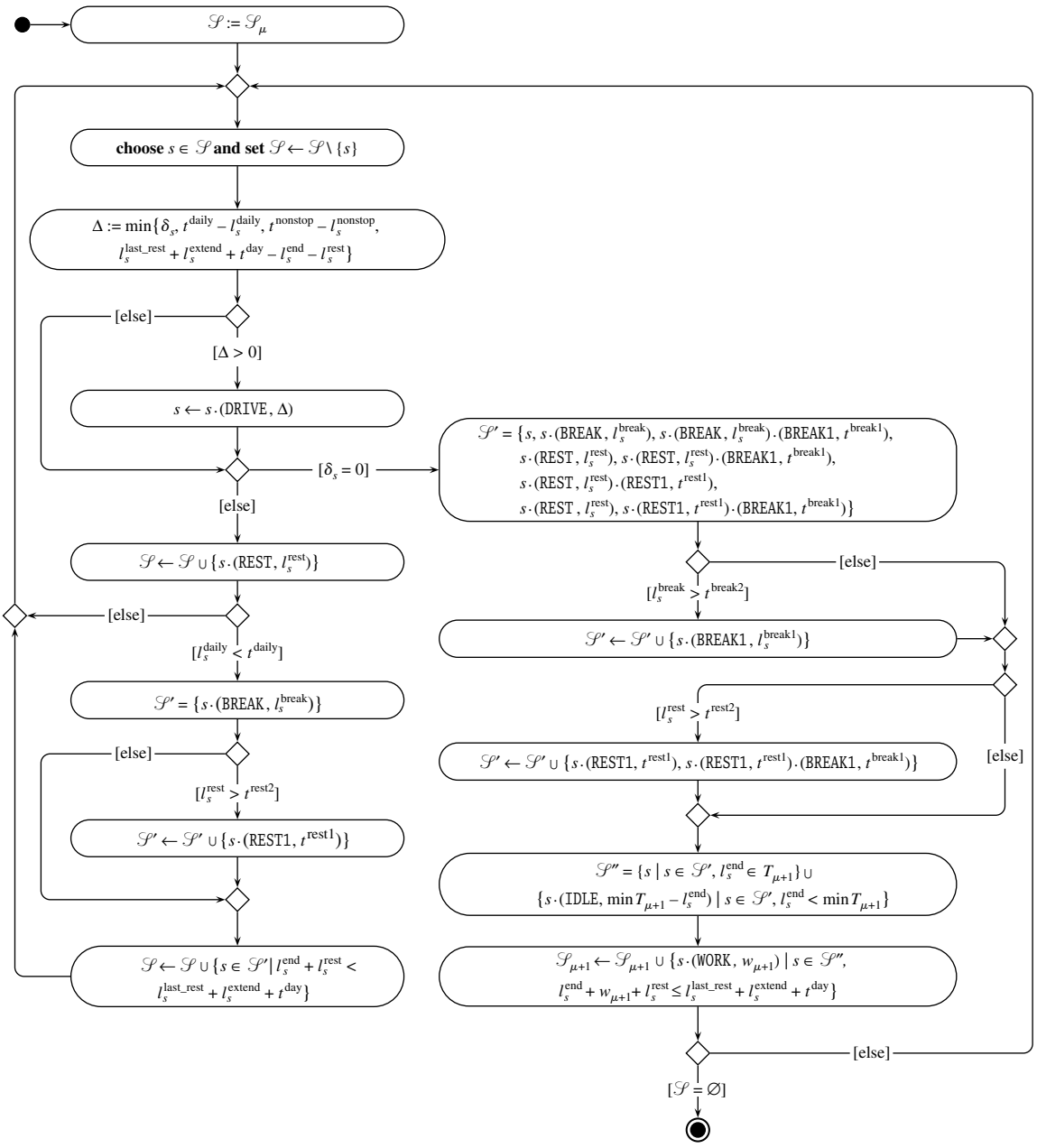


Figure 4 Scheduling Method

for each partial schedule s the remaining driving time required to reach the next location $n_{\mu+1}$.

Before describing the scheduling method in detail, let us make the following observation. Because of the normality conditions (N1), (N12), and (N14) we know that activities of type BREAK1 or IDLE must only be scheduled after arrival at the next location; i.e., they must only be scheduled if $\delta_s = 0$. Thus, if $\delta_s > 0$, we must only schedule activities of type DRIVE, BREAK, REST1, or REST until the next location is reached. Because of the normality conditions (N1), (N8), (N9), and (N10) we know that an activity of type BREAK, REST1, or REST can only be succeeded by another activity of type BREAK, REST1, or REST if the first activity is of type REST and the second is of type REST1. This activity of type REST1 must be succeeded by an activity of type DRIVE. This, however, violates condition (N15). Thus, we can make the observation that if $\delta_s > 0$ and the last activity of s is of type BREAK, REST1, or REST, then the next activity must be of type DRIVE.

The scheduling method starts by choosing a partial schedule $s \in \mathcal{S}$. In the beginning of the method $\mathcal{S}$ only contains schedules in $\mathcal{S}_\mu$. Thus, the last activity of the chosen schedule is of type WORK. Now, the method determines the minimum of the remaining driving time to the next location, the available time of driving

until the maximum daily driving time is reached, the available time of driving until the maximum nonstop driving time is reached, and the maximum time until a new rest period must be scheduled. If this value, which is denoted by Δ, is zero, then no driving activity must be scheduled. Otherwise, $\Delta > 0$. Assume that we would continue by scheduling an activity of type BREAK, REST1, or REST. Then this activity would have to be succeeded by a driving period. This, however, would violate condition (N15) or (N16). Thus, the scheduling method must continue by scheduling an activity of type DRIVE. The duration of this driving period must be Δ, because any smaller value would result in the need for another driving period which would have to be scheduled after a break or rest period. This, again, would violate condition (N15) or (N16).

If $\delta_s = 0$ at the second junction in the figure, the last activity of s must be the last driving activity before reaching location $n_{\mu+1}$. The method creates a copy of the schedule for every combination of break, rest, and idle periods that can be appended to the schedule s without violating conditions (N1) to (N14). The work activity is appended to each of these copies, and each schedule satisfying condition (11′) is included in the set $\mathcal{S}_{\mu+1}$.

If $\delta_s > 0$ at the second junction in the figure, an activity of type BREAK, REST1, or REST must be scheduled before the next driving period. The method adds a copy of the schedule, appends an activity of type REST, and includes it in the set $\mathcal{S}$. If $l_s^{\text{daily}} = t^{\text{daily}}$, this is the only schedule that must be considered. Otherwise, the method creates another copy of the schedule and appends an activity of type BREAK. If $l_s^{\text{rest}} > t^{\text{rest2}}$, the method creates a third copy of the schedule and appends an activity of type REST1. However, these two copies are only included in the set $\mathcal{S}$ if $l_s^{\text{end}} + l_s^{\text{rest}} < l_s^{\text{last_rest}} + l_s^{\text{extend}} + t^{\text{day}}$, because otherwise, an activity of type REST would have to follow the activity of type BREAK or REST1. This, however, would violate condition (N8) or (N10).

As long as $\mathcal{S} \neq \varnothing$, the method continues by choosing the next partial schedule in the set. The last activity of this schedule may now be of type WORK, BREAK, REST1, or REST. The algorithm continues with the same steps as described above to find all pseudofeasible schedules in normal form for $\mu+1$ that can be determined when starting with the set $\mathcal{S}_\mu$.

If we start with $\mathcal{S}_1$ and $\mu = 1$, we can solve the EU-TDSP by repeatedly invoking the scheduling method and increasing μ until $\mu = \lambda$ or $\mathcal{S}_\mu = \varnothing$. Without removing dominated schedules from $\mathcal{S}_\mu$, the number of partial schedules generated by such an approach may grow exponentially. Luckily, we can remove all dominated schedules from $\mathcal{S}_\mu$ each time before we invoke the scheduling method. Although the number of nondominated partial schedules may remain very large, we can solve the EU-TDSP without constructing unnecessarily many partial schedules.

7. Computational Experiments

Scheduling driving and working hours is of particular importance for long distance haulage where drivers do not return home every day. To evaluate the scheduling method presented in this paper we generate benchmark instances for a planning horizon starting on Monday morning and ending on Friday evening. In the benchmark set each handling activity requires one hour of working time (i.e., $w_\mu = 1$ for all $1 \le \mu \le \lambda$) and the driving time between two subsequent locations is 4, 8, 12, or 16 hours (i.e., $\delta_{\mu,\mu+1} \in \{4, 8, 12, 16\}$ for all $1 \le \mu < \lambda$). Assuming an average speed of 75 km/h, this implies that the distance between two subsequent locations ranges from 300 km to 1,200 km. Each location must be visited on a specific day between 6.00 h and 13.59 h or between 14.00 h and 21.59 h.

In total, around 15.6 million instances were generated in which time window constraints could be satisfied if driving and working times were unrestricted. Only 662,308 of these instances do not exceed the accumulated weekly driving and working times of 56 and 60 hours, respectively.

We compare various variations of the scheduling method presented in this paper with the naïve and multilabel method presented by Goel (2009) and the method presented by Kok et al. (2009). The method presented in the previous section can be interpreted as a breadth-first search (BFS) algorithm. A variation of the BFS method can be obtained if we do not schedule any activities of type BREAK1, i.e., if no split breaks are scheduled. We denote this variation by NSB-BFS. Three more variations of the NSB-BFS method have been implemented which perform an incomplete search; i.e., only the most promising partial schedules in the set $\mathcal{S}_\mu$ are considered when invoking the scheduling method. For this, the schedules $s \in \mathcal{S}_\mu$ are ordered in a lexicographical way based on the schedule characteristics

$$\big(l_s^{\text{end}}, -\min\{t^{\text{daily}} - l_s^{\text{daily}},\ t^{\text{nonstop}} - l_s^{\text{nonstop}},\ l_s^{\text{last_rest}} + l_s^{\text{extend}} + t^{\text{day}} - l_s^{\text{end}} - l_s^{\text{rest}}\},\ l_s^{\text{rest}}\big).$$

That is, the schedules are ordered based on their completion time, the maximum duration of the next driving period, and the amount required to complete a rest period. We denote the methods in which only the first three, four, or five schedules in $\mathcal{S}_\mu$ are considered by NSB-IBFS3, NSB-IBFS4, and NSB-IBFS5.

Table 2 gives an overview of the number of instances for which a feasible schedule is found by

Table 2 Results

Algorithm	Instances	Time	$\lambda = 3$	$\lambda = 4$	$\lambda = 5$	$\lambda = 6$	$\lambda = 7$	$\lambda = 8$	$\lambda = 9$	$\lambda = 10$	$\lambda = 11$	$\lambda = 12$
BFS	43,882	160'17"	100	100	100	100	100	100	100	100	100	100
NSB-BFS	43,797	69'01"	100	100	100	99.80	99.93	99.65	99.92	99.69	99.89	98.48
NSB-IBFS5	40,610	53'34"	100	100	99.21	96.76	93.96	91.84	90.68	89.56	92.22	90.91
NSB-IBFS4	40,022	49'25"	100	100	98.52	95.81	92.86	90.90	89.17	88.16	90.10	92.31
NSB-IBFS3	39,024	46'06"	100	100	98.52	95.28	91.70	88.56	85.93	83.00	81.44	77.27
Multilabel	37,484	17'23"	100	100	95.85	92.33	88.57	86.00	82.90	76.82	65.11	45.45
Kok et al. (2009)	34,520	16'34"	100	100	95.45	91.30	87.06	81.10	71.88	58.20	40.78	15.15
Naïve	33,952	15'35"	100	100	94.27	87.54	81.98	77.34	72.84	66.69	57.22	36.36

Note. The number of instances for which a feasible schedule is found by the method, total computation time required, and percentage of instances for which a feasible schedule is found is shown above.

the method and total computation time required. Furthermore, Table 2 shows the percentage of instances with a given length λ, for which the approach finds a feasible schedule with respect to the number of instances for which a feasible schedule exists. As the BFS method finds a feasible schedule for a given instance if such a schedule exists, we can conclude that for 43,882 instances a feasible schedule exists, and for the other 618,426 instances which do not exceed the accumulated weekly driving and working time, no feasible solution exists for the EU-TDSP. We can see that the BFS method finds a feasible schedule for significantly more instances than the methods presented by Goel (2009) and Kok et al. (2009). However, the computational effort to solve the EU-TDSP is much higher than the effort required by the simpler methods. This increased computational effort results from the relatively large number of partial schedules that are considered by the BFS algorithm. In our experiments, up to 230 different nondominated partial schedules were considered by the BFS algorithm for a single instance. The NSB-BFS method which does not schedule any activity of type BREAK1 requires significantly lower computation time and only fails to find a feasible schedule in 85 cases. We can conclude that taking break periods in two parts does not generate a big benefit for long distance haulage. NSB-IFBS3, NSB-IFBS4, or NSB-IFBS5 can be used to reduce computation time at the cost of not finding a feasible schedule for some of the instances.

It must be noted that we can also use a depth-first search (DFS) variant of the algorithm outlined above. In each iteration, this DFS version would use exactly one of the partial schedules in $\mathcal{S}_\mu$ to generate feasible partial schedules for the tour $n_1, n_2, \ldots, n_{\mu+1}$. Such a DFS algorithm would have the advantage that it would be able to find a feasible schedule for some of the "easy" instances in linear time and, thus, just as quickly as the fast approaches presented in previous papers. However, the DFS version has the disadvantage that it may consider some partial schedules that are dominated and removed in the BFS version. Thus, for "harder" instances, a higher computational effort may arise for the DFS approach. In computational experiments we were able to observe this disadvantage and saw that the DFS approach actually requires slightly more time than the BFS version.

In Table 2 we can see that methods presented in previous papers succeed in finding a feasible schedule for a very small percentage of instances with larger numbers of locations in the tour. To increase profitability, carriers seek to increase the number of tasks to be performed by each vehicle. For those tours that are likely to be among the most profitable ones (i.e., those having the highest number of locations) the methods presented in this paper bring the highest benefit compared to the simpler—yet faster—methods presented by Goel (2009) and Kok et al. (2009).

8. Conclusions

Scheduling working and driving hours of truck drivers is a highly complex task in the presence of legislatory constraints. Despite their importance, restrictions on drivers' working hours have attracted very little interest in the literature. This paper presents a BFS method and several variations for scheduling driving and working hours of European truck drivers. Unlike previous approaches, the BFS approach presented in this paper is guaranteed to find a feasible truck driver schedule, if such a schedule exists. Computational experiments demonstrate that previous approaches succeed in finding a feasible schedules for less than a half of the instances with the highest number of locations to be visited. The experiments furthermore demonstrate that taking break periods in two parts, i.e., a first part of at least 15 minutes and a second part of at least 30 minutes, does not bring a big benefit for long distance haulage. The NSB-BFS method, which does not consider the possibility of taking break periods in two parts, can be used to significantly reduce the computational effort, compared to the BFS. Further computational gains can be achieved by performing an incomplete search. The NSB-IFBS3, NSB-IFBS4, and NSB-IFBS5 approaches only consider the most promising three, four, or five

partial schedules in each iteration. However, NSB-IFBS4, and NSB-IFBS5 still find a feasible schedule for about 90% of all instances generated.

All methods can be embedded into local search-based optimisation methods for combined vehicle routing and truck driver scheduling. Whenever a new tour for a vehicle is obtained by some neighbourhood move, the methods can be used to determine whether the modified tour satisfies time window constraints and regulation (EC) No. 561/2006. Depending on the amount of computation time available, the BFS, one of its variants, or the previous methods presented by Goel (2009) or Kok et al. (2009) can be applied. If schedules are prepared on a weekly basis, the entire weekend may be available for optimisation. The significant better performance in terms of number of instances for which a feasible schedule can be found by one of the methods presented in this paper, should be worth the additional computational effort.

References

Arabeyre, J. P., J. Fearnley, F. C. Steiger, W. Teather. 1969. The airline crew scheduling problem: A survey. *Transportation Sci.* **3**(2) 140–163.

Archetti, C., M. W. P. Savelsbergh. 2009. The trip scheduling problem. *Transportation Sci.* **43**(4) 417–431.

Cordeau, J.-F., G. Stojkovic, F. Soumis, J. Desrosiers. 2001. Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Sci.* **35**(4) 375–388.

Ernst, A. T., H. Jiang, M. Krishnamoorthy, B. Owens, D. Sier. 2004. An annotated bibliography of personnel scheduling and rostering. *Ann. Oper. Res.* **127**(1) 21–144.

European Commission. 2003. Europe at a crossroads—The need for sustainable transport. Office for Official Publications of the European Communities, Luxembourg.

European Transport Safety Council. 2001. The role of driver fatigue in commercial road transport crashes. ETSC Review, www.etsc.eu.

European Union. 2006. Regulation (EC) No. 561/2006 of the European Parliament and of the Council of 15 March 2006 on the harmonisation of certain social legislation relating to road transport and amending Council Regulations (EEC) No. 3821/85 and (EC) No. 2135/98 and repealing Council Regulation (EEC) No. 3820/85. Official Journal of the European Union L 102, 11.04.2006, Luxembourg.

Freling, R., D. Huisman, A. P. M. Wagelmans. 2003. Models and algorithms for integration of vehicle and crew scheduling. *J. Scheduling* **6**(1) 63–85.

Goel, A. 2009. Vehicle scheduling and routing with drivers' working hours. *Transportation Sci.* **43**(1) 17–26.

Haase, K., G. Desaulniers, J. Desrosiers. 2001. Simultaneous vehicle and crew scheduling in urban mass transit systems. *Transportation Sci.* **35**(3) 286–303.

Huisman, D., R. Freling, A. P. M. Wagelmans. 2005. Multiple-depot integrated vehicle and crew scheduling. *Transportation Sci.* **39**(4) 491–502.

Kohl, N., S. E. Karisch. 2004. Airline crew rostering: Problem types, modeling, and optimization. *Ann. Oper. Res.* **127** 223–257.

Kok, A. L., C. M. Meyer, H. Kopfer, J. M. J. Schutten. 2009. A dynamic programming heuristic for the vehicle routing problem with time windows and European Community social legislation. *Transportation Sci.* **44**(4) 442–454.

Laurent, B., J.-K. Hao. 2007. Simulated annealing, simultaneous vehicle and driver scheduling: A case study in a limousine rental company. *Comput. Indust. Engrg.* **53**(3) 542–558.

Mercier, A., J.-F. Cordeau, F. Soumis. 2005. A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem. *Comput. Oper. Res.* **32**(6) 1451–1476.

Powell, W. B., A. Marar, J. Gelfand, S. Bowers. 2002. Implementing real-time optimization models: A case application from the motor carrier industry. *Oper. Res.* **50**(4) 571–581.

Sandhu, R., D. Klabjan. 2007. Integrated airline fleeting and crew-pairing decisions. *Oper. Res.* **55**(3) 439–456.

Savelsbergh, M. W. P., M. Sol. 1998. DRIVE: Dynamic routing of independent vehicles. *Oper. Res.* **46** 474–490.

Valouxis, C., E. Housos. 2002. Combined bus and driver scheduling. *Comput. Oper. Res.* **29**(3) 243–259.

Xu, H., Z.-L. Chen, S. Rajagopal, S. Arunapuram. 2003. Solving a practical pickup and delivery problem. *Transportation Sci.* **37**(3) 347–364.