Scheduling Locomotives and Car Transfers in Freight Transport

Armin Fügenschuh[1], Henning Homfeld[1],
Andreas Huck[2], Alexander Martin[1], and Zhi Yuan[1]

[1] Technische Universität Darmstadt, Arbeitsgruppe Optimierung
64289, Darmstadt, Schlossgartenstr. 7, Germany
{fuegenschuh,homfeld,martin,yuan}@mathematik.tu-darmstadt.de
[2] Deutsche Bahn AG, Konzernentwicklung, GSU 1
60326, Frankfurt, Stephensonstraße 1, Germany
andreas.huck@bahn.de

Abstract. We present a new model for a strategic locomotive scheduling problem arising at the Deutsche Bahn AG. The model is based on a multi-commodity min-cost flow formulation that is also used for public bus scheduling problems. However, several new aspects have to be additionally taken into account, such as cyclic departures of the trains, time windows on starting and arrival times, network-load dependent travel times, and a transfer of cars between trains. The model is formulated as an integer linear programming problem (ILP). The formulation is improved by preprocessing and additional cutting planes. Solutions are obtained using a randomized greedy heuristic in combination with commercial ILP solvers. Computational results are presented for several real-world test instances.

Keywords. Railroad Freight Transport, Cyclic Vehicle Scheduling, Time Windows, Integer Linear Programming, Heuristics, Cutting Planes.

1 Introduction

Deutsche Bahn AG (DB) is the largest German railway company with 216,000 employees and a turnover of 25 billion Euros in 2005. DB is active in both passenger and freight transportation. Per year, 1.8 billion passengers (72 billion passenger kilometers) and 253 million tons of goods (77 billion ton kilometers) are transported. Moreover, DB is the owner of the German railway system, where DB freight and passenger trains travel 887 million kilometers per year and external railway companies around 110 million kilometers. The overall length of the railways is 34,000 kilometers, about 4,400 freight trains and 30,000 passenger trains per day traverse this network [1,2]. All in all, DB's network is considered as one of the most dense and most frequently used railway networks in the world.

For long-term simulations and future predictions of the network load, DB developed a complex simulation tool. The entire simulation tool may be considered as a chain, which decomposes into several components. To this end it is

possible to model the normal course of business operations and to analyze the influence of changing external (such as future demands on goods or the sensitivity to the price for oil) as well as internal parameters (such as possible network expansions or changes in strategy). The components of the tool chain interact by generating output data, which in turn serves as input for further parts of the chain. The entire simulation tool has evolved over the last 5 years and is still under continuous improvement. In this article we describe the development of a new segment for the tool chain.

Currently, a chain's segment called *train scheduler* is responsible for the buildup of trains from individual cars. Concerning time, the train is started as soon as enough cars are assembled. That means, the starting times of the trains are not geared to some timetable, but follow the estimated customers' production and demand peaks in the simulation. In this process a locomotive is supposed to be always and immediately available for pulling this train, that is, there is no scheduling of the locomotives. Since they are among the most expensive resources of operation of any railroad company, their efficient deployment is of high priority. Our focus is on the effective scheduling of locomotives.

The remainder of this paper is organized as follows. In Section 2 we start with a description of the problem in greater detail. We continue in Section 3 with a survey of the available locomotive scheduling models from literature. In Section 4 we provide a stepwise refined model, formulated as an integer linear programming problem. We address primal and dual aspects in finding feasible solutions and lower bounds in Section 5. In Section 6 we present computational results for the different variants of our model. An outlook to further work is finally given in Section 7.

2 The Problem Settings

In this section we give details of the integrated scheduling problem and introduce the terminology used at DB.

2.1 Cars

A *car* is a rolling stock for freight transport. The cars have to be delivered between a source and a destination point (goods station) within the railway network. Major customers produce and/or consume very large amounts of goods allowing to order complete trains, also called block trains. In these cases, the cars are transported in one single train.

Smaller customers order individual cars. These cars are assembled into trains along with cars ordered by other small customers and pulled together to an intermediate destination (a shunting yard), where the trains are split up and reassembled into new trains. Observe that one car may travel in more than one train from its source to its destination. The assembling of cars into trains, the so called *blocking problem*, which determines the individual schedule for each car, is a difficult combinatorial optimization problem on its own (see, for instance,

Ahuja et al. [3]). In our model we assume that a solution to this problem is given. Hence, the trains and the yards where the cars transfer between trains are known in advance. When changing the starting times of the trains, one must ensure that these transfers still remain feasible.

2.2 Trains and Trips

A *freight train* consists of several *cars*. The train – together with its start and destination as well as its starting and arrival times – is called a *trip* and represents the transport activity to be scheduled. These trips are also called *active trips* to clearly distinguish them from deadhead trips, see Section 2.4). The starting and arrival times of the trips are either fixed or vary within given intervals. The trips are supposed to be cyclic every 24 hours. The *trip duration* is the time difference between start and arrival. The average travel speed of freight trains is not as high as in passenger transport, especially at daytime, when passenger trains have priority causing some trips to last up to three days (see Section 4.1 for details). The trains have different lengths and weights and thus require locomotives with sufficient driving power. Mostly a train is pulled by a single locomotive. Only a minority of the heaviest trains need a double traction (two pulling locomotives). At the start a locomotive is attached to the train and at the destination station it is detached (uncoupled). For both *coupling* processes, a certain trip-dependent amount of time has to be taken into account (15 to 30 minutes) to carry out technical checks and refueling of diesel locomotives.

2.3 Locomotives

DB uses up to 30 different locomotive types from several manufacturers. However, the differences between them are often minor allowing to group them into three to six *classes*. The main differences among the classes are power and tractive force of the engines and the *traction system* (i.e., the motor type, diesel or electrical). Electric powered locomotives require electrified tracks, whereas diesel powered locomotives (with some exceptions) can be used on the complete rail network. It follows from these technical restrictions that it is only possible to assign such locomotives to trains that have sufficient power to pull the train and the right traction system for the track.

2.4 Deadheads

A locomotive is either active, i.e., pulling a train, or deadheading, i.e., driving without pulling a train from the destination station of one train to the departing station of another train. There is a third state called *light deadheading* or *passive deadheading* where the locomotive is put in a train just as a car. The costs for passive deadheading are lower than for active deadheading, since crew and fuel costs are saved. However, we decided to not include this state in our model. This is mainly due to the fact that the current amount of light deadheads at DB

is small in comparision to the active deadheads. On the other hand the model formulation would be unreasonably more complicated.

The duration for a deadhead trip depends rather on the distance between these two points and on the class of the locomotive (diesel and electrical might have to use different routes) than on the network load (i.e., independent of daytime or nighttime), because it is assumed that a single locomotive can always be pushed through.

2.5 Goals and Objectives

The main goal is to generate feasible *schedules* of the locomotives, which are cyclic sequences of trips and deadhead trips. The main objective is to reduce operating expenses, that is, to use as few locomotives (ties are broken by the cost of their type) as possible to pull all trains. On a subordinate level the locomotives are supposed to be scheduled in such a way that the deadhead trips are as short as possible.

In Section 4 we present a hierarchy of models with increasing complexity. On the lowest complexity level, the first model will have exactly the above mentioned goal and objectives. All other more evolved models allow an alternation of the starting times of the trips. Hence, the assignment of proper starting times to the trips is an additional goal. We will see that these starting times affect the number of deployed locomotives. Due to this additional degree of freedom a reduction compared to the case of fixed starting times may be expected. In addition, the provided quality of service, namely the time a car needs from its origin to its final destination, must be taken into account. Too long delivery times will finally result in a negative market response (fewer customers for DB). So a further objective is the reduction of these travel times, which obviously is in conflict with the objective of reducing the locomotive fleet.

3 Survey of the Literature

For a general survey of combinatorial problems arising in railroad operating systems we refer to the articles of Bussieck, Winter, and Zimmermann [4], Caprara, Fischetti, Toth, and Vigo [5], and Cordeau, Toth, and Vigo [6]. In the sequel we will focus on articles that are particularly devoted to locomotive scheduling problems.

In 1980, Booler [7] presented a heuristic for solving a locomotive scheduling problem with variable starting times of the trips. For each trip an interval can be specified in which the start has to take place. A multi-commodity flow formulation is used to handle different types of locomotives. The selection of proper starting times from the interval is based on the dual variables from the flow model. The assignment of locomotives to jobs is done by a greedy cheapest-available rule. Each of the trips (called jobs) have to occur exactly once in the schedule and there is no cyclic structure: All locomotives start and end their

schedule at depots. The goal is to reduce the operating cost whereas the number of locomotives per type is fixed.

In 1989, Smith and Sheffi [8] gave a nonlinear multi-commodity flow formulation on a time-expanded network graph (time-space network) for a locomotive scheduling problem with uncertain demand. In this graph, nodes correspond to railway stations together with a certain point in time. Hence for one station there are multiple time-dependent copies in the graph. The authors consider a finite planning horizon. The uncertainty is due to the unknown number of cars within a train, which leads to an uncertain strength requirement to pull these trains. Smith and Sheffi present a heuristic algorihm to solve the problem. The solutions are compared with a lower bound.

The relationship between single- and multi-depot bus scheduling and locomotive scheduling was perhaps first noted in 1991 by Forbes, Holt, and Watts [9]. They gave a multi-commodity flow formulation which also includes the heterogeneous locomotive fleet as a limiting factor. The trips are cyclic on a daily base. Trips longer than 24 hours are also possible. The number of locomotives needed for the connection of two trips is the number of midnight passes, similar to formula (4) to be explained below in this article. Locomotives are allowed to carry out deadhead trips between stations (deadhead trips are called light runs). The starting times of the trips are given. As a solution technique, the authors propose a linear programming based branch and bound method, and solve instances with up to 200 trips and 5 different types of locomotives to optimality within 21,000 sec. on a – at that time modern – 25MHz Intel-486 platform. It is remarked that the LP-relaxation of the model has a very small integrality gap.

In 1999, Ziarati et al. [10] present a time-space-network approach for a locomotive scheduling problem with a heterogeneous fleet. The locomotives are able to perform active and passive deadheading. Maintenance intervals of the locomotives are also scheduled. The trains are non-cyclic with given fixed start and ending times. The goal is to provide suitable horsepower for each of the trains, which can come from one or more locomotives. The problem is formulated as a mixed-integer linear program. The initial formulation is strengthened by problem-specific cutting planes and then solved via Dantzig-Wolfe decomposition. They apply their approach to real-world data provided by a Canadian railway company, and found near-optimal solutions for instances with up to 2,000 trains and 26 different types of locomotives, for which they need around 1,000 locomotives.

Ziarati, Chizari, and Nezhad [11] propose a multi-commodity flow formulation for a cyclic heterogeneous locomotive scheduling problem. The problem is solved by a heuristic genetic algorithm. However, no information on dual bounds is provided. The data instance is from a real company and consists of up to 1,700 trips and 1,200 available locomotives, which could be solved after 20 hours computation time on a 1GHz Pentium-III platform.

Ahuja et al. [12] present a time-space network model for the locomotive scheduling problem with fixed starting times. The corresponding mixed-integer linear program is too large to be solved directly with an ILP solver. Hence they

decompose the problem into smaller parts that are solved using a very large scale neighborhood search heuristic. Mixed-integer linear programs also occur as subproblems and are solved using Ilog Cplex 7. Solutions are compared to lower bounds, if available. A weekly schedule is computed in a day-by-day fashion. Deadhead trips (light travels) of the locomotives are permitted. Locomotives can be assembled to trains so that only a few of them are actively pulling, which is of course cheaper than individual deadhead trips per locomotive. The trains can also be pulled by more than one locomotive. As an additional cost term, the so called consist busting is considered, which is the cost for breaking up sets of locomotives. The largest real-world scenario solved consists of up to 3,300 trips and locomotives of five different types from a pool of 3,300.

Recently Powell and Bouzaiene-Ayari [13], [14] suggested an approximative dynamic programming approach. They are able to include several more real-world requirements that were not addressed by other approaches before. Among them are shop routing (a locomotive is sent to a shop for maintenance), or certain operation patterns. To handle these additional difficulties they suggest a heuristic procedure based on dynamic programming recursion. In order to avoid the full enumeration, they carry out a limited search between two adjacent time steps only.

Our Contribution

The scope of our research is to formulate a locomotive scheduling model that is adapted to the needs of DB's freight business unit. Similar to some of the models in literature, we also deal with a heterogeneous fleet of locomotives, cyclic departures of the trains, and time windows on starting and arrival times. Our model is formulated as an integer linear programming problem (ILP, for short), based on a multi-commodity flow formulation, similar to [7], [9], or [11]. In particular we do not use time-expanded graphs. Although they seem to be appropriate for problems with fixed starting times, but are too inflexible when including starting time windows.

However, two additional aspects have to be taken into account which are not part of the approaches presented above. First, the travel times of the trips are network-load dependent. That means, when we change the starting time of a trip the overall trip duration has to be adapted as well. And second, some of the cars transfer between trains. Hence the starting times of these trips have to be linked so that the transfer is possible whatever starting times are chosen. These are perhaps the most important constraints in our model, since a missed transfer means a delay in the handling of the car and thus customer dissatisfaction.

We will see that standard ILP solvers like ILOG Cplex [15] used as black-box will reach their limits very soon and are not able to solve instances of realistic size. We extend the solver's capability by adding problem specific presolving techniques, developing a primal heuristic. And to improve the dual bound, we add valid cutting planes to the model. Both, the heuristic and the cuts, are integrated in the branch and bound framework of the ILP solver. Numerical results on various test instances show that our extensions outperform the simple

black-box use of the solving software and are altogether able to solve instances of realistic size.

It was already noted by Forbes, Holt, and Watts [9] that there are several similarities between bus and locomotive scheduling problems. Also our approach has some ties to similar problems in the area of public bus transport studied before. It was noted by Daduna and Völker [16] that a significant amount of vehicles can be saved if the starting times of the bus trips are altered within some small interval. Later, Fügenschuh [17] also included the customers' demands into the optimization such that the scheduling problem of the vehicles can be solved together with the starting time problem of the trips with respect to further customers' demand constraints, which leads to further reductions of vehicles and thus costs. When changing the starting times of the trips, it must be ensured that the customers who transfer between trips are able to catch their desired bus, similar to the cars in our problem. In this respect the work presented in this paper can be seen as an approach to carry over the results in [17] from the area of public bus transport to railroad freight transportation systems.

4 Models

The models we describe in this section can be classified by one main characteristic, the starting time intervals. In the first variant the starting time is given by the pre-scheduler, one module of the tool chain. According to this module, a trip is started as soon as enough cars are assembled. What "enough" in this context means is guided by a local criterion, which is mainly based on the quantity, total weight, and total length of the cars. However, this local criterion does not take into account the availability of locomotives. It is assumed that a locomotive is available for any trip at any time. A scheduling model that assures this availibility under the assumption of fixed starting times is presented in Section 4.1.

On the other hand, there is always a little flexibility in the departure and arrival of the trips, which has to be negotiated with the customers. In Section 4.2 we describe a model where the starting time can vary within a given interval. This model is much more complex, since it has to take care of the synchronization of the trip departures and the car schedules. A further refinement of this model is given in Section 4.3. Here the trip durations are not constant, but vary dynamically depending on the actual network load.

For all models we use the following notation. Let $\mathcal{V}$ be the set of active trips and let $\mathcal{B}$ be the set of locomotive classes. We introduce a parameter $a_{b,i} \in \{0,1\}$ with $a_{b,i} = 0$ if a class b locomotive cannot serve trip i (e.g. if it is too weak to pull the train). Let $\mathcal{A} := \mathcal{V} \times \mathcal{V}$ denote the set of all potential connections of pairs of trips. The elements of $\mathcal{A}$ are called deadhead trips in general, even if the first trip's destination is equal to the start location of the second. Denote $a_{b,(i,j)} := a_{b,i} \cdot a_{b,j} \in \{0,1\}$, then we have $a_{b,(i,j)} = 0$ if and only if the deadhead trip from i to j is not feasible for a class b locomotive. Moreover, those $a_{b,(i,j)}$ can be set to zero where the corresponding deadhead trip exceeds a certain length.

4.1 Fixed Starting Times

We take the starting and arrival times for the trips as they were computed by the train scheduler of the tool chain. This computation also ensures that individual cars can transfer between trains. We introduce a decision variable $x_{b,(i,j)} \in \{0, a_{b,(i,j)}\}$ with $x_{b,(i,j)} = 1$ if trips i and j are connected and both served with a locomotive of class b, and $x_{b,(i,j)} = 0$ otherwise. In this context "connected" means that trip j is served following trip i by a locomotive of the same type. Adding a deadhead between the destination of trip i and the starting point of trip j might be necessary. Each trip j must be served with one class b of locomotive, that is,

$$\sum_{b\in\mathcal{B}} \sum_{i:(i,j)\in\mathcal{A}} x_{b,(i,j)} = 1. \tag{1}$$

There is a flow conservation in the sense that two connected trips can only be served by the same type of locomotive $b \in \mathcal{B}$:

$$\sum_{i:(i,j)\in\mathcal{A}} x_{b,(i,j)} = \sum_{k:(j,k)\in\mathcal{A}} x_{b,(j,k)}. \tag{2}$$

Equations (1) and (2) imply that each trip j has a unique predecessor trip i and a unique successor trip k. Hence, a feasible solution to these equations can be seen as a partitioning of the digraph $(\mathcal{V}, \mathcal{A})$ into disjoint cycles. Note that in this article two different notions of cycles are in use. The first one is that of a *cyclic trip*, which means that the trips occur every day. The other notion introduced here is that of a *cycle* which combines several – cyclic – trips in one schedule. One type of locomotive is assigned to each cycle. The actual number of locomotives of this particular type per cycle is determined as follows.

We denote by $\lambda_{b,(i,j)}$ the number of locomotives of class b that is contributed to the total number of locomotives in the cycle due to a connection of i with j. This number equals the number of midnight passes between the end time of trip i and the beginning of trip j, including possible deadheading from i to j, as well as coupling and decoupling times. Forbes, Holt, and Watts [9] (see also Liebchen and Möhring [18]) show that this number is the smallest integer such that

$$\hat{t}_j + 1440 \cdot \lambda_{b,(i,j)} \geq \hat{t}_i + \delta_{b,(i,j)}, \tag{3}$$

which can be computed as

$$\lambda_{b,(i,j)} := \left\lceil \frac{\hat{t}_i + \delta_{b,(i,j)} - \hat{t}_j}{1440} \right\rceil \geq 0, \tag{4}$$

where $\hat{t}_i, \hat{t}_j$ are the pre-scheduled starting times of trips i and j, respectively. The constant 1440 refers to the number of minutes per day, which is the basis of the cycles (i.e., all trips are repeated on a daily base) and $\delta_{b,(i,j)}$ denotes the total trip and deadhead trip duration, that is,

$$\delta_{b,(i,j)} := \delta_i^{\text{trp}} + \delta_i^{\text{uncpl}} + \delta_{b,(i,j)}^{\text{dhd}} + \delta_j^{\text{cpl}}, \tag{5}$$

where

- δ_i^{trp} denotes the trip duration, i.e., the time the locomotive is active while serving trip i,
- δ_i^{uncpl} denotes the time for uncoupling the locomotive from the train at the arrival,
- $\delta_{b,(i,j)}^{\text{dhd}}$ denotes the time for deadheading from the end of trip i to the start of trip j, and
- δ_j^{cpl} denotes the time for coupling the locomotive to the train at the start of trip j.

The driving time δ_i^{trp} is assumed to be independent of the actual class, whereas the deadhead time $\delta_{b,(i,j)}^{\text{dhd}}$ is class dependent (since diesel and electrical might use different routes). For certain other constraints of the model (see (18) below) it is necessary to know (or to define) upper bounds on these parameters: The longest trip duration δ_i^{trp} is assumed to be three days. Though this is an overestimation for most instances, it is driven by the need to handle international traffic as well. A typical value for the uncoupling δ_i^{uncpl} is 15 minutes, whereas the coupling δ_j^{cpl} can take up to 45 minutes, due to an additional security test of the brakes. For the deadheads $\delta_{b,(i,j)}^{\text{dhd}}$ we have a limit of 24 hours, which is sufficient even for the largest deadhead trips. Altogether, $\delta_{b,(i,j)}$ is at most 5820 minutes.

A *capacity* in form of an upper bound B_b on the number of available locomotives of class b can be specified:

$$\sum_{(i,j)\in\mathcal{A}} \lambda_{b,(i,j)} x_{b,(i,j)} \leq B_b. \tag{6}$$

As objective function we minimize the total costs defined as

$$\sum_{b\in\mathcal{B}} \sum_{(i,j)\in\mathcal{A}} (\gamma_b^{\text{cls}} \lambda_{b,(i,j)} + \gamma_{b,(i,j)}^{\text{dhd}}) x_{b,(i,j)} \tag{7}$$

where

- γ_b^{cls} denotes the costs for a locomotive of class b,
- $\gamma_{b,(i,j)}^{\text{dhd}}$ denotes the costs in connection with deadheading from i to j with locomotive b.

The most important objective is the reduction of the deployed locomotives followed by the reduction of deadhead costs. Hence $\gamma_b^{\text{cls}} \gg \gamma_{b,(i,j)}^{\text{dhd}}$.

The optimization problem is the minimization of (7) subject to the constraints (1), (2), (6), and the integrality of all $x_{b,(i,j)}$. Note that we do not have a depot where the schedules of the locomotives start and end. The predecessor resp. successor of any trip is always some other trip. Due to the cyclic character of the schedules of the locomotives we call this problem the *capacitated cyclic vehicle scheduling problem* (CVSP). In the case of no upper bounds (or $B_b = \infty$) we also speak of the *uncapacitated* CVSP.

For $|\mathcal{B}| = 1$, this problem reduces to a single-commodity minimum-cost flow problem, which can be solved efficiently by polynomial or pseudopolynomial

algorithms (for instance by the Hungarian Method, see Ahuja, Magnanti, and Orlin [19] for details). For $|\mathcal{B}| > 1$, the uncapacitated CVSP is a multi-commodity min-cost flow problem, which is known to be NP-hard. Moreover, it is NP-complete to decide whether a feasible solution exists for the capacitated CVSP (see Löbel [20]).

4.2 Variable Starting Times

We change the above model for locomotive scheduling for the case where the starting times of the trips are allowed to be changed within a given interval. The corresponding model is called (capacitated or uncapacitated) *cyclic vehicle scheduling problem with time windows* (CVSPTW). The CVSPTW is more complex than the CVSP, because of the time windows. Moreover, the synchronization of transfers yields a further complication.

We first introduce bounds on the starting times of the trips. The starting time for $i \in \mathcal{V}$ is denoted by $t_i \in \mathbb{Z}_+$. The set $\mathcal{V}$ is divided into two subsets, $\mathcal{V} = \mathcal{C} \dot\cup \mathcal{S}$. Set $\mathcal{C}$ comprises all trips i with a connected starting time interval $[\underline{t}_i, \overline{t}_i] \subseteq [0, 1439] \cap \mathbb{Z}$ bounding the starting time:

$$\underline{t}_i \le t_i \le \overline{t}_i. \tag{8}$$

In particular, each trip has to start during the first day. If the starting time exceeds this limit, the interval is split into two. This is the case for all trips $i \in \mathcal{S}$. Their starting time must be in $([\underline{t}_i, 1439] \cup [0, \overline{t}_i]) \cap \mathbb{Z}$. We introduce a binary variable $y_i \in \{0, 1\}$ with $y_i = 1$ if the starting time is in $[0, \overline{t}_i]$ (after midnight) and $y_i = 0$ if it is in $[\underline{t}_i, 1439]$ (before midnight). We then obtain the following constraints for the starting times:

$$\underline{t}_i(1 - y_i) \le t_i \le 1439 + (\overline{t}_i - 1439)y_i. \tag{9}$$

Note that the partition of $\mathcal{V}$ into disjoint subsets $\mathcal{C}$ and $\mathcal{S}$ strongly depends on the selection of the reference time system. It is of course natural to take midnight as the origin (time 0 or 1440). However, this natural selection is not the best from a computational point of view. It turns out that the solution time is higher the larger the set $\mathcal{S}$ is. Hence by a suitable selection of the reference time (for instance, in our model 0 represents 10 a.m. in reality) we can minimize the number of elements in $\mathcal{S}$ (hence maximize $|\mathcal{C}|$). The details and computational merits will be further discussed below.

The starting time of the trips should only be changed if necessary. That means, we introduce a variable v_i for all trips $i \in \mathcal{V}$ that keeps track of the time difference between the current starting time $\hat{t}_i$ and the planned starting time t_i. A minimization of all v_i is then part of the objective function. Formulating the constraints we have to distinguish trips in $\mathcal{C}$ from trips in $\mathcal{S}$. For those in $\mathcal{C}$ it is required that $|\hat{t}_i - t_i| \le v_i$ or in terms of linear inequalities

$$\hat{t}_i - t_i \le v_i, \tag{10}$$

$$t_i - \hat{t}_i \le v_i, \tag{11}$$

whereas for trips in $\mathcal{S}$ we have four constraints, depending on whether the current starting $\hat{t}_i$ time is shortly before or shortly after midnight (or time 0, to be more precise). That is, for all $i \in \mathcal{S}$ with $\hat{t}_i \geq \underline{t}_i$ (i.e., shortly before midnight) we have

$$\hat{t}_i - (t_i + 1440\, y_i) \leq v_i, \tag{12}$$

$$(t_i + 1440\, y_i) - \hat{t}_i \leq v_i, \tag{13}$$

and for the others with $\hat{t}_i \leq \overline{t}_i$ (i.e., shortly after midnight) we have

$$\hat{t}_i - (t_i - 1440\,(1 - y_i)) \leq v_i, \tag{14}$$

$$(t_i - 1440\ (1 - y_i)) - \hat{t} \leq v_i. \tag{15}$$

As in the CVSP model we introduce decision variables $x_{b,(i,j)} \in \{0, a_{b,(i,j)}\}$ for the deadhead trips and use the same multi-commodity flow formulation:

$$\sum_{b \in \mathcal{B}} \sum_{i:(i,j) \in \mathcal{A}} x_{b,(i,j)} = 1 \tag{16}$$

and

$$\sum_{i:(i,j) \in \mathcal{A}} x_{b,(i,j)} = \sum_{k:(j,k) \in \mathcal{A}} x_{b,(j,k)}. \tag{17}$$

If $i, j \in \mathcal{V}$ are connected, then the corresponding starting times must be synchronized,

$$t_i + \delta_{b,(i,j)} - 1440 l_{b,(i,j)} \leq t_j + 7260(1 - x_{b,(i,j)}). \tag{18}$$

The constant $7260 = 1440 + 5820$ reflects the assumption that a trip's starting time is fixed to the first day (hence it is at most 1440) and 5820 is an upper bound for $\delta_{b,(i,j)}$ (see Section 4.1). Since the starting time selection is now integrated in the model, we cannot compute the number of locomotives $\lambda_{b,(i,j)}$ beforehand, as in the case of the CVSP. Instead we introduce a variable $l_{b,(i,j)} \in \mathbb{Z}_+$ which represents the number of additional locomotives with respect to the connection of i with j.

Finally we have to synchronize those trips $i, j \in \mathcal{V}$ where cars transfer from one to the other. Denote by $\mathcal{P}$ the set of pairs (i, j) where cars transfer from i to j. Since all trips operate cyclically every day, a car that misses its transfer due to a late arrival will be picked up by the same trip on the next day. That is, if j departs before the arrival of i, then the cars have to wait at most 24 hours until the arrival of the next trip j. This is modeled by the following inequalities:

$$0 \leq (t_j + 1440 q_{i,j}) - (t_i + \delta_i^{\text{trp}} + \delta_{i,j}^{\text{shnt}}) \leq 719 + 720 p_{i,j}, \tag{19}$$

where

- $\delta_{i,j}^{\text{shnt}}$ denotes the time for shunting the car from i to j.
- $q_{i,j} \in \{0, \ldots, 4\}$ is an auxiliary variable to map the starting time of j into the same day as the arrival of i. The upper bound 4 results from the fact that in all our instances no trip is longer than four days.

– $p_{i,j} \in \{0,1\}$ is a decision variable with $p_{i,j} = 1$ if and only if the cars from i to j are idle for more than 12 hours. In this case we speak of a *missed transfer*.

Since missed transfers, which lead to long idle waiting times, are undesirable, the variables $p_{i,j}$ are put into the objective function with a suitably large scaling coefficient. This way it is possible to analyze how many locomotives could potentially have been saved if some transfers were missed.

The objective is to minimize the total costs defined as

$$\sum_{(i,j)\in\mathcal{A}} \gamma_{i,j}^{\text{idle}} p_{i,j} + \sum_{b\in\mathcal{B}} \sum_{(i,j)\in\mathcal{A}} (\gamma_b^{\text{cls}} l_{b,(i,j)} + \gamma_{b,(i,j)}^{\text{dhd}} x_{b,(i,j)} + \gamma_i^{\text{diff}} v_i), \tag{20}$$

where γ_b^{cls} and $\gamma_{b,(i,j)}^{\text{dhd}}$ are defined as above, $\gamma_{i,j}^{\text{idle}}$ denotes the costs for idle cars, i.e., a car missing its subsequent train resulting in a waiting time of more than 12 hours, and γ_i^{diff} denotes the costs for changing the starting time of a trip. In general, these coefficients reflect the following ordering: Reducing idle cars is most important, since high contract penalties for late arrivals have to be paid. The reduction of the deployed locomotives is second and the reduction of deadhead costs comes third. The least important goal is the minimization of the starting time differences.

It might also be possible to set $p_{i,j} := 0$ for all transfers $(i,j) \in \mathcal{P}$, which gives a hard constraint, and hence also reflects that the transfers are much more important than saved locomotives. However, in this approach it has to be checked beforehand that feasible solutions exist for the problem instance. This can for instance be done by solving the CVSP with fixed starting times.

4.3 Netload-dependent Travel Times

We further refine the above CVSPTW model to the case that the duration of the trips is not constant, but a function depending on the total network load. At daytime passenger trains have higher priority than (most of the) freight trains which often leads to waiting times for the latter. Hence the average traveling speed of a freight train is much lower than at nighttime. To this end, the whole day is partitioned into a discrete number of time slices $\mathcal{H} = \{1,\dots,H\}$, that is, $[0,1439] = \dot{\bigcup}_{h\in\mathcal{H}} [\underline{\tau}_h, \overline{\tau}_h]$. For each trip i we introduce decision variables $z_{i,h} \in \{0,1\}$ with $z_{i,h} = 1$ if and only if the trip starts within time slice h. Exactly one slice must be selected:

$$\sum_{h\in\mathcal{H}} z_{i,h} = 1. \tag{21}$$

The slice selection is coupled to the starting time of trip i such that the "right" slice h is chosen:

$$\underline{\tau}_h - t_i \le 1439(1 - z_{i,h}), \tag{22}$$

$$t_i - \overline{\tau}_h \le 1439(1 - z_{i,h}). \tag{23}$$

Then the trip duration of trip i is given by

$$d_i^{\text{trp}} = \sum_{h \in \mathcal{H}} \delta_{i,h}^{\text{trp}} z_{i,h}, \tag{24}$$

where $d_i^{\text{trp}} \in \mathbb{Z}_+$ is a new variable and $\delta_{i,h}^{\text{trp}}$ is a parameter giving the trip duration of trip i when being started in slice h. The actual values of $\delta_{i,h}^{\text{trp}}$ are statistically estimated from historical data. In the CVSPTW model, δ_i^{trp} is replaced by d_i^{trp}.

Moreover, in case of dynamic trip durations it is desired to specify bounds on the arrival time of some of the trips in addition to the starting time bounds. Again, we have to distinguish between trips in $\mathcal{C}$ and trips in $\mathcal{S}$. For all $i \in \mathcal{C}$ it is required that

$$\underline{T}_i \leq t_i + d_i^{\text{trp}} \leq \overline{T}_i, \tag{25}$$

where $[\underline{T}_i, \overline{T}_i] \subseteq [0, 5759] \cap \mathbb{Z}$ is the arrival time interval of trip i. Here the constant $5759 = 3 \cdot 1440 + 1439$ reflects the largest trip duration (3 days) plus the selection of the starting time (some time on the first day). For all trips $i \in \mathcal{S}$ we have

$$\underline{T}_i \leq t_i + (1440 y_i) + d_i^{\text{trp}} \leq \overline{T}_i, \tag{26}$$

if $\hat{t}_i \geq \underline{t}_i$ (i.e., the train starts shortly before midnight) and

$$\underline{T}_i \leq t_i - (1440(1 - y_i)) + d_i^{\text{trp}} \leq \overline{T}_i, \tag{27}$$

if $\hat{t}_i \leq \bar{t}_i$ (i.e., the trip starts shortly after midnight).

5 Solving the Model: Primal and Dual Aspects

For the solution of difficult combinatorial optimization problems, such as the CVSPTW, one can attack the problem from two sides.

One side, the primal one, deals with the development of efficient heuristics which are able to generate feasible solutions in a short amount of time. Any such solution is an upper bound on the value of an optimal solution of the problem. In Section 5.1 we describe our heuristic approach for the problem at hand.

The other side is the computation of dual (lower) bounds for the objective function value of an optimal solution. One way to obtain such bounds is to compute the linear programming (LP) relaxation, where all but the integrality constraints of the model are taken into account. The LP relaxation can efficiently be computed by numerical methods, such as Dantzig's Simplex algorithm. In order to improve the lower bound, one re-introduces the integrality constraints by further cutting planes. The whole procedure is embedded in a branch-and-bound framework. A linear programming based branch-and-bound algorithm that makes use of cutting planes is also called *branch and cut* method. Today, good implementations of this method are available which can solve even difficult and large-scale instances of (mixed) integer linear programming problems (for instance the solver Cplex from ILOG, which we will use in the sequel).

However, we can further speed up the solution process and improve the quality of the generated solutions, if we additionally help such general purpose solver to better understand our particular problem. That means, we can hand over our best primal solution as starting solution to the solver, which help to reduce the size of the branch-and-bound tree. Further, we found that special purpose reformulation and presolving techniques (see Section 5.2 and Section 5.3), and the inclusion of valid cutting planes (Section 5.4) are crucial to improve the formulation of the problem and thus to solve complex instances in shorter time.

5.1 A Primal Heuristic

We developed a randomized PGreedy-type heuristic that constructs feasible solutions from scratch. (For an introduction to parametrized greedy (PGreedy) heuristics we refer to [21].) This heuristic consists of two steps, where the output of the first step is taken as input for the second.

Step 1: Car Transfers. In the first step, the heuristic seeks to determine a minimum number of missed car transfers, because they have the largest impact in the objective function (20). To this end, the set $\mathcal{P}$ of all car transfers is split into three disjoint subsets $\mathcal{P} = \mathcal{P}^w \dot\cup \mathcal{P}^g \dot\cup \mathcal{P}^b$, called white, gray, and black, respectively. The actual assignment of transfers $(i,j) \in \mathcal{P}$ to one of the three groups is based on the influence of the transfers on the starting time windows of trips i and j. That means, we tentatively set $p_{i,j} := 0$ and consider the following subsystem:

$$0 \le (t_j + 1440 q_{i,j}) - (t_i + \delta_i^{\mathrm{trp}} + \delta_{i,j}^{\mathrm{shnt}}) \le 719, \tag{28}$$

$$\underline{t}_i \le t_i \le \bar{t}_i, \tag{29}$$

$$\underline{t}_j \le t_j \le \bar{t}_j. \tag{30}$$

Using constraint propagation techniques we evaluate the influence of (28) on the starting time windows (29) and (30). One of the following three cases will occur.

White. If $\underline{t}_i, \bar{t}_i, \underline{t}_j, \bar{t}_j$ remain unchanged, then (i,j) is a "white" relation and hence put into $\mathcal{P}^w$. That is, the starting time of trips i and j is totally independent from the car transfer (i,j).

Black. If the system (28), (29), (30) is infeasible, then (i,j) is called a "black" relation and put into $\mathcal{P}^b$. It is not possible to select starting times for both trips such that the cars have to wait no more than 720 minutes at the transfer station.

Gray. All other transfers (i,j) which are neither white nor black are called "gray" and put into $\mathcal{P}^g$. That means, the car transfer has some influence on the starting time windows $\underline{t}_i, \bar{t}_i, \underline{t}_j, \bar{t}_j$.

For the white transfers $(i,j) \in \mathcal{P}^w$ we permanently set $p_{i,j} := 0$, and for the black transfers $(i,j) \in \mathcal{P}^b$ we set $p_{i,j} := 1$.

It remains to decide which of the gray ones can be considered as white and which as black. This is done iteratively by selecting one element $(i,j) \in \mathcal{P}^g$ setting tentatively $p_{i,j} := 0$ and updating the time windows by constraint propagation of the corresponding subsystem (28), (29), (30). If the subsystem is feasible, we remove (i,j) from $\mathcal{P}^g$, permanently set $p_{i,j} := 0$, and put it into $\mathcal{P}^w$. Otherwise, if the subsystem is infeasible, we also remove (i,j) from $\mathcal{P}^g$, but now we permanently set $p_{i,j} := 1$, and put it into $\mathcal{P}^b$. Afterwards the next element from $\mathcal{P}^g$ is selected, as long as this set is non-empty. Note that each time we move an element (i,j) from $\mathcal{P}^g$ into $\mathcal{P}^w$, we keep the updated (hence smaller) time windows on the starting times of the trips.

In the end we obtain a disjoint partition of $\mathcal{P}$ into two subsets $\mathcal{P}^b$ and $\mathcal{P}^w$. Since the size of these two sets depends on the actual ordering in which elements from $\mathcal{P}^g$ are selected, we repeat the overall procedure a number of times, each time with different ordering among the elements of $\mathcal{P}^g$. The output of this first step of the heuristic is the overall best partition, i.e., the partition with the smallest set $\mathcal{P}^b$ and largest set $\mathcal{P}^w$.

Step 2: Operating Costs. In the second part of the heuristic we seek a minimum number of locomotives and a minimum total length of all deadhead trips, given the car transfers from the first step as fixed input data. The heuristic iteratively assigns locomotives to active trips and deadhead trips. To this end, the following steps are repeated until all trips are assigned.

Locomotive selection. We first make a selection of the locomotive type the heuristic shall proceed with. We hereto count the number of available deadhead trips in $\mathcal{A}$ per type class b. This number A_b is divided by the cost γ_b^{cls} of this class. The ratio $\frac{A_b}{\gamma_b^{\text{cls}}}$ can be interpreted as a value for the flexibility of the particular type of locomotive. We randomly select a class b with a bias corresponding to the flexibility ratios. That means, the higher the value $\frac{A_b}{\gamma_b^{\text{cls}}}$, the higher the probability for class b to be chosen.

Start trip selection. Among all trips which are unassigned so far we seek one with a high number of possible deadhead trips of the same type as our selected locomotive type class b. Again, we select this start trip i randomly, with a bias corresponding to the number of possible deadheads per trip.

Starting time selection. We fix the starting time of the selected trip. The starting time t_i can be taken from the interval $[\underline{t}_i, \overline{t}_i]$. We select t_i in such way that the arrival time is the earliest possible. (Note that we have netload dependent driving times, which means that we have to subdivide the interval corresponding to the time slices.) Once the starting time is selected, we might have to evaluate the influence on the other starting time windows. Such an influence occurs if there is some trip j with (i,j) or $(j,i) \in \mathcal{P}$, since trips i and j (and thus their starting time windows) are dependent in this case. Using constraint propagation the starting time interval of trip j might change.

Deadhead trip selection. The deadhead trip for the locomotive to the start of the next trip is also selected randomly. Here the process is biased according to the length of the deadhead trip.

In the above heuristic all steps rely on random selections. Hence we repeat the whole procedure several times (up to a few hundred, depending on the given time limit and the size of the instance), and finally return the best overall solution.

5.2 Selecting Time 0

It was mentioned in Section 4.2 that the partition of $\mathcal{V}$ into $\mathcal{S}$ and $\mathcal{C}$ depends on the position of the origin of the time. Since trips in $\mathcal{S}$ require an additional binary decision variable and twice as many constraints (and with more complicated structure) as trips in $\mathcal{C}$, it is desired to have $|\mathcal{S}|$ as small as possible. The natural choice "0 = midnight" leads to a high number of split starting time intervals, because most freight trains operate during the night when no passenger trains congest the railway network. Hence this choice results in large sets $\mathcal{S}$.

Taking this into account we first compute the best position for the origin of the time before setting up the model. For each integer value $n \in [0, 1439]$ we obtain the partition $\mathcal{V} = \mathcal{S}_n \dot{\cup} \mathcal{C}_n$ as follows: Trip $i \in \mathcal{V}$ is in $\mathcal{S}_n$ if and only if $n \in [\underline{t}_i, \bar{t}_i]$. The particular value n_0 that minimizes $|\mathcal{S}_n|$ is taken as the new origin of time. Figure 1 shows a plot of the graph of $n \mapsto |\mathcal{S}_n|$ for one of our test instances. One can clearly see a peak between $1,000$ and $1,300$ (i.e., around 5 p.m. till 10 p.m.) which reflects the fact that most freight trains start in the late afternoon/evening, when fewer passenger trains are on the tracks. Vice versa, there is a low mark in the morning between 350 and 600 (i.e., around 6 a.m. till 10 a.m.), which corresponds to the morning rush hours. For this instance, $n_0 := 415$ with $|\mathcal{S}_{n_0}| = 17$ would be the best possible selection.

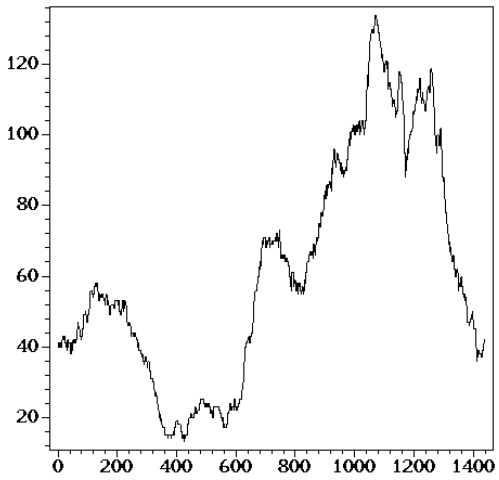

Fig. 1. Number of trips in $\mathcal{S}_n$ for time origin at n.

5.3 Preprocessing

Before actually solving the model by branch-and-cut techniques it is desirable to have a compact formulation with as few constraints and variables as possible. In our model we can remove some of the $l_{b,(i,j)}$ variables.

Denote by $\underline{d}_i^{\text{trp}}$ and $\overline{d}_i^{\text{trp}}$ the shortest and the longest possible trip duration, respectively. That is,

$$\underline{d}_i^{\text{trp}} := \min\{\delta_{i,h}^{\text{trp}} : h \in \mathcal{H}\}, \tag{31}$$

$$\overline{d}_i^{\text{trp}} := \max\{\delta_{i,h}^{\text{trp}} : h \in \mathcal{H}\}. \tag{32}$$

Then we can compute lower and upper bounds on the number of locomotives for each deadhead trip (i, j) and type class b as follows:

$$\underline{l}_{b,(i,j)} := \left\lceil \frac{\underline{t}_i + \underline{d}_i^{\text{trp}} + \delta_i^{\text{uncpl}} + \delta_{b,(i,j)}^{\text{dhd}} + \delta_j^{\text{cpl}} - \overline{t}_j}{1440} \right\rceil, \tag{33}$$

$$\overline{l}_{b,(i,j)} := \left\lceil \frac{\overline{t}_i + \overline{d}_i^{\text{trp}} + \delta_i^{\text{uncpl}} + \delta_{b,(i,j)}^{\text{dhd}} + \delta_j^{\text{cpl}} - \underline{t}_j}{1440} \right\rceil. \tag{34}$$

If $\underline{l}_{b,(i,j)} = \overline{l}_{b,(i,j)}$ then we can fix variable $l_{b,(i,j)}$. In detail, we replace inequalities (18) by

$$t_i + \delta_{b,(i,j)} - 1440\,\underline{l}_{b,(i,j)} \le t_j + 7260\,(1 - x_{b,(i,j)}), \tag{35}$$

and in the objective function we replace

$$\gamma_b^{\text{cls}} l_{b,(i,j)} + \gamma_{b,(i,j)}^{\text{dhd}} x_{b,(i,j)} \tag{36}$$

by

$$(\gamma_b^{\text{cls}} + \gamma_{b,(i,j)}^{\text{dhd}}) x_{b,(i,j)}. \tag{37}$$

Otherwise, if $\underline{l}_{b,(i,j)} < \overline{l}_{b,(i,j)}$, we can use this information as an additional valid inequality.

5.4 Valid Inequalities

Valid inequalities or cutting planes are generally used in integer linear programming to strengthen the linear programming relaxation (see for instance text books such as [22] or survey articles like [23]).

Analyzing the LP relaxation of our model we found that almost all variables $l_{b,(i,j)}$ are at their lower bound 0. This is because the only coupling of $l_{b,(i,j)}$ variables to other variables is via inequalities (18), which is weak due to the big-M linearization. To improve on that it is necessary to give additional stronger combinatorial constraints. Such constraints can be obtained using formulas (33) and (34):

$$\underline{l}_{b,(i,j)} x_{b,(i,j)} \le l_{b,(i,j)} \le \overline{l}_{b,(i,j)} x_{b,(i,j)}, \tag{38}$$

that is, if $x_{b,(i,j)}$ is set to 1, then $l_{b,(i,j)}$ is forced to be between $\underline{l}_{b,(i,j)}$ and $\bar{l}_{b,(i,j)}$, the minimal or maximal number of locomotives of type b for this connection, respectively. As we will see in Section 6, these inequalities (and especially the left one $\underline{l}_{b,(i,j)} x_{b,(i,j)} \leq l_{b,(i,j)}$) lead already to a significant improvement of the LP relaxation. But, we can do more.

Observe that for all type classes b and all trips j at most one trip i with $(i,j) \in \mathcal{A}$ is chosen (i.e., $x_{b,(i,j)} = 1$ in a feasible solution), because of constraints (16) and (17). There are two implications. First, we can replace all variables $l_{b,(i,j)}$ by $l_{b,i}$, which is a large reduction in the number of variables. Of course, inequalities (18) have to be altered to

$$t_i + \delta_{b,(i,j)} - 1440 l_{b,i} \leq t_j + 7260(1 - x_{b,(i,j)}). \tag{39}$$

And second, we can reformulate inequality (38) as

$$\sum_{j:(i,j)\in\mathcal{A}} \underline{l}_{b,(i,j)} x_{b,(i,j)} \leq l_{b,i} \leq \sum_{j:(i,j)\in\mathcal{A}} \bar{l}_{b,(i,j)} x_{b,(i,j)}, \tag{40}$$

that is, for given trip i and class b we add up the left-hand sides and the right-hand sides in (38) over all j with $(i,j) \in \mathcal{A}$. This is valid since at most one of the corresponding binary variables $x_{b,(i,j)}$ is active ($= 1$) in a feasible solution.

The computational merits of this reformulation will also be discussed in the next section.

6 Computational Results

The CVSP and the CVSPTW problems are formulated as integer linear programming (ILP) problems. For an introduction to integer programming we refer to the literature (Nemhauser and Wolsey [22], or Dell'Amico et al. [24] for instance). For small problem sizes one can use standard ILP solvers to compute proven optimal solutions. For larger problem sizes we may not be able to solve these problems by using such solvers simply as a black-box. Indeed that is the reason for the primal and dual techniques presented in the previous section. Even if large instances remain unsolved, it is still possible to use the optimal solution values for smaller instances as a benchmark for evaluating the heuristic algorithm.

For our computational studies we used ILOG Cplex 10 [15], one of the most robust ILP solvers. We made exhaustive tests with several parameters that guide the solution process and obtained best results when cut generation and probing parameters were set to their highest level. These settings yield the shortest computation times that are reported in the sequel. The software was running on a Linux server with 16 GByte main memory and 4 dual core AMD Opteron 880 processors running at 2.4 GHz each. We used parallel Cplex to exploit all four processors.

From the data base of the tool chain we extracted 7 test instances, which we refer to as `A`, `B`, `C`, `D`, `E`, `F`, `G` in the sequel. Here `A` is the smallest instance with

42 trips and 3 classes of locomotives, whereas G is the largest, having 1,537 trips and 4 classes of locomotives (for details see the first three columns of Table 1 or Table 2). The instances are related to certain regions within the DB railway network. For example, instance G consists of trips mainly from the south of Germany, trips from A serve north-south connections, whereas E comprises trips from all over Germany.

6.1 Results for the CVSP

At first we take the fixed starting times for the trips in the form they were generated by the train scheduling part of the tool chain. Here each trip is assumed to depart as soon as enough cars are assembled. For most instances the solver was able to find optimal solutions within short amount of time, except for G, where more than 3 hours of computation time were needed. Note that an instance with around n trips and m classes leads to integer linear programs with around n^2m variables, which is more than 3 million for instances F and G. For the instances from A to G the solution times mainly increase with increasing problem sizes (see column "time" of Table 1 or Table 2). Problems F and G show that problems of roughly similar size can have widely varying computational times.

name	trips	classes	time [sec]	locomotives	$\sum$	km
A	42	3	1	19/2/2	23	3,808
B	82	3	2	3/41/1	45	52
C	120	4	18	35/1/6/1	43	2,560
D	340	6	20	70/2/2/55/49/1	179	28,274
E	727	6	325	40/1/1/76/67/77	262	40,101
F	1,507	4	652	28/19/287/52	386	58,446
G	1,537	4	9,502	53/22/26/72	173	2,942

Table 1. Solutions for the uncapacitated CVSP.

The actual amount of locomotives per class that is needed to serve all trips is shown in column "locomotives", the sum of these is given in column "$\sum$". Table 1 shows the solutions for the uncapacitated case, that is, it is possible to deploy arbitrarily many locomotives of each class. In contrast, Table 2 shows the solutions of the capacitated case, that is, the number of locomotives per class is limited to the actual stock of DB. In general, the solution times for the capacitated case were higher, in particular for the larger instances E, F, G. However, the total number of locomotives is unchanged (with the exception of instance F), only the locomotives per class are different between the capacitated and the uncapacitated case. The last column of Table 1 and Table 2 shows the sum of deadhead trips lengths for all locomotives. Interestingly, the total length (in kilometer) of deadhead trips decrease when switching from the uncapacitated

name	trips	classes	time [sec]	locomotives	$\sum$	km
A	42	3	1	6/15/2	23	412
B	82	3	2	3/26/16	45	52
C	120	4	7	15/26/1/1	43	1,919
D	340	6	32	35/2/30/62/49/1	179	24,124
E	727	6	2,837	57/26/15/76/66/22	262	40,101
F	1,507	4	4,971	37/19/216/123	395	56,130
G	1,537	4	13,864	53/22/46/52	173	2,838

Table 2. Solutions for the capacitated CVSP.

to the capacitated version of the CVSP. This could be ascribed to an increase in diesel locomotives for example, whose deadhead trips are in general shorter. We further remark that the little amount of deadhead kilometers in instance `G` (in comparison to `E` and `F`) is due to the fact that this instance consists of lots of shuttle services.

6.2 Results for the CVSPTW

In the CVSP instances the starting times of all trips were fixed to the times that were computed by the tool chain's scheduler. We now allow the starting times to be altered within a small time window centered around the pre-scheduled starting time, that is, we consider the (uncapacitated) vehicle scheduling problem with time windows (CVSPTW). It turns out that the computation time heavily depends on the actual size of the time windows. Generally speaking, the larger the time window, the more time the solver needs. For all computations we now impose a time limit of 3,600 seconds. We consider the four smallest instances `A`, `B`, `C`, and `D` and take intervals of $\pm 10, \pm 30, \pm 60$, and ± 120 minutes around the current (pre-scheduled) starting time. For instance `E`, we only consider ± 10 and ± 30. The computational results are shown in Table 3. For the instances marked with a $-$ as well as for the other instances `F` and `G` the solver did not find feasible solutions or did not even solve the root LP relaxation within the time limit. In column "time" of Table 3 we show the computation time in seconds. An asterisk (*) marks whether the given limit was reached. In those cases the integrality gap (i.e., the difference of the best known solution and the corresponding lower bound devided by the best upper bound) shown in column "gap" is non-zero. The last three columns of Table 3 depict the quality of the optimal or best feasible solution. It turns out that there is a significant potential to save locomotives by changing the departure times of the trips. The possible savings are higher the wider the time windows are. The price one has to pay for this additional flexibility is the increasing solution time.

instance	time w.	time [sec]	gap	locomotives	$\sum$	km	dev
A-42-3	± 10	1	0.00 %	18/2/2	22	3,954	5
A-42-3	± 30	3,600*	9.90 %	13/2/2	17	2,420	222
A-42-3	± 60	3,600*	29.16 %	8/3/4	15	2,773	523
A-42-3	±120	3,600*	55.54 %	10/3/2	15	1,447	1,312
B-82-3	± 10	3	0.00 %	3/41/0	44	52	24
B-82-3	± 30	92	0.00 %	3/37/0	40	52	174
B-82-3	± 60	3,600*	8.45 %	2/33/0	35	1,603	594
B-82-3	±120	3,600*	22.96 %	2/30/1	33	5,287	1,521
C-120-4	± 10	9	0.00 %	35/1/5/1	42	2,558	68
C-120-4	± 30	3,600*	35.29 %	30/1/10/1	42	3,711	443
C-120-4	± 60	3,600*	83.78 %	29/1/10/1	41	5,247	1,004
C-120-4	±120	3,600*	96.85 %	23/1/20/1	45	4,372	3,041
D-340-6	± 10	3,600*	0.68 %	69/0/0/55/49/0	173	30,094	110
D-340-6	± 30	3,600*	– %	–	–	–	–
D-340-6	± 60	3,600*	– %	–	–	–	–
D-340-6	±120	3,600*	– %	–	–	–	–
E-727-6	± 10	3,600*	– %	–	–	–	–
E-727-6	± 30	3,600*	– %	–	–	–	–

Table 3. CVSPTW with different time window sizes.

6.3 Results for the CVSPTW with Netloads

Finally we consider the (uncapacitated) CVSPTW with netload-dependent travel times for the trips. This is the most evolved model in our hierarchy and it comes with no surprise that the solution times are even higher than for the CVSPTW with constant traveling times. Our results are summarized in Table 4. Depending on the starting time the total travel time varies between 70 % and 130 % of the constant travel time. The bias of the travel time is estimated using historical data.

The results in Tables 1–4 give an impression on the current state of problem sizes that can be solved using Cplex out of the box with some altered parameter settings. This way the solver is far away from solving even medium-size instances to optimality.

6.4 Results for the Heuristic and Cutting Planes

In this last section of the computational results we describe how the solution behaviour is changed by including the techniques described in Section 5.

It was noted by Rothberg et al. [25] that in cases of lexicographic multi-objective ILPs it is computationally advantageous to solve the overall problem as a sequence of subproblems. In each iteration, they suggest to take a single objective function while keeping the objective function values of the higher ranking

instance	time w.	time [sec]	gap	locomotives	$\sum$	km	dev
A-42-3	± 10	1	0.00 %	18/2/2	22	3,954	5
A-42-3	± 30	3,600*	9.90 %	13/2/2	17	2,730	218
A-42-3	± 60	3,600*	29.90 %	9/2/4	15	2,773	523
A-42-3	±120	3,600*	56.04 %	11/2/2	15	1,848	1,312
B-82-3	± 10	3	0.00 %	3/41/0	44	52	24
B-82-3	± 30	92	0.00 %	3/37/0	40	52	173
B-82-3	± 60	3,600*	8.45 %	2/33/0	35	1,603	594
B-82-3	±120	3,600*	22.96 %	2/30/1	33	5,287	1,521
C-120-4	± 10	9	0.00 %	35/1/5/1	42	2,558	68
C-120-4	± 30	3,600*	35.29 %	30/1/10/1	42	3,948	454
C-120-4	± 60	3,600*	83.78 %	29/1/10/1	41	5,247	1,004
C-120-4	±120	3,600*	96.85 %	23/1/20/1	45	4,191	3,034
D-340-6	± 10	3,600*	0.69 %	69/0/0/55/50/0	174	30,094	126
D-340-6	± 30	3,600*	– %	–	–	–	–
D-340-6	± 60	3,600*	– %	–	–	–	–
D-340-6	±120	3,600*	– %	–	–	–	–
E-727-6	± 10	3,600*	– %	–	–	–	–
E-727-6	± 30	3,600*	– %	–	–	–	–

Table 4. CVSPTW with netload-dependent travel times.

objectives from the previous iterations as hard constraints. This also applies to our model. Hence the actual solution is carried out in the following way:

Step 1. It turns out that the model can be quickly solved for all instances if we only take

$$\sum_{(i,j)\in\mathcal{A}} \gamma_{i,j}^{\text{idle}} p_{i,j} \tag{41}$$

as objective function instead of (20). That is, it is only desired to find the minimum number $\underline{m}$ of missed car transfers (no matter how many locomotives have to be in use in that case).

Step 2. Next we call the primal heuristic to obtain a feasible solution to the problem. This in particular gives an upper bound $\overline{m}$ on the number of missed car transfers. Usually the heuristic produces good solutions which are not far away from the lower bound, within at most one missed transfers of the optimal solution (comparing the values in column "dual (Cplex)/$\underline{m}$" with "primal heuristic/$\overline{m}$" in Table 5).

Step 3. We include another constraint saying that the number of missed car transfers should be between the lower and the upper bound:

$$\underline{m} \le \sum_{(i,j)\in\mathcal{P}} p_{i,j} \le \overline{m}. \tag{42}$$

As objective function we take

$$\sum_{b\in\mathcal{B}}\sum_{(i,j)\in\mathcal{A}}(\gamma_b^{\text{cls}}l_{b,(i,j)}+\gamma_{b,(i,j)}^{\text{dhd}}x_{b,(i,j)})+\gamma_i^{\text{diff}}v_i). \tag{43}$$

Now we call the ILP solver to either prove optimality of the heuristic solution or to find better ones. The results are given in Table 5. The instances are described in the first three columns. In column "$|\mathcal{S}|$" the number of split starting time intervals of the trips is given. The columns under the headline "dual (Cplex)" show the results of the ILP solver Cplex. In Step 1 we start with a computation of the minimum number of missed car transfers. Their number is shown in column 5 ($\underline{m}$) and the time needed for their computation is in column "dual(Cplex)/$\underline{m}$-time". Then we call the primal heuristic in Step 2, which provides a starting solution, see columns under the headline "primal heuristic". Column "primal heuristic/$\overline{m}$" gives the number of missed car transfers, which is the upper bound in Step 3. The number of locomotives and the total deadhead kilometers are given in columns "primal heuristic/locom." and "primal heuristic/km", respectively. The last column, "primal heuristic/root gap" shows the dual bound of the heuristic solution to the root LP relaxation of the model. Together with the lower bound of Step 1 and the upper bounds of Step 2 we now enter Step 3, where the ILP solver tries to improve the solution, or to prove optimality. We give a time limit of one hour. If this is exceeded, a non-zero gap remains, see column "dual(Cplex)/gap" and column "dual (Cplex)/time". The best solution found is presented in columns "dual (Cplex)/locomotives per type class", column "dual (Cplex)/sum of all locomotives", column "dual (Cplex)/km", i.e., the total length of all deadhead trips, and column "dual(Cplex)/dev.", i.e., the average deviation in the starting time when comparing the previous and the optimized starts of the trips.

Comparing the results of Table 5 with those of Table 3 and 4 we see that by inclusion of the presented techniques within the branch-and-cut framework we can solve larger instances of the problem in less time. Consider for example instance D-340-6. Originally the solver was only able to solve those instances with fixed starting times to optimality (Table 1 or 2) or with ±10 minutes starting time intervals to near optimality. For all other instances with more than ±10 minutes the solver did not find a feasible solution within the given time limit. After including our techniques we can solve instances with up to ±30 minutes to optimality, and obtain reasonably good solutions for up to ±120 minutes. The solver now does not only uses the value of the solutions found by the heuristic, but is also able to improve them during the branch-and-bound process.

instance			dual (Cplex)								primal heuristic			
name	time w.	$\|\mathcal{S}\|$	$\underline{m}$	$\underline{m}$-time	gap	time [sec]	locom.	$\sum$	km	*dev.*	$\overline{m}$	locom.	km	root gap
A-42-3	± 10	0	0	0	0.00 %	1	18/2/2	22	3,954	17	0	22	11,088	4.57 %
A-42-3	± 30	3	0	0	0.00 %	1	13/2/2	17	2,730	223	0	19	8,306	13.69 %
A-42-3	± 60	5	0	0	0.00 %	1	11/2/2	15	3,022	482	0	17	6,208	23.88 %
A-42-3	±120	9	0	0	0.00 %	1	10/2/2	14	1,240	1,861	0	16	4,958	25.24 %
B-82-3	± 10	0	0	0	0.00 %	1	3/41/0	44	52	24	0	44	25,247	4.82 %
B-82-3	± 30	0	0	0	0.00 %	1	3/37/0	40	52	173	0	40	22,689	2.43 %
B-82-3	± 60	0	0	0	0.00 %	13	2/33/0	35	1,603	589	0	36	21,214	9.24 %
B-82-3	±120	1	0	0	0.00 %	145	2/29/0	31	678	1,516	0	31	16,494	16.32 %
C-120-4	± 10	0	0	0	0.00 %	2	35/1/5/1	42	2,021	68	0	42	3,538	3.32 %
C-120-4	± 30	0	0	0	0.00 %	22	30/1/5/1	37	3,668	553	0	40	5,031	5.19 %
C-120-4	± 60	2	0	0	0.00 %	99	29/1/4/1	35	5,376	1,417	0	37	6,394	11.62 %
C-120-4	±120	10	0	0	10.20 %	$3,600^*$	25/1/6/1	33	4,012	3,446	0	37	12,456	17.66 %
D-340-6	± 10	0	119	12	0.00 %	33	69/0/0/55/50/0	174	30,094	143	119	204	118,281	18.41 %
D-340-6	± 30	1	90	14	0.00 %	325	65/0/0/52/49/0	166	30,600	1,053	90	201	122,776	21.85 %
D-340-6	± 60	9	75	18	7.12 %	$3,600^*$	61/1/0/54/47/0	163	39,071	3,056	75	197	124,021	26.39 %
D-340-6	±120	26	45	23	30.24 %	$3,600^*$	63/16/5/35/51/1	171	65,397	10,371	45	185	111,309	41.80 %
E-727-6	± 10	0	271	93	0.00 %	1,243	42/0/0/74/68/76	260	45,955	840	271	291	138,000	14.73 %
E-727-6	± 30	4	226	103	19.69 %	$3,600^*$	38/24/21/32/101/53	269	119,693	5,283	227	274	131,905	21.55 %

Table 5. Netload-CVSPTW with primal heuristic and cutting planes.

7 Conclusions and Further Work

In this article we presented a new model for strategic locomotive scheduling. It was in part inspired by models from the locomotive scheduling literature and in part by models for similar optimization problems in public bus transport. However, some additional rail-specific requirements emerged such that neither model could be directly carried over. Our problem was formulated as an integer linear program which allowed us to use general purpose ILP solvers. The evaluation of the capability of such software was part of our project. It turned out that for the cyclic vehicle scheduling problem without time windows, the software was able to solve even larger instances. As soon as time windows enter the scene, the sizes for which global optimal solutions were computed was reduced by orders of magnitude. Only after including primal heuristics and model reformulation techniques, such as valid inequalities and preprocessing, solutions of proven quality can be obtained for instances of relevant size, sometimes even optimal solutions.

Our results show that a significant saving in locomotives can be achieved when the starting times are altered within some interval without degrading the quality of service (number of missed car transfers, for example). The model and solution methods presented in this article aim to be a new tool for the strategic simulation at DB. They will help to compute future predictions. In a typical run, a future customer demand is estimated and then all modes of operation within the company are virtually carried out. Here the presented models will play a crucial role to determine the future locomotive demand.

Further refinements of the presented model are also on our agenda. An example in this direction is the inclusion of deadheading locomotives which are included in trains and thus do not need own power and staff for operating.

Acknowledgements. We thank Dr. Gerald Pfau (Deutsche Bahn AG), Jörg Wolfner (Universität Dortmund, Lehrstuhl Verkehrssysteme und -logistik), and Andreas Ginkel (Universität Göttingen) for fruitful and productive discussions. Parts of this work were carried out at the Hausdorff Institute for Mathematics, Bonn. We are grateful for the kind hospitality. Last but not least we would like to thank the two anonymous reviewers of the ATMOS version of this paper and the three reviewers of this journal version for their various helpful comments.

References

1. Deutsche Bahn: Geschäftsbericht des Konzerns. Deutsche Bahn AG, Potsdamer Platz 2, 10785 Berlin (2005)
2. Railion Deutschland: Geschäftsbericht. Railion Deutschland AG, Rheinstraße 2, 55116 Mainz (2005)
3. Ahuja, R.K., Jha, K.C., Liu, J.: Solving real-life railroad blocking problems. Interfaces **37** (2007) 404 – 419
4. Bussieck, M., Winter, T., Zimmermann, U.: Discrete optimization in public rail transport. Mathematical Programming **79** (1997) 415 – 444

5. Caprara, A., Fischetti, M., Toth, P., Vigo, D.: Algorithms for railway crew planning. Mathematical Programming **79** (1997) 125 – 141
6. Cordeau, J., Toth, P., Vigo, D.: A Survey of Optimization Models for Train Routing and Scheduling. Transportation Science **32** (1998) 988 – 1005
7. Booler, J.: The solution of a railway locomotive scheduling problem. Journal of the Operational Research Society **31** (1980) 943 – 948
8. Smith, S., Sheffi, Y.: Locomotive scheduling under uncertain demand. Transportation Research Record **1251** (1989) 45 – 53
9. Forbes, M., Holt, J., Watts, A.: Exact solution of locomotive scheduling problems. Journal of the Operational Research Society **42** (1991) 825 – 831
10. Ziarati, K., Soumis, F., Desrosiers, J., Solomon, M.: A Branch-First, Cut-Second Approach for Locomotive Assignment. Management Science **45** (1999) 1156 – 1168
11. Ziarati, K., Chizari, H., Nezhad, A.: Locomotive Optimization Using Artificial Intelligence Approach. Iranian Journal of Science & Technology, Transaction B, Engineering **29** (2005) 93 – 105
12. Ahuja, R., Liu, J., Orlin, J., Sharma, D., Shughart, L.: Solving real-life locomotive scheduling problems. Transportation Science **39** (2005) 503 – 517
13. Powell, W., Bouzaiene-Ayari, B.: Approximate dynamic programming for locomotive optimization. Technical report, Princeton University, Princeton, NJ 08544 (2006)
14. Powell, W., Bouzaiene-Ayari, B.: Approximate dynamic programming for rail operations. Technical report, Princeton University, Princeton, NJ 08544 (2007)
15. ILOG Ltd.: ILOG Cplex 10 Solver Suite. Technical report, ILOG Cplex Division, 889 Alder Avenue, Suite 200, Incline Village, NV 89451, USA (2006)
16. Daduna, J., Völker, M.: Fahrzeugumlaufbildung im ÖPNV mit unscharfen Abfahrtszeiten. Der Nahverkehr **11** (1997) 39 – 43
17. Fügenschuh, A.: The Integrated Optimization of School Starting Times and Public Transport. Logos Verlag, Berlin (2005)
18. Liebchen, C., Möhring, R.: The modeling power of the periodic event scheduling problem: Railway timetables – and beyond. Technical Report 2004/20, Technische Universität Berlin (2004)
19. Ahuja, R., Magnanti, T., Orlin, J.: Network Flows: Theory, Algorithms, and Applications. Prentice Hall, New Jersey (1993)
20. Löbel, A.: Optimal Vehicle Scheduling in Public Transit. Shaker Verlag, Aachen (1997)
21. Fügenschuh, A.: Parametrized Greedy Heuristics in Theory and Practice. In Aguilera, M.B., Blum, C., Roli, A., Sampels, M., eds.: Hybrid Metaheuristics, Second International Workshop, HM 2005, Barcelona, Spain, August 29-30, 2005. Number 3636, Lecture Notes in Computer Science (2005) 21 – 31
22. Nemhauser, G., Wolsey, L.: Integer and Combinatorial Optimization. Wiley Interscience, New York (1999)
23. Fügenschuh, A., Martin, A.: Computational Integer Programming and Cutting Planes. In: Handbooks in Operations Research and Management Science. Volume 12. K. Aardal, G. Nemhauser, R. Weismantel (2005) 69 – 122
24. Dell'Amico, M., Maffioli, F., Martello, S., eds.: Annotated bibliographies in combinatorial optimization. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons Ltd., Chichester (1997)
25. Rothberg, E., Bixby, R., Fenelon, M., Gu, Z., Wunderling, R.: Solving multiobjective models. In Bixby, R., Martin, A., Simchi-Levi, D., Zimmermann, U., eds.: MFO Report No. 19/2004, Mathematisches Forschungsinstitut Oberwolfach (2004) 1028 – 1030