A Hypergraph Model for Railway Vehicle Rotation Planning
A Hypergraph Model
for
Railway Vehicle Rotation Planning*

Ralf Borndörfer
Markus Reuther
Thomas Schlechte
Steffen Weider

September 22, 2011

Abstract

We propose a model for the integrated optimization of vehicle rotations and vehicle compositions in long distance railway passenger transport. The main contribution of the paper is a hypergraph model that is able to handle the challenging technical requirements as well as very general stipulations with respect to the “regularity” of a schedule. The hypergraph model directly generalizes network flow models, replacing arcs with hyperarcs. Although NP-hard in general, the model is computationally well-behaved in practice. High quality solutions can be produced in reasonable time using high performance Integer Programming techniques, in particular, column generation and rapid branching. We show that, in this way, large-scale real world instances of our cooperation partner DB Fernverkehr can be solved.

1 Introduction

Vehicle rotation planning is concerned with the assignment of vehicles to trips of a timetable and the concatenation of these trips to rotations. A ICE railcar, as operated by Deutsche Bahn, is a very expensive asset. Therefore, the integrated mathematical optimization of vehicle resources and deadhead trips1 is of enormous interest. However, despite intense research efforts of the railway optimization community in the past decades, see [1], [2], [4], [6], and [8], the solution of large-scale scenarios that integrate vehicle scheduling, train composition, and regularity aspects remains a mathematical and computational challenge until today.

*This research was funded by DB Fernverkehr AG.
1A deadhead trip is a trip without passengers transferring vehicles between passenger trips.
A high level description of the vehicle rotation planning problem is as follows. A timetabled trip can be operated by several alternative vehicle configurations. A vehicle configuration is a composition of a multiset of single vehicles. It is a planning decision which vehicle configuration is used for timetabled and moreover for deadhead trips. The choice of vehicle configurations is governed by a set of rules.

We focus in this paper on strategic rolling stock decisions by considering a cyclic planning horizon over one standard week. The structure of the timetable, which is our input schedule, is almost periodic. Only few trips or parts of the trips differ over the single week days of the standard week. In view of this structure, it is desirable to also construct a regular vehicle rotation plan. Such a plan is compactly representable, easy to communicate, and easy to operate. We propose a novel concept to define and optimize regularity.

The above mentioned requirements of a vehicle rotation plan, i.e., train composition and regularity, can be handled by constructing a suitable dense directed hypergraph, that represents a compact formulation for the train composition and regularity requirements. Based on this hypergraph, the vehicle rotation planning problem can be modeled by an integer program. The structure of this IP resembles a classical network flow problem (although the problem is NP-hard in general). It can be solved by column generation and large scale Integer Programming techniques. To our best knowledge there is no standard approach in the literature which can handle all of these technical requirements from practice in a fully integrated way.

The paper is organized as follows. In Section 2 we describe the Vehicle Rotation Planning Problem from a practical point of view. Section 3 explains the developed graph-theoretic and Integer Programming model. The solution method is described in Section 4. We use an adaption of the arc generation LP solving technique, see [7], as well as a specialization of the well known IP branching heuristic – called rapid branching, see [9]. We work in a close cooperation with our partner DB Fernverkehr AG, who is one of the largest intercity railway companies in Europe. We have extensively evaluated our model and algorithm on a large set of real world problem instances. In Section 5 we present computational results for a large set of real-world instances.

2 The Vehicle Rotation Planning Problem

In this section we give a formal description of the considered vehicle rotation planning problem (VRP) by introducing major technical concepts of our railway application at DB Fernverkehr.

As mentioned above we focus on a cyclic planning horizon of one week. A date is a certain point in time in our standard week specified by a week day and a time of the day. The duration from date $a$ to date $b$ is the minimal time
needed to wait from \( a \) until \( b \). Therefore, the duration is well defined since, by definition, a duration is always less the duration of the week.

Consider a set of timetabled trips \( T \). A trip \( t \in T \) consists of a list of successive stops. A stop has a location, an arrival date, and a departure date. The first stop of a trip has no arrival date, the last stop has no departure date.

A vehicle group is the most basic type of the physical vehicle resources. In other contexts this is called vehicle type, fleet, or even commodity. It is called "group" because it can represent a traction unit, an aggregated composition of wagons or locomotives, or even single rail cars. The set of vehicle groups is denoted by \( F \). The amortization costs for one week for a vehicle group \( f \in F \) are denoted by \( c(f) \).

A vehicle configuration (or short configuration) is a non-empty multiset of vehicle groups. It represents a temporary coupling of its vehicle groups. A trivial configuration is a configuration of cardinality one. The set of vehicle configurations is denoted by \( C \). The operational cost per kilometer of a configuration \( c \in C \) is denoted by \( c(c) \). Note that the operational costs are per vehicle configuration and not per vehicle group. This is because the costs for allocating a track – for passenger and also for deadhead trips – are per trip and not per rail car. It is much cheaper to allocate a track for two vehicles in a non-trivial configuration than for two vehicles in trivial configurations individually.

For each trip \( t \in T \) there exists a set of feasible vehicle configurations \( C(t) \subseteq C \) which can be used to operate \( t \). A vehicle configuration can be changed at the departure of the first stop and at the arrival of the last stop of a trip but not inside a trip. A change of a vehicle configuration is called coupling\(^2\). For \( t \in T \) and \( c \in C(t) \) we have a special technical time – called turn time for cleaning and maintaining the involved vehicle resources after the trip \( t \) is done. Note that this time depends on the used vehicle configuration:

**Example 1.** Consider a set of two vehicle groups \( F = \{f_1, f_2\} \) and a trip \( t \in T \) which has three feasible vehicle configurations \( C(t) = \{c_1, c_2, c_3\} \subseteq C \). Let \( c_1 = \{f_1\}, c_2 = \{f_1, f_2\}, \) and \( c_3 = \{f_1, f_1\} \). This can be interpreted as follows. It is possible to operate \( t \) with a trivial and two non-trivial configurations. Moreover it is sufficient to cover \( t \) by the trivial configuration \( c_1 \). But in addition it is possible to haul two alternative vehicle groups by operating \( t \). Another point of view for the feasible vehicle configurations of \( t \) is that \( c_2 \) and \( c_3 \) are two alternatives for \( c_1 \). Both can be used to enforce the passenger capacity of \( c_1 \).

Let \( t_1, t_2 \in T \) be two trips with vehicle configurations \( c_1 \in C(t_1) \) and \( c_2 \in C(t_2) \). We denote by \( d(t_1, t_2) \) the duration from the arrival date of \( t_1 \) to the departure date of \( t_2 \). In order to check if it is feasible to connect \( t_1 \) with \( t_2 \) several technical requirements must be fulfilled.

**Rule 1.** If \( c_1 = c_2 \) we check if the turn time after operating \( t_1 \) with \( c_1 \) plus

\(^2\)We consider only coupling activities that can be made on the fly, i.e., without the need of special machines and crews.
the driving time from the arrival location of \( t_1 \) to the departure location of \( t_2 \) is smaller or equal than \( d(t_1, t_2) \).

**Rule 2.** If \( c_1 \neq c_2 \) we first decouple \( c_1 \) and \( c_2 \) into trivial configurations and consider all connections between two equal trivial configurations of \( t_1 \) and \( t_2 \). We proceed as in the first rule using the turn time of \( c_1 \) for these connections.

A **vehicle rotation** is a cyclic concatenation of trips which are operated by a vehicle group. The number of physical vehicle groups needed to operate a vehicle rotation is the number of times the cycle passes the whole standard week. It is not decidable whether a single rotation is feasible or not without knowing the vehicle configurations of the involved trips.

A **vehicle rotation plan** is an assignment of vehicle configurations, timetabled trips, and a set of feasible connections between these configurations such that each used vehicle group rotates in a vehicle rotation.

As motivated in Section 1 regularity in vehicle rotation planning is an important aspect of the VRP. A **train** is a non-empty set of at most seven trips having the same departure time, departure location, arrival time, and arrival location but pairwise different days. The set of all trains is denoted by \( \mathcal{T} \).

The main aim of regularity is to construct the vehicle rotation plan such that the connections of trains are preferably repeating on the seven week days like the trips of an almost periodic timetable repeating on the seven week days.

The vehicle rotation problem is to find a cost optimal vehicle rotation plan.

### 3 Hypergraph based Integer Programming model

The considered vehicle rotation planning problem can be modeled by using a hypergraph based Integer Programming formulation. First of all, we describe how all the technical aspects from Section 1 are handled in our graph theoretic model. Second, we introduce an Integer Programming model which integrates the whole VRP.

#### 3.1 Hypergraph model

Since a vehicle configuration \( c \in C \) is a multiset, we denote the number of elements – called **multiplicity** – in \( c \) of a vehicle group \( f \in F \) by \( m(f, c) \). In order to clearly identify the elements of a vehicle configuration \( c \in C \) we index all elements of vehicle group \( f \in F \) in \( c \) by natural numbers \( \{1, \ldots, m(f, c)\} \subseteq \mathbb{N} \).

We define a directed hypergraph \( G = (V, H, A) \) with node set \( V \), hypernode set \( H \) and hyperarc set \( A \). Our definition of a directed hypergraph is slightly
sets different to definitions from the literature (see [5]) and therefore we define the
sets $V$, $H$, and $A$ as follows:

A node $v \in V$ is a four-tuple $v = (t, c, f, m) \in T \times C \times F \times \mathbb{N}$ and represents a
trip $t \in T$ operated with a vehicle configuration $c \in C(t)$ and with vehicle
group $f \in c$ of multiplicity $m \in \{1, \ldots, m(f,c)\}$.

The set $V(t,c) = \{(t,c,f,m) | t = t, c = c\}$ denotes all nodes belonging to a
trip $t \in T$ operated with a vehicle configuration $c \in C(t)$. Each $V(t,c)$ with
t $\in T$ and $c \in C(t)$ is a hypernode $h \in H$. A hypernode can been seen as a
feasible assignment of a vehicle configuration to a trip.

A link is a tuple $(v, w) \in V \times V$. A hyperarc $a \in A$ – or short arc – is a non-
empty set of links, thus $a \subseteq V \times V$. For $a \in A$ we define the tail component
of $a$ by tail$(a) = \{v \in V | \exists w \in V : (v, w) \in a\}$ and the head component by
head$(a) = \{v \in V | \exists u \in V : (u, v) \in a\}$. Note that in contrast to [5] we
assume that the tail set and head set of a hyperarc must be not empty and of
equal cardinality. In addition we do not assume that the tail set and head set
have to be disjoint.

The arcs $A$ of the graph $G$ can be partitioned in three sets. In the following
we describe the construction:

**Step 1.** We construct all configuration conserving arcs – all arcs without a
coupling activity. This means that we iterate over all pairs of trips $t_1, t_2 \in T$
having a common feasible vehicle configuration $c \in C(t_1) \cap C(t_2)$. Then we
apply Rule 1 to check if this connection is possible. If so, we add a hyperarc $a$
to the arc set $A$ of our graph $G$. The arc $a$ consists of $|V(t_1,c)| = |V(t_2,c)|$
links. Each link $(v, w) \in a$ with $v \in V(t_1, c)$ and $w \in V(t_2, c)$ connects nodes
with the same vehicle group and multiplicity and so $a$ is well-defined.

**Step 2.** Regular hyperarcs are conjunctions of configuration conserving arcs
as introduced in Step 1. For each tail train $t_1 \in \mathcal{T}$, head train $t_2 \in \mathcal{T}$, vehicle
configuration $c \in C$, and number of overnights $a \in \{0, \ldots, 6\}$ we create a
regular hyperarc as follows. We collect all arcs $a \subseteq A$ connecting $t_1 \in t_1$ and
$t_2 \in t_2$ with configuration $c$, such that midnight is passed $a$ times if one waits
from the arrival date of $t_1$ until the departure date of $t_2$. The set a can be seen
as maximal ”hyper-connection”’ of $t_1$ and $t_2$ with configuration $c$. In the non-
trivial case, i.e., $|a| \geq 2$, we add a regular hyperarc $a = \{(u,v) \in V \times V | \exists a^* \in
a : (u,v) \in a^*\}$ to the arc set $A$ of our graph $G$.

**Step 3.** The last step constructs all arcs that implement a coupling activity,
called coupling arcs. We apply Rule 2 to all links $(v, w) \in V \times V$ having the
same vehicle group and which have not been considered in Step 1. If the link
$(v, w)$ fulfills Rule 2 we add a simple arc $a = \{(v, w)\}$ to the arc set $A$ of our
graph $G$.

**Example 2.** Figure 1 gives an example of our construction of regular hyper-
arcs. It shows two trains $t_1, t_2 \in \mathcal{T}$ connected by configuration conserving arcs $a_1, \ldots, a_7 \in A$ and regular hyperarc $a_7 \in A$. For the sake of simplicity all nodes have only trivial configurations.
Let \( a \in A \) be an arc of \( G \) with vehicle configuration \( c(a) \in C \). The deadhead distance of \( a \) is denoted by \( l(a) \in \mathbb{Q}^+ \). Let \( v(a) \in \mathbb{Q}^+ \) be the duration of the tail trip of \( a \) plus the duration from the arrival of the tail trip of \( a \) to the departure of the head trip of \( a \) divided by the duration of the standard week. Thus \( v(a) \) is the fractional number of physical vehicles "consumed" by \( a \).

For example, if the tail trip of \( a \) departs on Monday at 12 p.m., arrives on Monday at 18 p.m., and the head trip of \( a \) departs on Tuesday at 12 p.m., we have \( v(a) = \frac{1}{7} \). Note that \( v(a) \) can be greater than one if the departure of the head trip of \( a \) is between the departure and arrival of the tail trip of \( a \).

If \( a \) is a coupling arc then \( p(a) \in \mathbb{Q}^+ \) is a constant penalty for the involved coupling activities, otherwise \( p(a) \) is zero. Finally, if \( a \) is not a regular arc \( r(a) \in \mathbb{Q}^+ \) is a constant penalty for violating regularity. In case of a regular arc \( r(a) \) is zero. The objective function \( c : A \mapsto \mathbb{Q}^+ \) is defined as follows:

\[
c(a) := c_a := \underbrace{r(a)}_{(ir-)regularities} + \underbrace{p(a)}_{couplings} + \underbrace{c(c(a)) \cdot l(a)}_{deadheads} + \sum_{f \in c(a)} m(f, c(a)) \cdot c(f) \cdot v(a). \]

As denoted above, the multi-objective function, which minimizes vehicle cost, minimizes deadhead cost, minimizes coupling cost, and maximizes regularity is combined in a single objective function \( c \).
3.2 Integer Programming formulation

Let \( G = (V, H, A) \) be a hypergraph modeling the \( \text{VRP} \) as described above. We introduce binary decision variables \( x_a \in \{0, 1\} \) and \( y_h \in \{0, 1\} \) for each hyperarc \( a \in A \) and each hypernode \( h \in H \) of \( G \). Those variables take value one if the corresponding nodes and hyperarcs are used in the vehicle rotation plan and otherwise zero. The set of all hypernodes \( h \in H \) for trip \( t \in T \) is denoted by \( H(t) \) and \( H(v) \) denotes the set of all hypernodes of \( G \) containing \( v \). By definition, the set \( H(v) \) for \( v \in V \) has cardinality one. The set of all ingoing hyperarcs of \( v \in V \) is defined as \( \delta^{\text{in}}(v) := \{ a \in A \mid \exists (u, w) \in a : w = v \} \subseteq A \), in the same way \( \delta^{\text{out}}(v) := \{ a \in A \mid \exists (u, w) \in a : u = v \} \subseteq A \) denotes the set of all outgoing hyperarcs of \( v \).

Our hyperflow based Integer Programming formulation states:

\[
\min \sum_{a \in A} c_a x_a \quad \text{(HFIP)}
\]

\[
\sum_{h \in H(t)} y_h = 1, \quad \forall t \in T \quad \text{(covering)}
\]

\[
\sum_{a \in \delta^{\text{in}}(v)} x_a - \sum_{h \in H(v)} y_h = 0, \quad \forall v \in V \quad \text{(in-flow)}
\]

\[
\sum_{a \in \delta^{\text{out}}(v)} x_a - \sum_{h \in H(v)} y_h = 0, \quad \forall v \in V \quad \text{(out-flow)}
\]

\[
x_a \in \{0, 1\}, \quad \forall a \in A
\]

\[
y_h \in \{0, 1\}, \quad \forall h \in H.
\]

Our objective function minimizes the total cost. The covering constraints assign one hypernode of graph \( G \) to each trip of the \( \text{VRP} \). This models the configuration assignment of vehicle configurations to trips. Constraints in-flow and out-flow can be seen as flow conservation constraints for each node \( v \in V \). If one interprets an in-flow equation as a departure and the out-flow equation as an arrival node, a hypernode \( h \in H \) can be even seen as a hyperarc between these departure and arrival nodes. With this interpretation the in-flow and out-flow constraints become constraints conserving hyperflow on the trips and connections between trips.

**Example 3.** Figure 2 shows a part of our hypergraph. The set of nodes is \( V = \{v_1, \ldots, v_{18}\} \). The pair of red and blue circles for each \( v \in V \) indicates the in-flow and out-flow accordingly the departure and arrival of a node. The colors of the circles indicating two vehicle groups - a red and a blue one. The set of hypernodes is \( H = \{h_1, \ldots, h_{11}\} \). Trips \( t_1 \in T \) and \( t_2 \in T \) have both two trivial and two non-trivial configurations, trip \( t_3 \in T \) has only one possible non-trivial configuration. Arc \( a_1 \in A \) implements a coupling activity after the
arrival of $t_1$. The hyperarcs $a_2, a_3, a_4, a_5 \in A$ are configuration conserving hyperarcs.

Note that the pure row representation of model HFIP does not directly involve any vehicle composition or regularity requirements. This is because vehicle composition and regularity is solely modeled by the underlying hypergraph. Thus, the main aspects of the VRP are modeled by columns.

4 Solving the Vehicle Rotation Planning Problem

In case of only trivial configurations and without regular hyperarcs the hypergraph is a standard graph. In this case our problem reduces to the Integer Multi-Commodity-Flow problem, which is known to be NP-hard, see [7]. Furthermore, if all trip configurations are fixed, problem VRP is a simple assignment problem and hence an optimal solution of the LP relaxation of model HFIP is already integral.

Due to the NP-hardness of problem VRP, we propose in this section a heuristic Integer Programming approach to solve model HFIP. We are mainly utilizing two general techniques.

First we use a column generation approach to solve the LP-relaxation of model HFIP. Note, that the number of variables is very large, i.e., one for each hyperarc and hypernode. We start with all rows of model HFIP and add all $y$-variables and a few $x$-variables representing arcs with a duration from the departure to the arrival smaller or equal 90 Minutes in advance. The remaining pricing problem is to decide whether there is a hyperarc left with negative
reduced cost – we simply answer this question by enumeration. The best outgoing arc of each node \( v \in V \) and the best outgoing arc of each hypernode \( h \in H \) with negative reduced cost are priced in each column generation round. Furthermore, this allows us to compute in each column or arc generation round a valid global lower bound.

Second, we apply the rapid branching method introduced in [9] and [3] for integrated vehicle and duty scheduling in public transport and for railway track allocation to produce high quality integral solutions. We adapt this heuristic to consider only a subset of the variables – in our case the \( y \)-variables for the hypernodes assigning the vehicle configurations to the trips. The reason is the observation that the model is almost integral and rather easy to solve if the configurations for the trips are fixed.

After the arc generation and rapid branching we use CPLEX 12.2 to solve the generated model so far, i.e., a restricted variant of model HFIP, as a static IP. By means of this approach we can provide valid global lower bounds, as well as high quality solutions as we will see in the next section.

5 Computational Results

We tested the hypergraph based model HFIP and our algorithmic approach on a large set of real world instances that are provided by our project partner DB Fernverkehr. The problem set contains small and rather easy instances, e.g., instance vrp019 and vrp028 with only 8 trains, as well as very large scale ones, e.g., instance vrp011 and vrp014 with more than 24 million hyperarcs. We consider instances for the current operated high speed intercity vehicles (ICE) of DB Fernverkehr as well as instances of conceptional studies for future rolling stock fleets. Today, there are some fleets in operation that can not be coupled on the fly and some of the conceptional studies also consider only scenarios with trivial configurations. Therefore half of the instances contain only trivial configurations. Those instances with non-trivial configurations contain up to 19 configurations of 10 vehicle groups. However, most of them do not contain as many as this. This is because a vehicle group represents a whole traction with engine car and passenger wagons and only a few of them can be coupled together to ensure some constraints about the length of the passenger platform. Note that due to the regularity requirements an instance with only trivial configurations does not reduce to an other problem class.

Table 1 gives some statistics on the number of trains \( |\Sigma| \), the number of vehicle groups \( |F| \), and the number of vehicle configurations \( |C| \). In addition, the number of nodes \( |V| \) and the total number of hyperarcs \( |A| \) of the hypergraphs associated with model VRP are listed. The number of regular arcs constructed in Step 2 is denoted by \( |A_r| \). Column \( |H| \) gives the number of hypernodes. In case of only trivial configurations this number equals \( |V| \), otherwise it has to
be smaller because $H$ is a partition of $V$.

All our computations were performed on computers with an Intel Core 2 Extreme CPU X9650 with 3 GHz, 6 MB cache, and 16 GB of RAM. CPLEX Barrier was running with 4 threads as well as the CPLEX MIP solver. We were able to solve all 31 instances to nearly optimality by the solution approach presented in Section 4. Table 2 shows the detailed results, i.e., the number of vehicles $v$ to operate the $|\Sigma|$ trains, the total objective value of the solutions, the optimality gap\(^3\), and the total running time in seconds. We marked 5 instances which are solved to proven optimality. Except for instance vrp005 the gap is considerably below 1%. This demonstrates that our solution approach can be used to produce high quality solutions for large-scale vehicle rotation planning problems.

### 6 Conclusions

We proposed a novel model for the integrated optimization of vehicle rotations, vehicle compositions, and regularity requirements in long distance railway passenger transport. Our main contribution is a new hypergraph based IP formulation that is able to handle challenging technical requirements of railway optimization in a very compact model. We introduced an associated large-scale method to solve the model and we showed that the overall approach

\[^3\]The relative gap is defined between the best integer objective $UB$ and the objective of the best lower bound $LB$ as $100 \cdot \frac{UB-LB}{UB+10^{-18}}$. 

---

| Test Case | $|\Sigma|$ | $|V|$ | $|H|$ | $|A|$ |
|----------|----------|----------|----------|----------|
| vrp001   | 410      | 8        | 10913    | 10913    |
| vrp002   | 61       | 1        | 310      | 310      |
| vrp003   | 288      | 6        | 2433     | 2038     |
| vrp004   | 298      | 6        | 7379     | 7379     |
| vrp005   | 298      | 24       | 26396    | 26396    |
| vrp006   | 298      | 2        | 2753     | 2753     |
| vrp007   | 298      | 8        | 9896     | 9896     |
| vrp008   | 298      | 18       | 7474     | 7474     |
| vrp009   | 298      | 8        | 3619     | 3619     |
| vrp010   | 298      | 7        | 2913     | 2913     |
| vrp011   | 443      | 16       | 13538    | 13538    |
| vrp012   | 443      | 16       | 9275     | 9275     |
| vrp013   | 252      | 1        | 406      | 406      |
| vrp014   | 443      | 24       | 20124    | 20124    |
| vrp015   | 19       | 4        | 534      | 387      |
| vrp016   | 19       | 4        | 534      | 387      |
| vrp017   | 19       | 2        | 534      | 387      |
| vrp018   | 11       | 4        | 321      | 232      |
| vrp019   | 8        | 4        | 288      | 204      |
| vrp020   | 19       | 4        | 534      | 387      |
| vrp021   | 61       | 1        | 310      | 310      |
| vrp022   | 288      | 6        | 2435     | 2040     |
| vrp023   | 137      | 7        | 2373     | 1815     |
| vrp024   | 19       | 5        | 486      | 360      |
| vrp025   | 19       | 2        | 486      | 360      |
| vrp026   | 11       | 5        | 305      | 224      |
| vrp027   | 8        | 5        | 270      | 196      |
| vrp028   | 19       | 5        | 486      | 360      |
| vrp029   | 556      | 19       | 6145     | 4753     |
| vrp030   | 135      | 6        | 1848     | 1288     |

Tab. 1: Characteristics of the VRP test instances.
can be used to produce near optimal and sometimes proven optimal solutions for large-scale real world problem instances of our cooperation partner DB Fernverkehr.

In the near future, we must calibrate the regularity part of the model in a way that is most useful in practice. Many possible variants of our regularity approach must be considered, varying the cost for regularity and alternatives for "partial regularity". At present it has already become clear that ignoring regularity leads to solutions that are not accepted by the practitioners. In the long run, we have to integrate further constraints and optimization goals, e.g., maintenance and robustness.

References


