Georg-August-Universität Göttingen

Algorithms for delay management with capacity constraints

Michael Schachtebeck, Anita Schöbel

Nr. 2007-15

Preprint-Serie des
Instituts für Numerische und Angewandte Mathematik
Lotzestr. 16-18
D - 37083 Göttingen

Algorithms for Delay Management with Capacity Constraints *

Michael Schachtebeck Anita Schöbel

August 5, 2007

Delay management is an important issue in the daily operations of any railway company. The task is to decide which connections should be maintained in case of delays and to update the planned timetable to a *disposition timetable* in such a way that the inconvenience for the passengers is as small as possible.

In a railway setting, a crucial constraint is the limited capacity of the track system. It has not been treated in optimization approaches so far. In the current paper, we add the capacity constraints to the delay management formulation. Different solution algorithms are suggested and evaluated both from a theoretical and a numerical point of view. In our case study, we use the railway network in the region of Harz, Germany.

1 Introduction

The delay management problem deals with (small) source delays of a railway system as they occur in the daily operational business of any public transportation company. In case of such delays, the planned timetable is not feasible any more. The main question which is considered in the (pure) delay management problem is to decide which trains should wait for delayed feeder trains and which trains better depart on time (*wait-depart decisions*). If these decisions have been made, one furthermore needs to update the original timetable to obtain a so-called *disposition timetable*. A first integer programming formulation for the (pure) delay management problem has been given in [Sch01] and has been further developed in [GHL06, Sch07], see also [Sch06b] for an overview about various models. The complexity of the problem has been investigated in [GJPS05, GGJ+04], where it turns out that the problem is NP hard even in very special cases. Other publications about delay management include an online-approach ([GJPW07]), a model in the context of max-plus-algebra ([RdVM98, Gov98]), a formulation as discrete time-cost tradeoff problem ([GS07]) and simulation approaches ([SM99, SMBG01])

Most of these studies neglect the limited capacity of the track system while dealing with delay management. Adding these constraints, the problem becomes significantly harder to solve.

*This work was partially supported by the Future and Emerging Technologies Unit of EC (IST priority - 6th FP), under contract no. FP6-021235-2 (project ARRIVAL).

Some first ideas on how to model these constraints in the context of delay management have been presented in [Sch06a]. Capacity constraints are also taken into account in a real-world application studied within the project *DisKon* supported by Deutsche Bahn (see [BGJ+05]). Here the delay management problem is solved at two different levels of detail: A macroscopic approach deals with the wait-depart decisions while a microscopic model schedules the trains on the tracks and hence changes the macroscopic solution until it is feasible.

Another line of research is to design delay-resistant timetables, i.e. timetables in which source delays will usually not spread out too badly, see [LSS+07] and references therein. In this paper, different timetables have been evaluated and discussed under optimal delay management policies.

In the following we will present a model and four approaches dealing with the *capacitated* delay management problem, i.e. our goal is to find wait-depart decisions and a disposition timetable, respecting the limited capacity of the track system. We tested our approaches numerically within a case study and report on their behavior.

The remainder of the paper is structured as follows. In Section 2 we present a model for the capacitated delay management problem. Our four approaches are described in Section 3. A theoretical analysis of the approaches is developed in Section 4, while the numerical results are presented in Section 5. We conclude the paper mentioning ideas for further research.

2 Models

The (pure) delay management problem is defined as follows: Given the public transportation network $PTN = (V, E)$ (consisting of the set V of stations and the set E of direct links between stations), the set $\mathcal{F}$ of trains, the set $\mathcal{C} \subset \mathcal{F} \times \mathcal{F} \times V$ of connections and some source delays, decide which connections should be maintained and which connections should be dropped such that the average delay of a passenger at his final destination is minimal. This problem was first introduced in [Sch01]. If we take into account the limited capacity of the tracks, we get the delay management problem with capacity constraints, see [Sch06a]. In this section, we show how this problem can be modeled as a directed graph and as an integer program.

At first, we show how to model the delay management problem with capacity constraints as an event-activity network. The event-activity network is a directed graph $\mathcal{N} = (\mathcal{E}, \mathcal{A})$ whose vertices are events and whose edges are activities. Events are

- (g, u, arr): the arrival of a train $g \in \mathcal{F}$ at a station $u \in V$ and
- (h, v, dep): the departure of a train $h \in \mathcal{F}$ from a station $v \in V$.

By $\mathcal{E}_{arr}$ and $\mathcal{E}_{dep}$, we denote the set of all arrival and all departure events, respectively. Then, $\mathcal{E} = \mathcal{E}_{arr} \cup \mathcal{E}_{dep}$. The set of all activities $\mathcal{A} = \mathcal{A}_{drive} \cup \mathcal{A}_{wait} \cup A_{change} \cup \mathcal{A}_{head}$ consists of four different types of activities:

- driving activities $\mathcal{A}_{drive} \subset \mathcal{E}_{dep} \times \mathcal{E}_{arr}$ model the driving of a train between two consecutive stations (including turn-around edges when the vehicle has ended a line and is about to start a new one),

- waiting activities $\mathcal{A}_{wait} \subset \mathcal{E}_{arr} \times \mathcal{E}_{dep}$ represent the waiting of a train in a station to let passengers board and de-board,
- changing activities $\mathcal{A}_{change} \subset \mathcal{E}_{arr} \times \mathcal{E}_{dep}$ allow passengers to transfer from one train to another one.

Up to now, these are the constraints of the pure delay management problem. If two events $i, j \in \mathcal{E}$ are connected by an activity $(i, j) \in \mathcal{A}$, then event i has to be performed before event j can take place. To get the delay management problem with capacity constraints, we add

- headway activities $\mathcal{A}_{head} \subset \mathcal{E}_{dep} \times \mathcal{E}_{dep}$ which model the limited capacity of the track system (security distances for trains driving into the same direction or for oncoming trains on single-way tracks).

Headway activities model disjunctive constraints: if $(i, j) \in \mathcal{A}_{head}$, then $(j, i) \in \mathcal{A}_{head}$, too, and for each such pair of headway activities, exactly one of them has to be respected.

Now we show how to model the delay management problem with capacity constraints as an integer program. By L_a, we denote the minimal time needed to perform activity $a \in \mathcal{A}$, and h_{ij} is the minimal buffer time we have to respect if event i is performed before event j and if $(i, j) \in \mathcal{A}_{head}$. We define w_i as the number of passengers getting on or off at event $i \in \mathcal{E}$ and w_a as the number of passengers who want to use a connection $a \in \mathcal{A}_{change}$. The (original) timetable is a mapping $\Pi : \mathcal{E} \to \mathbb{N}, i \mapsto \Pi_i$, assigning a time to each event. However, if some event $i \in \mathcal{E}$ (or some activity $a \in \mathcal{A}$) has a source delay d_i (or d_a, respectively), we have to change the original timetable to determine a *disposition timetable* $x : \mathcal{E} \to \mathbb{N}, i \mapsto x_i$. Furthermore, we assume that all lines have a common period T and that all trains are on time in the next period.

To model the wait-depart decisions, we introduce binary variables

$$z_a := \begin{cases} 0 & \text{if changing activity } a \text{ is maintained} \\ 1 & \text{otherwise} \end{cases}$$

for all changing activities $a \in \mathcal{A}_{change}$. To take into account the capacity constraints, we introduce binary variables

$$g_{ij} := \begin{cases} 0 & \text{if event } i \text{ is performed before event } j \\ 1 & \text{otherwise} \end{cases}$$

for all headway activities $(i, j) \in \mathcal{A}_{head}$. Then, the integer programming formulation reads as follows:

$$(\textbf{Cap-DM}) \ \min f(x, z) = \sum_{i \in \mathcal{E}_{arr}} w_i (x_i - \Pi_i) + \sum_{a \in \mathcal{A}_{change}} z_a w_a T \tag{1}$$

such that

$$\begin{aligned}
x_i &\geq \Pi_i + d_i \quad \forall i \in \mathcal{E} && (2)\\
x_j - x_i &\geq L_a + d_a \quad \forall a = (i,j) \in \mathcal{A}_{nice} && (3)\\
Mz_a + x_j - x_i &\geq L_a \quad \forall a = (i,j) \in \mathcal{A}_{change} && (4)\\
Mg_{ij} + x_j - x_i &\geq h_{ij} \quad \forall (i,j) \in \mathcal{A}_{head} && (5)\\
x_i &\in \mathbb{N} \quad \forall i \in \mathcal{E} && (6)\\
z_a &\in \{0,1\} \quad \forall a \in \mathcal{A}_{change} && (7)\\
g_{ij} &\in \{0,1\} \quad \forall (i,j) \in \mathcal{A}_{head} && (8)\\
g_{ij} + g_{ji} &= 1 \quad \forall (i,j) \in \mathcal{A}_{head} && (9)
\end{aligned}$$

with $\mathcal{A}_{nice} := \mathcal{A}_{wait} \cup \mathcal{A}_{drive}$ and M large enough.
The objective we minimize is the sum of all delays passengers have when starting their trips or at their final destinations plus the sum of all missed connections. It approximates the sum of all delays over all customers, and in some cases coincides with it (see [Sch07]). Furthermore, any optimal solution of this program is a Pareto solution with respect to the two objective functions *minimize the delay over all vehicles* and *minimize the number of missed connections.* Constraint (3) makes sure that the delay is passed on correctly along driving and waiting activities, and (4) and (5) do the same for changing and headway activities. Constraint (9) ensures that exactly one of each pair of headway constraints is respected in a feasible solution. Note that we allow two types of source delays: The first is a delay d_i at an event (e.g. a driver coming too late to his duty), which refers to a fixed point of time, such that $x_i \geq \Pi_i + d_i$ is required. The second is a delay d_a of an activity, e.g. an increase of traveling time between two stations due to construction work. It has to be added to the original planned duration L_a of activity a as done in (3).

If we neglect all constraints modeling the limited capacity of the tracks (i.e. constraints (5), (8) and (9)), we have the pure delay management problem:

$$(\textbf{DM}) \ \min f(x,z) = \sum_{i \in \mathcal{E}_{arr}} w_i(x_i - \Pi_i) + \sum_{a \in \mathcal{A}_{change}} z_a w_a T$$

such that (2), (3), (4), (6), (7) are satisfied.

If we furthermore ignore all constraints modeling the connections (i.e. constraints (4) and (7)), set $\mathcal{A}_{nice} := \mathcal{A}_{wait} \cup \mathcal{A}_{drive} \cup \{a \in \mathcal{A}_{change} : a \text{ should be maintained}\}$ and delete the constant term in the objective, (Cap-DM) reduces to a re-scheduling problem:

$$(\textbf{Re-Sched}) \ \min f(z) = \sum_{i \in \mathcal{E}_{arr}} w_i(x_i - \Pi_i)$$

such that (2), (3), (6) are satisfied.

If we set $\mathcal{A}_{nice} := \mathcal{A}_{wait} \cup \mathcal{A}_{drive} \cup \{a \in \mathcal{A}_{change} : a \text{ should be maintained}\}$ and ignore all constraints modeling the connections, but take into account the limited capacity of the track system, we have a re-scheduling problem with capacity constraints:

$$(\textbf{Cap-Re-Sched}) \ \min f(z) = \sum_{i \in \mathcal{E}_{arr}} w_i(x_i - \Pi_i)$$

such that (2), (3), (5), (6), (8) and (9) are satisfied.

(Cap-Re-Sched) has been shown to be NP complete in [CS07]. Since (Cap-Re-Sched) is a special case of (Cap-DM), namely with $\mathcal{A}_{change} = \emptyset$, also (Cap-DM) is NP complete.

Note that the pure delay management problem in its original path-based formulation has been shown to be NP hard in [GJPS05], but the complexity status of (DM) is not clear yet.

If all wait-depart decisions are already made and if the order of trains is fixed, we define $\mathcal{A}_{nice} := \mathcal{A}_{wait} \cup \mathcal{A}_{drive} \cup \{a \in \mathcal{A}_{change} : z_a = 0\} \cup \{(i,j) \in \mathcal{A}_{head} : g_{ij} = 0\}$ and have an instance of (Re-Sched) – this problem can be solved in polynomial time, e.g. by applying the forward phase of the critical path method (see [Sch06b]).

3 Heuristics

The idea of the first heuristic, "first scheduled, first served" (FSFS), is to fix the order of trains according to the original timetable Π. Doing so, the number of variables and the number of constraints in the integer programming formulation is reduced dramatically.

"First scheduled, first served" (FSFS):

1. Fix the order of trains according to the original timetable Π:

   For each $a = (i,j) \in \mathcal{A}_{head}$:

   If $\Pi_i \leq \Pi_j$ set $g_{ij} = 0, g_{ji} = 1$.

2. Set $\mathcal{A}_{nice} := \mathcal{A}_{drive} \cup \mathcal{A}_{wait} \cup \{(i,j) \in \mathcal{A}_{head} : g_{ij} = 0\}$.

3. Compute the exact solution of the corresponding instance of (DM).

The next heuristic, "first rescheduled, first served" (FRFS), only differs from FSFS in the way how the order of trains is fixed. Instead of using the original timetable, we solve the problem without capacity constraints (the pure problem) in a first step and use the resulting disposition timetable x to fix the order of trains.

"First rescheduled, first served" (FRFS):

1. Compute the exact solution of the problem (DM), ignoring constraints (5), (8) and (9) from the (Cap-DM) formulation $\Rightarrow$ disposition timetable x.

2. Fix the order of trains according to the disposition timetable x:

   For each $a = (i,j) \in \mathcal{A}_{head}$:

   If $x_i \leq x_j$ set $g_{ij} = 0, g_{ji} = 1$.

3. Set $\mathcal{A}_{nice} := \mathcal{A}_{drive} \cup \mathcal{A}_{wait} \cup \{(i,j) \in \mathcal{A}_{head} : g_{ij} = 0\}$.

4. Compute the exact solution of the corresponding instance of (DM).

FRFS has one advantage and one disadvantage compared to FSFS. The advantage is that a relaxation is solved in the first step, so we get a lower bound on the objective value of the optimal solution. The drawback is its running time, resulting from solving (DM) in the first

and in the last step. This observation leads to the third heuristic, "first rescheduled, first served with early connection fixing" (EARLYFIX). As we already did in FRFS, we fix the order of trains according to the disposition timetable that we get from the relaxed problem. Instead of computing an exact solution of the remaining problem then, we also keep the wait-depart decisions from the solution of the relaxation and compute the new disposition timetable by applying the critical path method to the remaining problem:

"First rescheduled, first served with early connection fixing" (EARLYFIX):

1. Compute the exact solution of the problem (DM), ignoring constraints (5), (8) and (9) from the (Cap-DM) formulation $\Rightarrow$ disposition timetable x_i, decision variables z_a.
2. Fix the order of trains according to the disposition timetable x:

   For each $a = (i, j) \in \mathcal{A}_{head}$:

   If $x_i \leq x_j$ set $g_{ij} = 0, g_{ji} = 1$.
3. Set $\mathcal{A}_{nice} := \mathcal{A}_{drive} \cup \mathcal{A}_{wait} \cup \{a \in \mathcal{A}_{change} : z_a = 0\} \cup \{(i, j) \in \mathcal{A}_{head} : g_{ij} = 0\}$.
4. Compute the exact solution of the corresponding instance of (Re-Sched).

In contrast to FRFS, Step 4 can now be solved very efficiently. The last heuristic we present is one that has polynomial runtime and does not need to solve any NP hard problem. It is called "first scheduled, first served with priority-based fixing" (FSFS-PBF) and is a modification of the FSFS heuristic we presented first. As we do in FSFS, we fix the order of trains according to the original timetable Π, but instead of solving the remaining problem exactly, we use a heuristic approach to make the wait-depart decisions. The idea is to maintain only the "most important" connections and do not care about the less important ones.

"First scheduled, first served with priority-based fixing" (FSFS-PBF):

1. Fix the order of trains according to the original timetable Π:

   For each $a = (i, j) \in \mathcal{A}_{head}$:

   If $\Pi_i \leq \Pi_j$ set $g_{ij} = 0, g_{ji} = 1$.
2. Maintain the "most important" connections:
   - Sort the changing edges in descending order according to their weights w_a.
   - Set $z_a = 0$ for the first $k\%$ of the connections.
3. Set $\mathcal{A}_{nice} := \mathcal{A}_{drive} \cup \mathcal{A}_{wait} \cup \{a \in \mathcal{A}_{change} : z_a = 0\} \cup \{(i, j) \in \mathcal{A}_{head} : g_{ij} = 0\}$.
4. Compute the exact solution of the corresponding instance of (Re-Sched).

4 Theoretical Results

At first, we will compare the objective values of the four heuristics.

Lemma 1. *If we denote by F^{DM} the objective value of the optimal solution of an instance of (DM) corresponding to an instance of (Cap-DM) with objective value F^{OPT}, the following holds:*

$$F^{\mathrm{DM}} \leq F^{\mathrm{OPT}}.$$

Proof. (DM) is a relaxation of (Cap-DM). □

Since the first step in both FRFS and in EARLYFIX is to solve (DM), we obtain from these heuristics not only an approximation of the optimal solution, but also a lower bound on its objective value, and hence their absolute errors can be bounded a posteriori by

$$F^{\mathrm{FRFS}} - F^{\mathrm{DM}} \text{ and } F^{\mathrm{EARLYFIX}} - F^{\mathrm{DM}},$$

respectively, where F^{FRFS} and F^{EARLYFIX} are the objective values of FRFS and EARLYFIX.

Lemma 2. *Let $F^{\mathrm{HEURISTIC}}$ denote the objective value of the solution of a fixed instance of (Cap-DM) computed by heuristic HEURISTIC $\in$ {FSFS, FSFS-PBF, FRFS, EARLYFIX}, and F^{OPT} the objective value of the optimal solution of this instance. Then,*

$$F^{\mathrm{OPT}} \leq F^{\mathrm{FSFS}} \leq F^{\mathrm{FSFS-PBF}}$$

and

$$F^{\mathrm{OPT}} \leq F^{\mathrm{FRFS}} \leq F^{\mathrm{EARLYFIX}}.$$

Proof. FSFS and FSFS-PBF fix the order of trains in exactly the same way. However, after fixing the order of trains, FSFS computes an exact solution of the remaining problem (thus making optimal wait-depart decisions for the remaining problem), while the wait-depart decisions in FSFS-PBF are gained by a heuristic approach.

FRFS and EARLYFIX also fix the order of trains in exactly the same way. Like FSFS, FRFS computes optimal wait-depart decisions for the remaining problem, while EARLYFIX keeps the wait-depart decisions from the solution of the relaxed problem. □

Now, we will give some results on the relative error of the heuristics. Unfortunately, we can prove that the results of all heuristics might get arbitrarily bad compared to the optimal solution:

Theorem 3. *For each heuristic that solves (Cap-DM), fixing the g_{ij} variables as they are set in the original timetable (i.e. according to step 1 in FSFS), the following holds: For each $k \in \mathbb{N}$, there exists an instance of (Cap-DM) such that*

$$\frac{F^{\mathrm{HEURISTIC}} - F^{\mathrm{OPT}}}{F^{\mathrm{OPT}}} > k$$

where $F^{\mathrm{HEURISTIC}}$ is the objective values of the heuristic.

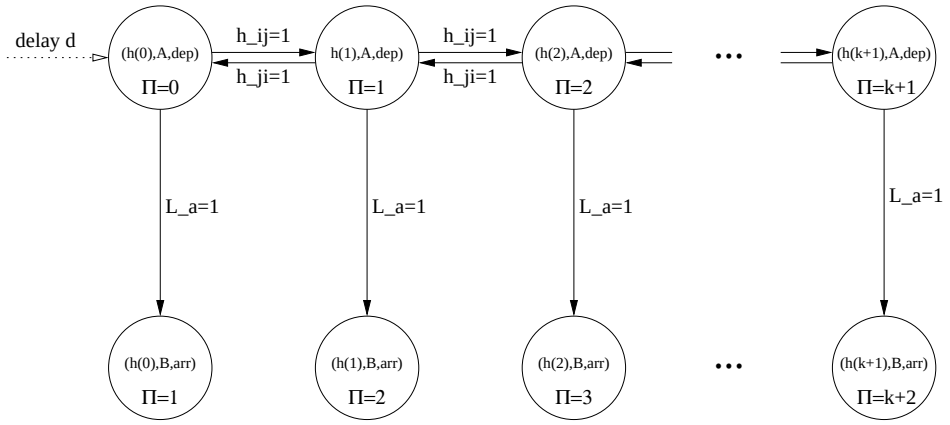

Figure 1: Event-activity network for the proof of theorem 3.

Proof. Let $k \in \mathbb{N}$. Assume that we have two stations A and B and $k+2$ trains $h_0, h_1, \ldots, h_{k+1}$. All trains drive from station A to station B. In the original timetable, the trains leave station A in the order $h_0, h_1, \ldots, h_{k+1}$ at the times $\Pi(h_i, A, dep) = i$ and arrive at station B at the times $\Pi(h_i, B, arr) = i + 1, \quad i \in \{0, \ldots, k+1\}$. For each $i \in \{0, \ldots, k\}$, the departure of train h_i and the departure of train h_{i+1} are connected by a pair of headway edges. All weights and all lower bounds are set to 1. The resulting event-activity network is shown in Figure 1.

Now, assume that (h_0, A, dep) is delayed by $d \geq k + 2$. In the optimal solution, the trains $h_1, \ldots, h_{k+1}$ leave and arrive on time, while train h_0 has a delay of d, so $F^{\mathrm{OPT}} = d$. If we solve the problem by a heuristic that sets the g_{ij} variables as they are set in the original timetable without delays, all trains get a delay of at least d, so $F^{\mathrm{HEURISTIC}} \geq (k + 2) \cdot d$, hence

$$\frac{F^{\mathrm{HEURISTIC}} - F^{\mathrm{OPT}}}{F^{\mathrm{OPT}}} \;\geq\; \frac{(k+2) \cdot d - d}{d} \;=\; k + 1 \;>\; k.$$

□

Theorem 4. *For each heuristic that solves (Cap-DM), fixing the g_{ij} variables as they are set in the optimal solution of the relaxed problem without capacity constraints (i.e. according to step 2 in FRFS), the following holds: For each $k \in \mathbb{N}$, there exists an instance of (Cap-DM) such that*

$$\frac{F^{\mathrm{HEURISTIC}} - F^{\mathrm{OPT}}}{F^{\mathrm{OPT}}} \;>\; k.$$

Proof. Let $k \in \mathbb{N}$. Assume that we have two stations A and B and two trains g and h. Both trains drive from station A to station B. In the original timetable, the trains leave station A in the order g, h at the times $\Pi(g, A, dep) = 0$ and $\Pi(h, A, dep) = 1$ and arrive at station B at the times $\Pi(g, B, arr) = 1$ and $\Pi(h, B, arr) = 2$. The departures of both trains are connected by a pair of disjunctive headway edges with weights 1 (headway edge from (g, A, dep) to (h, A, dep)) and $4 \cdot (k + 1)$, respectively. All weights and all other lower bounds are set to 1. The resulting event-activity network is shown in Figure 2.

Now, assume that (g, A, dep) is delayed by 2. In the optimal solution, both trains get a delay of 2, so $F^{\mathrm{OPT}} = 4$. In the optimal solution of the relaxation without capacity constraints,

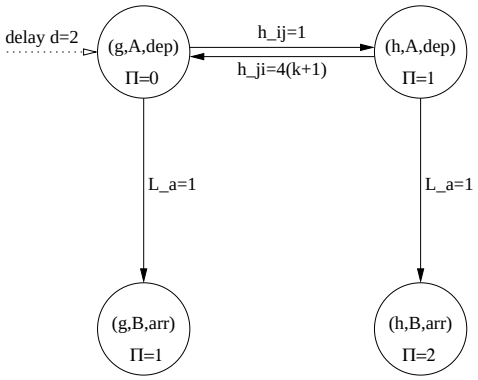

Figure 2: Event-activity network for the proof of theorem 4.

train h departs and arrives on time, and train g has a delay of 2. If the heuristic fixes the g_{ij} variables in this way, it has to respect the headway edge with $h_{ij} = 4 \cdot (k+1)$ in the next step, so train h is on time, while train g has a delay of at least $1 + 4 \cdot (k+1)$, hence

$$\frac{F^{\text{HEURISTIC}} - F^{\text{OPT}}}{F^{\text{OPT}}} \;\geq\; \frac{1 + 4 \cdot (k+1) - 4}{4} \;=\; \frac{1 + 4k}{4} \;>\; k.$$

□

However – as we will show in Section 5 – the heuristics do not behave as bad as one might think regarding the results above.
In the following, we will give an upper bound on the relative error of EARLYFIX.
Given a graph $G = (V, E)$ and a node $v \in V$, we denote by $\mathcal{P}(v)$ the set of all predecessors of v, that is

$$\mathcal{P}(v) \;:=\; \{u \in V\colon \exists\, w_1, \ldots, w_k \in V\colon (u, w_1), (w_1, w_2), \ldots, (w_{k-1}, w_k), (w_k, v) \in E\} \setminus \{v\}.$$

Obviously, we have

$$u \in \mathcal{P}(v) \;\Rightarrow\; \mathcal{P}(u) \subset \mathcal{P}(v). \tag{10}$$

Lemma 5. *Let x^{relax} be an optimal solution of (Re-Sched) with events $\mathcal{E}$ and activities $\mathcal{A}_{nice} = \mathcal{A}_{wait} \cup \mathcal{A}_{drive} \cup \mathcal{A}^{fix}_{change}$ and x^{cap} an optimal solution of (Re-Sched) with events $\mathcal{E}$ and activities $\mathcal{A}_{nice} = \mathcal{A}_{wait} \cup \mathcal{A}_{drive} \cup \mathcal{A}^{fix}_{change} \cup \mathcal{A}^{fix}_{head}$. Let*

$$\begin{aligned} \mathcal{A}^1 &= \left\{a = (i,j) \in \mathcal{A}^{fix}_{head} : \Pi_i \leq \Pi_j\right\} \\ \mathcal{A}^2 &= \left\{a = (i,j) \in \mathcal{A}^{fix}_{head} : \Pi_i > \Pi_j\right\}. \end{aligned}$$

Then, we have

$$x^{cap}_i \;\leq\; x^{relax}_i + \sum_{\substack{(k,l) \in \mathcal{A}^1\colon \\ k \in \mathcal{P}(i)}} (x^{relax}_k - \Pi_k) + \sum_{\substack{(k,l) \in \mathcal{A}^2 \\ k \in \mathcal{P}(i)}} (x^{relax}_k + h_{kl}) \quad \forall i \in \mathcal{E}. \tag{11}$$

Proof. An optimal solution of (Re-Sched) can be computed by applying the critical path method. We assume that $\mathcal{E} = \{1, \ldots, n\}$ is sorted such that for $i < j$, there is no directed path from j to i, and set $\mathcal{E}_1 = \{1\}$, $x_1^{cap} = \Pi_1 + d_1 = x_1^{relax}$. For $k = 2, \ldots, n$, we set $\mathcal{E}_k = \mathcal{E}_{k-1} \cup \{k\}$ and

$$x_k^{cap} = \max \left\{ \Pi_k + d_k, \max_{\substack{a=(j,k) \in \mathcal{A}_{nice}: \\ j \in \mathcal{E}_k}} x_j^{cap} + L_a \right\}.$$

For $k = 1$ and for $x_k^{cap} = \Pi_k + d_k$, (11) obviously is true. In the rest of the proof, we therefore assume $x_k^{cap} > \Pi_k + d_k$ and that (11) is true for all $j < k$. Let

$$\tilde{a} = (j,k) \ := \ \operatorname*{argmax}_{\substack{a=(j,k) \in \mathcal{A}_{nice}: \\ j \in \mathcal{E}_k}} x_j^{cap} + L_a.$$

Case 1: Assume that $\tilde{a} \in \mathcal{A}_{wait} \cup \mathcal{A}_{drive}\mathcal{A}_{change}^{fix}$. Then, $x_k^{cap} = x_j^{cap} + L_{\tilde{a}}$, and using $x_k^{relax} - x_j^{relax} \geq L_{\tilde{a}}$, (11) to estimate x_j^{cap} and (10) which yields $\mathcal{P}(j) \subset \mathcal{P}(k)$, we see that (11) is satisfied.

Case 2: Assume that $\tilde{a} \in \mathcal{A}^1$. Then, $x_k^{cap} = x_j^{cap} + h_{jk}$. Using (11), we get

$$x_k^{cap} \ \leq \ x_k^{relax} + \sum_{\substack{(l,m) \in \mathcal{A}^1: \\ l \in \mathcal{P}(j)}} (x_l^{relax} - \Pi_l) + \sum_{\substack{(l,m) \in \mathcal{A}^2: \\ l \in \mathcal{P}(j)}} (x_l^{relax} + h_{lm}) + x_j^{relax} - x_k^{relax} + h_{jk}. \quad (12)$$

Using $\Pi_k - \Pi_j \geq h_{jk}$, $x_k^{relax} \geq \Pi_k$ and $\mathcal{P}(j) \subset \mathcal{P}(k)$, we see that (11) holds.

Case 3: Assume that $\tilde{a} \in \mathcal{A}^2$. Then, $x_k^{cap} = x_j^{cap} + h_{jk}$. As in the second case, we get inequality (12). We use $x_k^{relax} \geq 0$, move $x_j^{relax} + h_{jk}$ to the second sum and use $\mathcal{P}(j) \subset \mathcal{P}(k)$ to prove the lemma for the third case. □

We can use Lemma 5 to get an upper bound on the relative error of EARLYFIX: We replace x^{relax} by x^{DM} and x^{cap} by $x^{EARLYFIX}$, and define $\mathcal{A}_{head}^{fix} = \left\{ (i,j) \in \mathcal{A}_{head} : x_i^{DM} \leq x_j^{DM} \right\}$. Using the delay $y_i = x_i - \Pi_i$ of event i instead of its time x_i in the disposition timetable, we have

$$y_i^{EARLYFIX} - y_i^{DM} \ \leq \ \sum_{\substack{(k,l) \in \mathcal{A}^1: \\ k \in \mathcal{P}(i)}} y_k^{DM} + \sum_{\substack{(k,l) \in \mathcal{A}^2: \\ k \in \mathcal{P}(i)}} (y_k^{DM} + \Pi_k + h_{kl}) \ \leq \ F^{\mathrm{DM}} + \sum_{(k,l) \in \mathcal{A}_{head}^{fix}} (h_{kl} + \Pi_k),$$

where the second inequality holds if we assume $w_i \geq 1 \forall \ i \in \mathcal{E}$. With this assumption, we hence obtain

$$F^{\mathrm{EARLYFIX}} - F^{\mathrm{OPT}} \ \leq \left(F^{\mathrm{DM}} + \sum_{(k,l) \in \mathcal{A}_{head}^{fix}} (h_{kl} + \Pi_k) \right) \sum_{i \in \mathcal{E}} w_i.$$

If, in addition, $\mathcal{A}^2 = \emptyset$, we have

$$F^{\mathrm{EARLYFIX}} - F^{\mathrm{OPT}} \ \leq \ F^{\mathrm{DM}} \sum_{i \in \mathcal{E}} w_i.$$

This yields the following.

Corollary 6. *If all events $i \in \mathcal{E}$ have weights $w_i \geq 1$ and if the solution x^{DM} of (DM) satisfies*

$$\Pi_i \leq \Pi_j \;\Rightarrow\; x_i^{DM} \leq x_j^{DM} \quad \forall\, (i,j) \in \mathcal{A}_{head}$$

(i.e. we do not change the order of trains compared to the original timetable Π), the following holds:

$$\frac{F^{\text{EARLYFIX}} - F^{\text{OPT}}}{F^{\text{OPT}}} \leq \sum_{i \in \mathcal{E}} w_i.$$

If all events $i \in \mathcal{E}$ have weights $w_i \geq 1$ and if $F^{\text{OPT}} \geq 1$, we have

$$\frac{F^{\text{EARLYFIX}} - F^{\text{OPT}}}{F^{\text{OPT}}} \leq \left(1 + \sum_{(k,l) \in \mathcal{A}^1} (h_{kl} + \Pi_k)\right) \sum_{i \in \mathcal{E}} w_i.$$

The corollary gives rise to the assumption that the quality of the solution depends on the size of $\sum_{(k,l)\in\mathcal{A}^1} h_{kl}$. This is studied in Figure 5.

5 Numerical Results

The numerical results are based on data from the Harz region in the center of Germany. The dataset contains 598 stations, 92 trains (vehicles) and 30 lines, each line with two directions. The sizes of the event-activity network for different observation periods are stated in Table 2. The ILP formulation was solved using Xpress-MP 2006 on a Pentium IV 3 GHz processor with 2 GB RAM. We generated about 600 different delay scenarios; in each of them, we assigned 25 randomly generated source delays of 1-20 minutes to 25 randomly chosen driving and waiting activities.

In Figure 3, we present four histograms of the relative errors for the heuristics FSFS, EARLYFIX and FRFS and for the approach running both FSFS and FRFS and taking the better solution. We took into account all events and all activities that take place during a fixed observation period of 3 hours. On the x-axis we see intervals of 0.1 length describing the relative error. The first interval hence corresponds to a relative error between 0 and 0.1, the second interval to a relative error between 0.1 and 0.2, and so on. We show in how many of the about 600 different delay scenarios the relative error of the respective heuristic takes a value in an interval of length 0.1 – for example, in about 55 scenarios out of 600, the relative error of FSFS is in the interval [0, 0.1].

EARLYFIX and FRFS are almost equal concerning the quality of their solutions, although FRFS is slightly better. For both of them, the number of scenarios with a small relative error is significantly higher than for FSFS. On the other hand, there are some scenarios in which EARLYFIX and FRFS have a very high relative error – FSFS does not have these outliers. If we combine FSFS and FRFS – this means that for each scenario, we take the solution with the smaller objective value – we benefit from the large number of scenarios with a small relative error in FRFS and from the fact that FSFS does not have those outliers as FRFS does have.

Table 1 gives an overview of how good FSFS, EARLYFIX and FRFS are compared to each other. We specify for each heuristic in how many cases it computes a solution at least as good as the solutions gained from the other heuristics. We take into account different observation

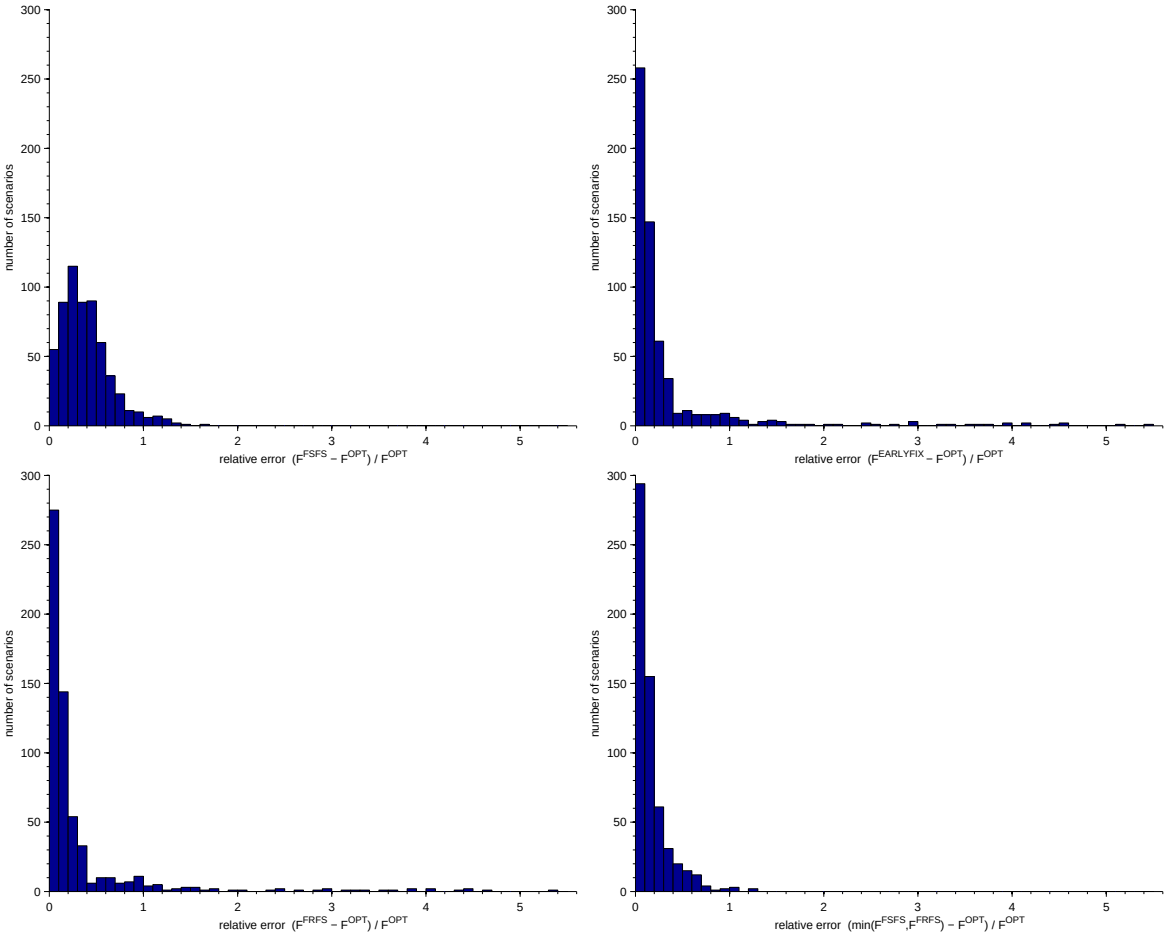

Figure 3: The relative error of FSFS, EARLYFIX, FRFS and the approach of running both FSFS and FRFS and taking the better solution for an observation period of 3 hours (7 104 events, 9 570 activities).

periods. For larger event-activity networks, EARLYFIX performs quite bad, while the number of scenarios in which FSFS computes the best solution grows significantly.

In Figure 4, we show how the relative errors of FSFS, EARLYFIX and FRFS grow with the length of the observation period, i.e. with the size of the event-activity network. The larger the event-activity network, the larger the relative error of all heuristics.

Figure 5 shows the relative error of EARLYFIX. We graphed the relative error (on the y-axis) against the sum of weights of headway activities in $\mathcal{A}^2$. The figure justifies our result that the maximum relative error becomes larger when the sum of headways of activities in $\mathcal{A}^2$ (i.e. headway activities fixed to another direction as in the original timetable) increases.

In Table 2, we finally specify the runtime of the exact solution and of the heuristics FSFS, FRFS and EARLYFIX for different observation periods and therewith for different sizes of the event-activity network. An observation period of k hours means that we considered a part of the event-activity-network containing all events and all activities during a fixed k-hours time slot.

heuristic	observation period of 3 hours	6 hours	10 hours
FSFS	141 (23.58%)	239 (39.97%)	263 (43.98%)
EARLYFIX	219 (36.62%)	83 (13.88%)	75 (12.54%)
FRFS	457 (76.42%)	361 (60.37%)	336 (56.19%)

Table 1: How often (out of 598 different scenarios) is FSFS, EARLYFIX and FRFS at least as good as the two other heuristics, w.r.t different observation periods?

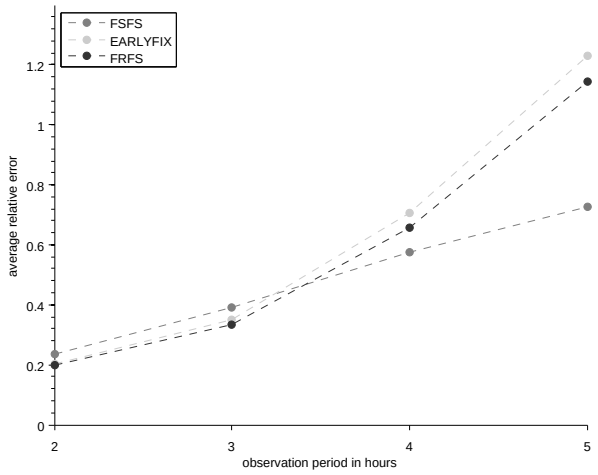

Figure 4: Average relative error of FSFS, EARLYFIX and FRFS for different observation periods between two and five hours.

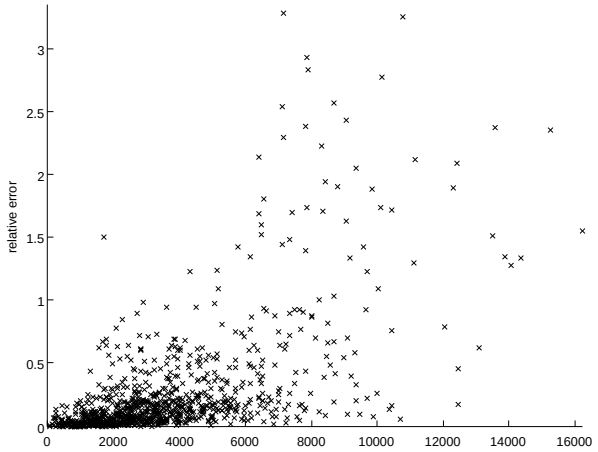

Figure 5: Relative error of EARLYFIX versus $\sum_{(i,j)\in\mathcal{A}_2} h_{ij}$ for an observation period of 4 hours (9 446 events, 14 079 activities).

All heuristics are significantly faster than the optimal solution (calculated via the ILP formulation by Xpress). EARLYFIX clearly outperforms FSFS and FRFS by a factor of 3. FSFS and FRFS are nearly equal considering the computation time.

size of the event-activity network					runtime (in seconds) of algorithm			
hours	$\|\mathcal{E}\|$	$\|\mathcal{A}\|$	$\|\mathcal{A}_{head}\|$	$\|\mathcal{A}_{change}\|$	exact	FSFS	EARLYFIX	FRFS
2	4 726	5 865	1 110	125	19.45	0.24	0.11	0.26
3	7 104	9 570	2 378	187	185.33	0.46	0.17	0.49
4	9 446	14 079	4 428	307	584.66	0.69	0.25	0.74
5	11 824	18 605	6 514	369	1 075.52	0.91	0.31	1.00
6	14 166	23 396	8 846	489	-	1.16	0.39	1.26
8	18 888	32 673	13 260	632	-	1.65	0.53	1.83
10	23 596	41 992	17 656	852	-	2.04	0.68	2.34
15	33 718	61 944	27 138	1 209	-	3.01	1.01	3.63

Table 2: Average runtime for different sizes of the event-activity network.

6 Conclusion and Further Research

In this paper, we suggested and evaluated four different heuristics to deal with the capacitated delay management problem. It turns out that in most cases, the relative errors of FRFS are better than those of FSFS, but the latter has shown to be more resistant against outliers. We conclude that running both of these approaches and taking the better solution seems to be an efficient approach for real-world applications. Other heuristics as a Branch & Bound approach using the features of pure delay management, dynamic programming and a kernel-based learning approach are under research.

While analyzing the heuristics in more detail, we tackled the question on how bad a solution of the pure delay management problem can become when adding capacity constraints. The difference to the solution of the corresponding capacitated problem reflects the *robustness* of a solution of the pure delay management problem. This difference increases when the order of many trains – compared to their order in the original timetable – is changed in the solution of the pure problem. It is ongoing research to increase the robustness of a solution of (DM) by adding just a few crucial headway constraints. An approach how such constraints (or other dependencies) can be identified is described in [CS07].

Another generalization of delay management is to allow to change the vehicle routes if this seems appropriate to reduce delays, and to include the microscopic routes of the trains, in particular at large stations.

Acknowledgment. We want to thank Jens Dupont of *Deutsche Bahn* and Christian Liebchen of TU Berlin for providing the data for the case study.

References

[BGJ+05] N. Bissantz, S. Güttler, J. Jacobs, S. Kurby, T. Schaer, A. Schöbel, and S. Scholl. DisKon - Disposition und Konfliktlösungs-management für die beste Bahn. *Eisenbahntechnische Rundschau (ETR)*, 45(12):809–821, 2005. (in German).

[CS07] C. Conte and A. Schöbel. Identifying dependencies among delays. Technical report, ARRIVAL Report TR-0061, 2007.

[GGJ+04] M. Gatto, B. Glaus, R. Jacob, L. Peeters, and P. Widmayer. Railway delay management: Exploring its algorithmic complexity. In *Proc. 9th Scand. Workshop on Algorithm Theory (SWAT)*, volume 3111 of *LNCS*, pages 199–211, 2004.

[GHL06] L. Giovanni, G. Heilporn, and M. Labbé. Optimization models for the delay management problem in public transportation. *European Journal of Operational Research*, 2006. to appear.

[GJPS05] M. Gatto, R. Jacob, L. Peeters, and A. Schöbel. The computational complexity of delay management. In D. Kratsch, editor, *Graph-Theoretic Concepts in Computer Science: 31st International Workshop (WG 2005)*, volume 3787 of *Lecture Notes in Computer Science*, 2005.

[GJPW07] M. Gatto, R. Jacob, L. Peeters, and P. Widmayer. On-line delay management on a single train line. In *Algorithmic Methods for Railway Optimization*, Lecture Notes in Computer Science. Springer, 2007. to appear.

[Gov98] R.M.P. Goverde. The max-plus algebra approach to railway timetable design. In *Computers in Railways VI: Proceedings of the 6th international conference on computer aided design, manufacture and operations in the railway and other advanced mass transit systems, Lisbon, 1998*, pages 339–350, 1998.

[GS07] A. Ginkel and A. Schöbel. To wait or not to wait? The bicriteria delay management problem in public transportation. *Transportation Science*, 2007. to appear.

[LSS+07] C. Liebchen, M. Schachtebeck, A. Schöbel, S. Stiller, and A. Prigge. Computing delay-resistant railway timetables. working paper, 2007.

[RdVM98] B. De Schutter R. de Vries and B. De Moor. On max-algebraic models for transportation networks. In *Proceedings of the International Workshop on Discrete Event Systems*, pages 457–462, Cagliari, Italy, 1998.

[Sch01] A. Schöbel. A model for the delay management problem based on mixed-integer programming. *Electronic Notes in Theoretical Computer Science*, 50(1), 2001.

[Sch06a] A. Schöbel. Capacity constraints in delay management. In *CASPT 06*, 2006. ARRIVAL Report TR-0017.

[Sch06b] A. Schöbel. *Customer-oriented optimization in public transportation*. Optimization and Its Applications. Springer, New York, 2006.

[Sch07] A. Schöbel. Integer programming approaches for solving the delay management problem. In *Algorithmic Methods for Railway Optimization*, Lecture Notes in Computer Science. Springer, 2007. to appear.

[SM99] L. Suhl and T. Mellouli. Requirements for, and design of, an operations control system for railways. In *Computer-Aided Transit Scheduling*. Springer, 1999.

[SMBG01] L. Suhl, T. Mellouli, C. Biederbick, and J. Goecke. Managing and preventing delays in railway traffic by simulation and optimization. In *Mathematical methods on Optimization in Transportation Systems*, pages 3–16. Kluwer, 2001.

Institut für Numerische und Angewandte Mathematik
Universität Göttingen
Lotzestr. 16-18
D - 37083 Göttingen

Telefon: 0551/394512
Telefax: 0551/393944

Email: trapp@math.uni-goettingen.de URL: http://www.num.math.uni-goettingen.de

Verzeichnis der erschienenen Preprints 2007:

2007-01	P. Serranho	A hybrid method for inverse scattering for sound-soft obstacles in 3D
2007-02	G. Matthies, G. Lube	On streamline-diffusion methods for inf-sup stable discretisations of the generalised Oseen Problem
2007-03	A. Schöbel	Capacity constraints in delay management
2007-04	J. Brimberg, H. Juel, A. Schöbel	Locating a minisum circle on the plane
2007-05	C. Conte, A. Schöbel	Identifying dependencies among delays
2007-06	D.S. Gilliam, T. Hohage, X. Ji, F. Ruymgaart	The Frechet-derivative of an analytic function of a bounded operator with some applications
2007-07	T. Hohage, M. Pricop	Nonlinear Tikhonov regularization in Hilbert scales for inverse boundary value problems with random noise
2007-08	C.J.S. Alves, R. Kress, A.L. Silvestre	Integral equations for an inverse boundary value problem for the two-dimensional Stokes equations
2007-09	A. Ginkel, A. Schöbel	To wait or not to wait? The bicriteria delay management problem in public transportation
2007-10	O. Ivanyshyn, R. Kress	Inverse scattering for planar cracks via nonlinear integral equations
2007-11	F. Delbary, K. Erhard, R. Kress, R. Potthast, J. Schulz	Inverse electromagnetic scattering in a two-layered medium with an application to mine detection
2007-12	M. Pieper	Vector hyperinterpolation on the sphere
2007-13	M. Pieper	Nonlinear integral equations for an inverse electromagnetic scattering problem
2007-14	I. Akduman, R. Kress, N. Tezel, F. Yaman	A second order Newton method for sound soft inverse obstacle scattering

2007-15	M. Schachtebeck, A. Schöbel	Algorithms for delay management with capacity constraints