The Trickle-In Effect: Modeling Passenger Behavior in Delay Management

Anita Schöbel
Technical University of Kaiserslautern, Germany
Fraunhofer Institute for Industrial Mathematics ITWM, Kaiserslautern, Germany
schoebel@mathematik.uni-kl.de

Julius Pätzold
University of Goettingen, Germany
j.paetzold@math.uni-goettingen.de

Jörg P. Müller
Department of Informatics, Clausthal University of Technology, Germany
joerg.mueller@tu-clausthal.de

Abstract

Delay management is concerned with making decisions if a train should wait for passengers from delayed trains or if it should depart on time. Models for delay management exist and can be adapted to capacities of stations, capacities of tracks, or respect vehicle and driver schedules, passengers' routes and further constraints. Nevertheless, what has been neglected so far, is that a train cannot depart as planned if passengers from another train trickle in one after another such that the doors of the departing train cannot close. This effect is often observed in real-world, but has not yet been taken into account in delay management.

We show the impact of this "trickle-in" effect to departure delays of trains under different conditions. We then modify existing delay management models to take the trickle-in effect into account. This can be done by forbidding certain intervals for departure. We present an integer programming formulation with these additional constraints resulting in a generalization of classic delay management models. We analyze the resulting model and identify parameters with which it can be best approximated by the classical delay management problem.

Experimentally, we show that the trickle-in effect has a high impact on the overall delay of public transport systems. We discuss the impact of the trickle-in effect on the objective function value and on the computation time of the delay management problem. We also analyze the trickle-in effect for timetables which have been derived without taking this particular behavioral pattern of passengers into account.

2012 ACM Subject Classification Applied computing → Transportation

Keywords and phrases Public Transport Planning, Delay Management, Integer Programming

Digital Object Identifier 10.4230/OASIcs.ATMOS.2019.6

Funding *Julius Pätzold*: supported by Simulation Science Center (SWZ) Clausthal-Göttingen.

1 Introduction

Delays constitute a major source of uncertainty when operating a railway or bus system. If a train is delayed, many rescheduling decisions have to be made, disturbing the nominal schedule of a public transport system. The question, whether an otherwise punctual train should wait for a delayed feeder train in order to allow transferring passengers to reach their connections, is known as *delay management problem* and has been studied extensively in the literature. The first papers dealing with this kind of question date back to [21, 23]. Integer programming models have been developed in [22, 5]. In order to make them more realistic, capacities along tracks have been included in [18], capacities at stations have been included in [7] and passenger re-routing has been studied in [9, 20, 17]. Rescheduling of timetables, rolling stock and crew is studied in [8]. For all these cases algorithms have been developed, see [10] for a recent survey on the state of the art.

Delay management aims at minimizing passengers' delays by taking dependencies between delays into account (see [4]). Delays propagate along driving activities, i.e., if a train departs with some delay, then it also arrives at its next station with some delay – reduced by buffer time possibly included in the timetable. Delays also propagate along waiting activities in stations: If a train arrives at a station with some delay, it will probably also depart with some delay which again might have been reduced by buffer time. Finally, delays can propagate along changing activities as well. This is the case if a dispatcher decides that a connection from one train to another train should be maintained. Then the outgoing train will receive some delay by waiting for the delayed feeder train.

Nevertheless, there is an effect that has been neglected in the literature so far: A dispatcher may decide that a train should depart on time, but it may not be possible for the train to do so. To illustrate this issue, suppose a delayed train A arrives at a station and some of its passengers want to transfer at this station to another train B. The delay management problem requires a decision, whether train B should depart on time or wait for the passengers from train A.

- If train B is supposed to depart before train A has arrived, the delay management models work correctly. In this case, no delay propagates from train A to train B.
- If train B is supposed to wait long enough, the delay management models also work correctly and the delay propagates along the changing activity to the departure of train B.
- If, however, train B is supposed to depart shortly after train A has arrived without waiting for the passengers from train A, then the models fail. This is the case because normal delay management models assume that there is one common time that passengers need for walking from train A's platform to train B's platform. Instead, there may be quick and slow passengers. If the fastest passenger reaches train B before its departure, she can board. While getting onto train B, another fast passenger might arrive and while he boards, the next one will arrive, and so forth. In this way, all passengers might enter the train in a continuous stream preventing the train doors to close. Train B hence has to wait until finally even the slowest passengers from train A arrive and board train B. This effect has been simulated in [1] where it is called *trickle-in effect.*
- The same effect may also prolong the waiting time of train B in the case that B is supposed to wait for the passengers of train A since it may take longer to allow all passengers to trickle in than the lower bounds on the changing times suggest.

Note that the trickle-in effect is not only triggered by passengers not moving with the same speed, but also by the fact that passengers are not able to unboard train A instantaneously. Most readers will have experienced the situation of standing in a train corridor while waiting a decent amount of time for the passengers in front of them to unboard. This can result in a different transfer time of two passengers, even though they are able to walk with similar speed.

As a consequence, there exists a time interval in which train B is not able to depart, namely between the arrival of the fastest passenger and the arrival of the slowest passenger (assuming that there is no gap in speed of the passengers big enough to allow the doors of train B to close). We will call this interval *trickling interval.*

[1] show that the trickle-in effect, which can also be observed in many real-world situations, is in fact relevant. Our experiments (see Section 5) show that delay management decisions, which are optimal in the sense of classical delay management models, often schedule trains to depart in the "forbidden" trickling interval. If, for example, the trickling interval is $(2, 5)$

minutes and if all changing activities are distributed uniformly in $[2, 62)$ minutes (assuming a time period of 60 minutes), we can expect about 5% of all train departures to lie in the trickling interval. These departures are most likely not realizable and will cause additional delays. Hence, it is necessary to add this additional constraint to delay management models which is exactly what we do in this paper. We show how such an additional constraint can be included in classical delay management models, subsequently analyze the mathematical relation between the classical model and the one with the additional constraints, and finally show in experiments that delay management strategies change if the trickle-in effect is considered. We believe that by adding this detail we take a further step in bringing delay management models closer to practice.

The remainder of the paper is structured as follows. In Section 2 we recap the classical model for delay management. Section 3 models the trickle-in effect by introducing an additional constraint to the classical delay management model. We investigate theoretical consequences when adding the trickle-in effect to the classical delay management model in Section 4. Section 5 studies its practical effects in an experimental study on close-to-real-world data from LinTim [11, 19]. Integrating the trickle-in effect in models for (periodic) timetabling is identified as an extension and briefly discussed in Section 6, where we also conclude the contributions, discuss limitations of our work as well as venues of future research.

2 The Classical Delay Management Model

The *delay management problem* is defined as follows: Given an event-activity network, a timetable and some source delays, decide which connections should be maintained and which should be dropped such that the average delay of all passengers at their final destinations is minimal. The delay management problem was first introduced in [21], a recent overview is given in [10].

We hence have to first introduce the concept of *event-activity networks* (see [14] for its application in timetabling and [22] for its application in delay management). An event-activity network is a directed graph $\mathcal{N} = (\mathcal{E}, \mathcal{A})$, where $\mathcal{E}$ consists of arrival and departure events $\mathcal{E}_{\text{arr}}$ and $\mathcal{E}_{\text{dep}}$, respectively. A timetable $\pi \in \mathbb{N}^{|\mathcal{E}|}$ assigns each event $i \in \mathcal{E}$ to a time $\pi_i \in \mathbb{N}$. If a delay occurs, the given timetable π has to be updated to a so-called *disposition timetable* $x \in \mathbb{N}^{|\mathcal{E}|}$. To represent the constraints that have to be satisfied by a (disposition) timetable, we need the following types of activities, $\mathcal{A} = \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}} \cup \mathcal{A}_{\text{change}}$. Each of them is assigned to a minimal duration $L_a > 0$. The meaning of these activities is given as follows (see also Figure 1).

- *Driving activities* $\mathcal{A}_{\text{drive}} \subset \mathcal{E}_{\text{dep}} \times \mathcal{E}_{\text{arr}}$ model the driving of a train between two consecutive stations, i.e. a driving activity connects a departure event of some train with its next arrival event. The duration $L_a > 0$ of a driving activity $a = (i, j)$ represents the minimal necessary driving time between the departure event i and the arrival event j. Note that turnaround edges may be handled in the same way as driving activities.
- *Waiting activities* (also called *dwelling activities*) $\mathcal{A}_{\text{wait}} \subset \mathcal{E}_{\text{arr}} \times \mathcal{E}_{\text{dep}}$ represent the time period in which a train is waiting at a station to let passengers get on or off; a waiting activity hence connects an arrival event of some train with its next departure event. Its duration $L_a > 0$ describes the minimal time required to allow boarding and unboarding; sometimes it also includes exchanging train crews or other actions.

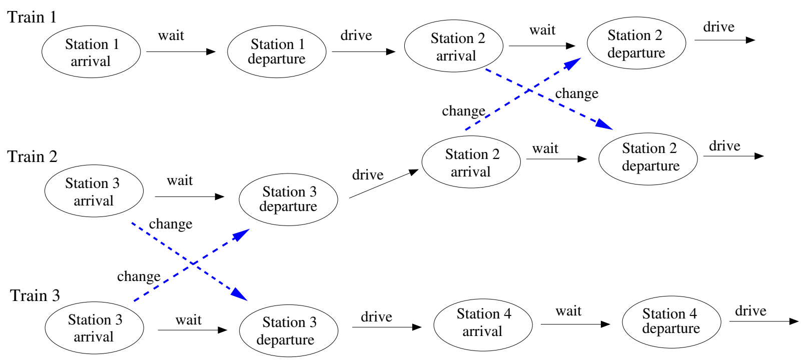


Figure 1 An event-activity network with three trains and four stations. The solid (black) arcs represent driving and waiting activities of the trains. The dashed (blue) arcs represent changing activities which are possible between Train 2 and Train 3 at Station 3 and between Train 1 and Train 2 at Station 2.

If two events $i, j \in \mathcal{E}$ are connected by an activity $(i, j) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}}$, then event i has to be performed before event j can take place. In particular, the disposition timetable x has to satisfy

$$x_j - x_i \geq L_a$$

for all $a = (i, j) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}}$.

- *Changing activities* $\mathcal{A}_{\text{change}} \subset \mathcal{E}_{\text{arr}} \times \mathcal{E}_{\text{dep}}$ allow passengers to transfer from an incoming train to an outgoing train. Hence, a changing activity connects an arrival event of some train at some station with a departure event of another train at the same station. The lower bound $L_a > 0$ refers to the minimum time a passenger needs to transfer between both trains. In order to solve the delay management problem we have to decide for each changing activity if it should be kept or if it can be deleted. In case that a changing activity $a = (i, j)$ is kept, the disposition timetable x must satisfy $x_j - x_i \geq L_a$. If the changing activity is deleted, the outgoing train can depart without waiting for the incoming train and this inequality does not need to be satisfied anymore.

We remark that other types of activities such as *headway activities* or *turnaround activities* may be added, see [10] for the respective models. Notwithstanding that, in this work we focus on the classical model.

To formulate an integer programming model of the delay management problem, we next have to formally introduce the delays. We assume that a set of unexpected *source delays* is known, e.g., caused by signaling problems, construction work, accidents, or bad weather conditions. These source delays cause *secondary delays*, e.g., for the same train at subsequent stations or for other trains that wait for the delayed train. In our work we allow two types of source delays: The first type is a delay $d_i \in \mathbb{N}$ at an event $i \in \mathcal{E}$ (e.g., staff coming too late to their duty) referring to a fixed point of time. In this case, $x_i \geq \pi_i + d_i$ is required. The second type of source delay is a delay d_a which increases the duration of an activity $a = (i, j) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}}$, e.g., an increase of traveling time between two stations due to construction work. Such a delay d_a has to be added to the minimal duration L_a of activity a. If an event or an activity has no source delay, we assume $d_i = 0$ or $d_a = 0$, respectively, to simplify the notation.

In the objective function we evaluate the disposition timetable from the passengers' point of view. To this end, let w_i be the number of passengers unboarding the train at event $i \in \mathcal{E}$ (thus, $w_i = 0$ for all $i \in \mathcal{E}_{\text{dep}}$) and w_a be the number of passengers who want to use a changing activity $a \in \mathcal{A}_{\text{change}}$. We assume $w_a > 0$ for all $a \in \mathcal{A}_{\text{change}}$ – otherwise, the changing activity could be removed from the network, since nobody uses it. We further assume that all lines have a common period T, i.e., every line is served by a train every T minutes. Note that this assumption can be relaxed by introducing periods T_a for all changing activities $a \in \mathcal{A}_{\text{change}}$.

We can now state the integer programming formulation for the basic version of the delay management problem. To model the wait-depart decisions, i.e., whether some train should wait for some other train at a station or not, 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 $a \in \mathcal{A}_{\text{change}}$. The integer programming formulation then reads as follows:

$$\min \sum_{i \in \mathcal{E}} w_i(x_i - \pi_i) + \sum_{a \in \mathcal{A}_{\text{change}}} z_a w_a T \tag{DM}$$

$$\begin{aligned}
\text{s.t.} \qquad x_i &\geq \pi_i + d_i & \forall i \in \mathcal{E} \quad &(1)\\
x_j - x_i &\geq L_a + d_a & \forall a = (i,j) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}} \quad &(2)\\
M z_a + x_j - x_i &\geq L_a & \forall a = (i,j) \in \mathcal{A}_{\text{change}} \quad &(3)\\
x_i &\in \mathbb{N} & \forall i \in \mathcal{E} \quad &\\
z_a &\in \{0,1\} & \forall a \in \mathcal{A}_{\text{change}} \quad &
\end{aligned}$$

where M is a fixed constant. The meaning of the objective function and of the constraints is explained next.

The first term of the objective function minimizes the sum of all delays of all events. If all connections were maintained, this would be the sum of delays for all passenger at their final destination. The second term adds the weighted sum of all missed connections with a penalty of one time period T (or T_a if we drop the assumption of a common period of all lines) a passenger has to wait for the next train of the same line. The objective function is hence an approximation of the sum of all delays over all passengers. It has been shown in [22] that it is not an approximation, but exactly computes the sum of all passengers' delays if the so-called never-meet property holds.

Constraints (1) and (2) ensure that the delay is passed on correctly along driving and waiting activities. (3) does the same for maintained changing activities (i.e. if $z_a = 0$). If, however, $z_a = 1$, constraints (3) get redundant if M is chosen big enough. If no capacity constraints are considered and $d_a = 0$ for all $a \in \mathcal{A}$, [22] shows that choosing M as the largest source delay $\max_{i \in \mathcal{E}} d_i$ is sufficient. Solution methods for (DM) mainly rely on integer programming, see [10] and references therein.

3 Modeling the Trickle-in Effect

In this section we adapt the classical delay management model (DM) by taking the following two phenomena into account:

1. Passengers do not change with the same speed. There may be fast and slow passengers and a decision for keeping a changing activity means practically that the train waits for all (even for the slowest) passengers.
2. Due to the trickle-in effect, trains are not able to depart while passengers are still boarding.

The first point is modeled by using a time interval $(L_a^{\min}, L_a^{\max})$ instead of a fixed time L_a to describe the duration of the changing activities. We hence replace L_a in constraints (3) by $L_a^{\max}$. The second point implies that a train can either depart before the fastest passenger has arrived or after the slowest one has boarded, i.e., it cannot depart in the interval $(x_i + L_a^{\min}, x_i + L_a^{\max})$. This restriction is modeled by adding new constraints as follows.

▶ **Lemma 1.** *Let $a = (i,j) \in \mathcal{A}_{\text{change}}$. There exists $z_a \in \{0,1\}$ such that*

$$Mz_a + x_j - x_i \geq L_a^{\max} \tag{4}$$

$$M(z_a - 1) + x_j - x_i \leq L_a^{\min} \tag{5}$$

are both satisfied if and only if

$$x_j \notin (x_i + L_a^{\min}, x_i + L_a^{\max}). \tag{6}$$

Proof. Let (4) and (5) hold for some $z_a \in \{0,1\}$. If $z_a = 0$, (4) reduces to $x_j \geq x_i + L_a^{\max}$. On the other hand, if $z_a = 1$ then (5) reduces to $x_j \leq x_i + L_a^{\min}$. In both cases, $x_j \notin (x_i + L_a^{\min}, x_i + L_a^{\max})$.

Vice versa, let $x_j \notin (x_i + L_a^{\min}, x_i + L_a^{\max})$. If $x_j \leq x_i + L_a^{\min}$ we choose $z_a = 1$ to see that both, (4) and (5) hold. On the other hand, if $x_j \geq x_i + L_a^{\max}$ then $z_a = 0$ guarantees that (4) and (5) are satisfied. ◀

The proof of Lemma 1 specifies two possible cases for a dispatcher:

- The changing activity is maintained ($z_a = 0$) if and only if the train departs after the last passengers have boarded $x_j \geq x_i + L_a^{\max}$.
- The changing activity is dropped ($z_a = 1$) if and only if the train departs before the first passengers have boarded $x_j \leq x_i + L_a^{\min}$.

The resulting model (DM-trick) is hence given as

$$\min \quad \sum_{i \in \mathcal{E}} w_i(x_i - \pi_i) + \sum_{a \in \mathcal{A}_{\text{change}}} z_a w_a T \tag{DM-trick}$$

$$\begin{aligned}
\text{s.t.} \quad x_i &\geq \pi_i + d_i && \forall i \in \mathcal{E} && (7)\\
x_j - x_i &\geq L_a + d_a && \forall a = (i,j) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}} && (8)\\
Mz_a + x_j - x_i &\geq L_a^{\max} && \forall a = (i,j) \in \mathcal{A}_{\text{change}} && (9)\\
M(z_a - 1) + x_j - x_i &\leq L_a^{\min} && \forall a = (i,j) \in \mathcal{A}_{\text{change}} && (10)\\
x_i &\in \mathbb{N} && \forall i \in \mathcal{E}\\
z_a &\in \{0,1\} && \forall a \in \mathcal{A}_{\text{change}}
\end{aligned}$$

We remark that (DM-trick) contains (DM) as a special case by setting $L_a^{\max} := L_a$ and $L_a^{\min} := L_a - 1$ for all $a \in \mathcal{A}$, i.e., it is a proper extension of the classical delay management model: Constraint (10) may not be contained as an explicit constraint in (DM), but is implicitly contained. This is the case because for every transfer $a = (i,j) \in \mathcal{A}$ that is missed ($z_a = 1$) it needs to hold that $x_j - x_i \leq L_a - 1$ since otherwise there would have been enough time to connect event i with event j and due to the objective function z_a would have been set to 0.

Trickle-in constraints can also be combined with all other extensions known for delay management, i.e., it is possible to consider headway constraints as in [18], station capacity constraints as in [7], or passenger routing constraints as in [9]. For the sake of simplicity we compare (DM) and (DM-trick) in their basic versions as given above.

4 Analyzing the New Model

As already mentioned in Section 2, for the classical delay management problem it suffices to choose M as large as the largest source delay $D := \max_{i \in \mathcal{E}} d_i$ if all $d_a = 0$. This does not hold any more for (DM-trick), but still the size of M can be bounded. To this end, we need the following two lemmas, both dealing with the original timetable π_i, $i \in \mathcal{E}$. For this chapter, we assume that the original timetable π is feasible, i.e., that

$$\pi_j - \pi_i \in [L_a, U_a] \quad \forall a = (i,j) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}}, \tag{11}$$

for all driving and waiting activities. For the changing activities we assume that the trickling constraints (6) applied to the original timetable π

$$\pi_j \notin (\pi_i + L_a^{\min}, \pi_i + L_a^{\max}) \quad \forall a = (i,j) \in \mathcal{A}_{\text{change}} \tag{12}$$

are satisfied, i.e., either nobody can change or everybody can. However, *changing* activities are the ones that allow passengers to change, so the case $\pi_j - \pi_i \leq L_a^{\min}$ cannot hold. We hence may assume that

$$\pi_j - \pi_i \in [L_a^{\max}, T + L_a^{\min}] \quad \forall a = (i,j) \in \mathcal{A}_{\text{change}} \tag{13}$$

where the upper bound $T + L_a^{\min}$ holds since every line runs at least once per time period T.

In order to simplify the notation, we will sometimes use the *delay* y_i of an event $i \in \mathcal{E}$ in its disposition timetable, which is defined as

$$y_i := x_i - \pi_i.$$

▶ **Lemma 2.** *Let π_i, $i \in \mathcal{E}$ be a feasible timetable. If all $d_a = 0$, then there exists an optimal solution with $y_j \leq D$ for all $a = (i,j) \in A$.*

Proof. The proof works by induction. Since the event-activity network does not contain any directed cycles, we can sort the events $i \in \mathcal{E}$ topologically. Let $i_1, \ldots, i_{|\mathcal{E}|}$ be the resulting order. Then the delay y_{i_1} of the first event i_1 is given by $d_{i_1} \leq D$. Now take any other event j and consider all of its incoming activities $(i,j) \in \mathcal{A}$. We now estimate how large the delay of event j can be. Note that there exists an optimal solution in which no disposition time can be reduced (i.e., which does not contain any unnecessary delays). This means one of the inequality constraints (7), (8), (9) is sharp.

- If (7) is sharp we get $x_j = \pi_j + d_j$, hence $y_j = x_j - \pi_j = d_j \leq D$.
- If (8) is sharp for $(i,j) \in \mathcal{A}$ we have that $x_j = x_i + L_a$, i.e., the delay of event j can be computed as

$$\begin{aligned} y_j &= x_j - \pi_j = L_a + x_i - \pi_j \\ &= L_a + y_i + \underbrace{\pi_i - \pi_j}_{\leq -L_a} \\ &\leq y_i \leq D \text{ by induction hypothesis} \end{aligned}$$

  where we have used feasibility of the timetable, see (11) and that event i is topologically smaller than event j.
- If (9) is sharp for $(i,j) \in \mathcal{A}$ we analogously have that $x_j = x_i + L_a^{\max}$, i.e., the delay of event j can be computed as

$$\begin{aligned} y_j &= x_j - \pi_j = L_a^{\max} + x_i - \pi_j \\ &= L_a^{\max} + y_i + \underbrace{\pi_i - \pi_j}_{\leq -L_a^{\max}} \\ &\leq y_i \leq D \text{ by induction hypothesis} \end{aligned}$$

  where we have used the second feasibility constraint for the timetable, see (13) and again that event i is topologically smaller than event j. ◀

Under the same conditions as in the above lemma we can hence estimate the size of big M, which is a bit larger than in (DM) but still of moderate size.

▶ **Lemma 3.** *If $d_a = 0$ for all $a \in \mathcal{A}$, then $M = T + D$ is large enough to correctly solve Model* (DM-trick)*.*

Proof. We have to find M that satisfies the following two conditions:

1. Constraint (4) should get redundant if $z_a = 1$, i.e., for an optimal solution we require that $M \geq L_a^{\max} + x_i - x_j$. We hence look for an upper bound of the right hand side:
$$\begin{aligned} L_a^{\max} + x_i - x_j &= L_a^{\max} + \pi_i + y_i - \pi_j - y_j \\ &= L_a^{\max} + \underbrace{\pi_i - \pi_j}_{\leq -L_a^{\max}} + \underbrace{y_i}_{\leq D} \underbrace{-y_j}_{\leq 0} \leq D, \end{aligned}$$
where we again used feasibility of the timetable, see (13).
2. Constraint (5) should get redundant if $z_a = 0$, i.e., for an optimal solution we require that $M \geq x_j - x_i - L_a^{\min}$. We again need an upper bound of the right hand side:
$$\begin{aligned} x_j - x_i - L_a^{\min} &= \pi_j + y_j - \pi_i - y_i - L_a^{\min} \\ &= \underbrace{\pi_j - \pi_i}_{\leq T + L_a^{\min}} + \underbrace{y_j}_{\leq D} \underbrace{-y_i}_{\leq 0} - L_a^{\min} \leq T + D, \end{aligned}$$
this time using the upper bound in (13).

We conclude that $M = D + T$ suffices for both constraints (4) and (5). ◀

In the case of $d_a > 0$, delays can increase for single trains and have to be bounded. This can theoretically be done by summing up all delays d_a or (better) by finding a longest path P in the event-activity network with respect to the weights d_a, see [18].

Let us now consider the case that the timetable is feasible according to its traditional definition *without* the trickle-in effect, i.e., it satisfies $\pi_j - \pi_i \in [L_a, T + L_a - 1]$ instead of (13) for some $L_a < L_a^{\max}$. Then the trickle-in effect may generate delays.

Let us illustrate this on a small example: Given a timetable π that schedules train A to arrive at 10:00 and train B to depart at 10:02 and given a trickling interval of $(1, 3)$ minutes, then the trickle-in effect is observable. The first passengers only need a little bit more than one minute to catch the train, but then a continuous stream of passengers boards the train leading to a delayed departure of train j at 10:03, i.e., to a delay of one minute. Thus, there may occur delays due to the trickle-in effect without the existence of any source delays.

However, even in this situation we can use (DM-trick) to find optimal wait-depart decisions dealing with both, source delays and delays occurring due to trickling constraints, and even in this situation we can bound M. To this end, assume a changing activity $a = (i, j)$ from event i to event j for which we have $L_a < \pi_j - \pi_i < L_a^{\max}$. Then the transfer of all passengers may take longer than the timetable allows. Hence, the trickle-in effect leads to a new type of "source delay" on this changing activity, namely a delay of $L_a^{\max} - (\pi_j - \pi_i)$. In order to find a bound for M we hence need to search for a longest path P' with respect to the weights

$$w'_a := \begin{cases} \max(0, d_a) & \text{if } a \in \mathcal{A}_{\text{wait}} \cup \mathcal{A}_{\text{drive}} \\ \max(0, d_a, L_a^{\max} - (\pi_j - \pi_i)) & \text{if } a = (i, j) \in \mathcal{A}_{\text{change}} \end{cases}$$

and add its length to M.

Hence, we receive a bound of

$$M = D + T + \text{length}(P')$$

in this case. The computational experiments show that $M = D + T + \text{length}(P')$ is of reasonable size and hence a sufficient upper bound for M.

We now analyze the new model (DM-trick) with respect to the intervals $[L_a^{\min}, L_a^{\max}]$ for the changing activities. Varying both, the lower and the upper bound on the duration of the changing activities gives the following result.

▶ **Lemma 4.** *Let $I_a(k) = [L_a^{\min}(k), L_a^{\max}(k)]$ for all $a \in \mathcal{A}_{change}$ be a sequence of nested intervals with*

$$L_a^{\min}(1) \leq L_a^{\min}(2) \leq \cdots \leq L_a \text{ and } L_a \leq \cdots \leq L_a^{\max}(2) \leq L_a^{\max}(1)$$

and let $z^*(k)$ be the optimal objective function value for (DM-trick) *with respect to the interval $I_a(k)$ and z^* be the optimal objective function value of* (DM)*. Then*

$$z^*(1) \geq z^*(2) \geq \cdots \geq z^*.$$

Proof. Since $I_a(k+1) \subseteq I_a(k)$ for all $a \in \mathcal{A}_{change}$, (DM-trick) with respect to the intervals $I(k+1)$ is a relaxation of (DM-trick) with respect to the intervals $I(k)$ and the result follows. ◀

As a consequence, (DM) is a relaxation of (DM-trick) whenever the changing times L_a in the classical model (DM) satisfy $L_a \in [L_a^{\min}, L_a^{\max}]$ for all $a \in \mathcal{A}_{change}$. Hence, solving (DM) gives a lower bound on (DM-trick). In the experiments in Section 5 we compare the gap between this lower bound and the real solution. The best approximation of (DM-trick) by (DM) is given if we set $L_a := L_a^{\max}$ for all $a \in \mathcal{A}_{change}$, i.e., making sure that also the slow passengers are able to board their next train. This is shown in the next Lemma.

▶ **Lemma 5.** *Let $[L_a^{\min}, L_a^{\max}]$ for all $a \in \mathcal{A}_{change}$ be the given data for* (DM-trick)*. Let $z^*(L_a)$ be the optimal objective function value for* (DM) *with data L_a for all $a \in \mathcal{A}_{change}$. Then an optimal solution to*

$$\max\{z^*(L_a) : L_a \in [L_a^{\min}, L_a^{\max}] \text{ for all } a \in \mathcal{A}_{change}\}$$

is provided by setting $L_a = L_a^{\max}$ for all $a \in \mathcal{A}_{change}$, i.e., the best lower bound obtainable from the classical model (DM) *is provided by setting $L_a := L_a^{\max}$ for all $a \in \mathcal{A}$.*

Proof. From Lemma 4 we already know that all $L_a \in [L_a^{\min}, L_a^{\max}]$ provide lower bounds. We hence have to show that the largest of them is obtained by setting $L_a := L_a^{\max}$ for all $a \in \mathcal{A}$. To this end, let $L'_a \leq L_a^{\max}$ for all $a \in \mathcal{A}$. Let (x, z) be a solution of (DM) with respect to $L_a^{\max}$. It hence satisfies (3) with $L_a^{\max}$ on the right hand side and hence also with $L'_a \leq L_a^{\max}$ on the right hand side. Hence, (x, z) is also feasible for (DM) with respect to L'_a. We conclude that (DM) with respect to L'_a is a relaxation of (DM) with respect to $L_a^{\max}$, and hence

$$z^*(L'_a) \leq z^*(L_a^{\max}).$$ ◀

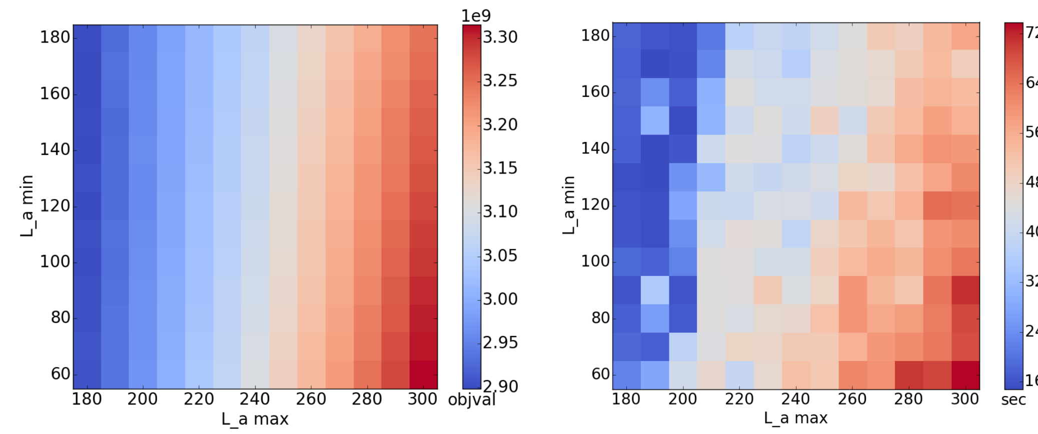
The computational results underline that using (DM) as a relaxation for (DM-trick) impose a good trade-off between computation time and (DM)'s quality as a lower bound.

5 Experiments

In this section we investigate the effects of the trickle-in effect computationally. To this end, we implemented (DM-trick) in LinTim, an open source software framework for public transport optimization, see [19, 11]. We focus on solving the *bahn* dataset, consisting of 250 nodes and 326 edges, modeling the German ICE network, see Figure 3 in the appendix.

For quickly determining the timetable we use the MATCH heuristic as described in [16] since it is faster than the modulo simplex [12, 15] or integer programming approaches [13]. We roll out the periodic timetable for 4 hours and receive an aperiodic event-activity-network with around 20000 events and 40000 activities. For generating delays we use a LinTim procedure which is parameterized to choose 1000 activities and to generate source delays uniformly distributed between 1 and 900 seconds for each of the chosen activities. In order to calculate a sufficiently big M as described in Section 4, we calculate length$(P) = 3500$ seconds and $D = 0$ (because we generate source delays only on activities) and T was chosen to be 3600 seconds, leading to a choice of $M = 7100$. We implemented (DM-trick) using Gurobi 8.0 with a relative optimality gap of 1% and run the experiments on a compute server with 12 cores of Intel(R) Xeon(R) CPU X5650 @ 2.67 GHz and 78 GB RAM.

In our first experiment we compare different trickling intervals with the lower bound $L_a^{\min}$ ranging from 60 to 180 seconds and $L_a^{\max}$ ranging from 180 to 300 seconds. The default minimum changing time L_a is assumed to be 180 seconds. The objective values, given in passengers times seconds, for solving (DM-trick) with these different intervals are depicted in Figure 2a.



(a) objective values in passengers × seconds. **(b)** runtimes in seconds.

Figure 2 (DM-trick) for different trickling intervals $(L_a^{\min}, L_a^{\max})$ in seconds.

We see that the instance with the smallest trickling interval ($[180, 180]$ seconds) has the lowest objective value of about $2.9 \cdot 10^9$, whereas the instance with the largest trickling interval ($[60, 300]$ seconds) has the largest objective value ($3.3 \cdot 10^9$). This is consistent with the theory since small trickling intervals are a relaxation of larger trickling intervals, see Lemma 4. A higher objective value is equivalent to higher passenger delays in the event-activity-network which makes sense as a larger trickling interval potentially leads to longer waiting times for trains. In general, one can observe that a larger interval correlates with a higher objective value, and furthermore that a change in $L_a^{\max}$ has a higher impact on the objective value than a change in $L_a^{\min}$.

Figure 2b now depicts the runtimes for different choices of the trickling interval.

Interestingly, also the instance with the smallest trickling interval has the lowest runtime and the instance with the largest trickling interval has the highest runtime. Also for the other instances the runtimes correlate primarily with the size of the trickling interval (although not as smoothly as the objective value) and a change in the upper bound $L_a^{\max}$ has again a higher impact on the runtime than a change in $L_a^{\min}$. The correlation between size of the trickling interval and runtime can be explained by the nature of integer programming. If the size of the "forbidden" trickling interval is increased, we get a weaker linear relaxation of the integer problem and hence need longer to solve it, e.g., via branch-and-bound.

The next experiment investigates the difference between a disposition timetable found by (DM) and a disposition timetable that respects the trickle-in effect. To this end, Figure 4 (in the appendix) depicts the number of changing activities of a disposition timetable from (DM) (with $L_a = 180$ seconds) that lie in the trickling interval. Hence, Figure 4 illustrates the difference between a disposition timetable found by solving (DM) and disposition timetables found by solving (DM-trick) for different trickling intervals. As can be seen in the figure, there exist up to 1194 infeasible change activities for a disposition timetable from (DM). In other words, if a disposition timetable from (DM) is found, it can be the case that 1194 changing activities (or about 6% of all changing activities) cause new delays due to the neglection of the trickle-in effect.

Furthermore, we investigate the results of Lemma 5, i.e., that solving (DM) with $L_a := L_a^{\max}$ yields the best approximation to (DM-trick).

One can see in Figure 5 (in the appendix) that the objective value indeed increases when L_a increases, culminating in a gap of only 3% if L_a is chosen to be $L_a^{\max}$. Hence, we get a reasonably good approximation of (DM-trick) by only solving an instance of (DM). Furthermore, it should be noted that solving (DM) takes only around 1 second, whereas solving (DM-trick) with trickling interval $[60, 300]$ seconds took around 77 seconds to solve. Hence, we indeed get a decent trade-off between computation time and solution quality.

Finally, we investigate the difference between the disposition timetables from (DM) and (DM-trick) (with a trickling interval of $[60, 300]$ seconds). We observe that the solution from (DM) schedules trains such that 566 connections are missed, whereas in the solution of (DM-trick) there are 684 missed connections. Interestingly, 513 of the missed connections coincide such that in this case considering the trickling effect yields to a change in 224 wait-depart-decisions. Thus, not only the objective values of the two models (DM) and (DM-trick) varies, but also the structure of the resulting delay management strategy.

As a final note, we also run the model (DM-trick) for the case if no source delays exist and received an objective value of around $5 \cdot 10^8$. This is roughly 15% of the objective value we encountered while working with the aforementioned 1000 source delays. Put differently, in this instance up to 15% of the delays might not be caused by source delays, but by the mere structure of the underlying periodic timetable and the trickle-in effect. Hence, the trickle-in effect has high relevance beyond delay management and should already be considered when planning a periodic timetable (which is not the case when using, e.g., MATCH for timetabling).

6 Conclusion and Suggestions for Further Research

In this paper we introduced the trickle-in effect, an observation on passenger behavior at train stations that highly influences delays in public transport. We introduced models for incorporating the trickle-in effect into standard delay management models and also showed how it already influences the periodic timetabling problem. We investigated mathematical properties of the resulting model and showed how (DM-trick) can be approximated best using the classical delay management problem. This allows to use approaches for classical delay management (such as [3, 17, 8, 6]) for heuristically solving (DM-trick). The computational experiments underlined our hypothesis that the trickle-in effect has a high impact on delay management: Here, the objective value of (DM-trick) exceeds the objective value of (DM) up to 15%. Finally, since the computation times for (DM-trick) rise significantly, we still can get a decent approximation of (DM-trick) by solving a modified version of (DM).

Further research includes simulation approaches to better understand the behaviour of the passengers and to derive practically relevant trickling intervals. To this end, an agent-based simulation as in [2] is currently developed. We are also interested in adding the trickling constraints to more sophisticated delay management models including passengers' routing and capacity constraints.

Finally, there is another line of research, namely adding trickling constraints to the timetabling problem. In Section 4 we have already seen that considering the trickle-in effect in a timetable that is not feasible with respect to (13) might cause source delays. The experiments justify this theoretical observation: a timetable might get significant delays just because of the trickle-in effect, i.e., even if no other source delays occur. We hence suggest to consider the trickle-in effect already in the timetabling phase. This means to add constraints of type (12) in timetabling such that either all passengers or none of the passengers can make a transfer. Hence, $\pi_j - \pi_i \in [L_a^{\max}, T + L_a^{\min}]$ needs to be respected for all changing activities (i, j) and even more general for all activities (i, j) from an arrival event of an incoming train to a departure event of (another) outgoing train. These constraints can be transferred also to periodic timetabling and considered as additional constraints in the periodic event scheduling problem (PESP). The analysis of them (runtime, impact on resulting timetable) are an interesting topic for future research which seems to be challenging and highly relevant from a practical point of view.

References

1 S. Albert, P. Kraus, J.P. Müller, and A. Schöbel. Passenger-induced delay propagation: Agent-based simulation of passengers in rail networks. In *Simulation Science*, volume 889 of *Communications in Computer and Information Science (CCIS)*, pages 3–23. Springer, 2018.

2 M. Aschermann, S. Dennisen, P. Kraus, and J.P. Müller. LightJason, a Highly Scalable and Concurrent Agent Framework: Overview and Application (demonstration paper). In M. Dastani, G. Sukthankar, E. Andre, and S. Koenig, editors, *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018)*, pages 1794–1796, 2018.

3 R. Bauer and A. Schöbel. Rules of Thumb — Practical online strategies for delay management. *Public Transport*, 6(1):85–105, 2014. URL: `http://num.math.uni-goettingen.de/preprints/files/2011-03.pdf`.

4 C. Conte and A. Schöbel. Identifying dependencies among delays. In *proceedings of IAROR 2007*, 2007. ISBN 978-90-78271-02-4.

5 L. De Giovanni, G. Heilporn, and M. Labbé. Optimization models for the single delay management problem in public transportation. *European Journal of Operational Research*, 189(3):762–774, 2008.

6 T. Dollevoet and D. Huisman. Fast Heuristics for Delay Management with Passenger Rerouting. *Public Transport*, 6(1-2):67–84, 2014.

7 T. Dollevoet, D. Huisman, L. Kroon, M. Schmidt, and A. Schöbel. Delay Management including capacities of stations. *Transportation Science*, 49(2):185–203, 2015.

8 T. Dollevoet, D. Huisman, L.G. Kroon, L.P. Veelenturf, and J.C.Wagenaar. Application of an iterative framework for real-time railway scheduling. *Computers and Operations Research*, 78:203–217, 2017.

9 T. Dollevoet, D. Huisman, M. Schmidt, and A. Schöbel. Delay Management with Rerouting of Passengers. *Transportation Science*, 46(1):74–89, 2012.

10 T. Dollevoet, D. Huisman, M. Schmidt, and A. Schöbel. Delay propagation and delay management in transportation networks. In R. Borndörfer et al., editor, *Handbook of Optimization in the Railway Industry*. Springer, 2018.

11 M. Goerigk, M. Schachtebeck, and A. Schöbel. Evaluating Line Concepts using Travel Times and Robustness: Simulations with the Lintim toolbox. *Public Transport*, 5(3), 2013.

12 M. Goerigk and A. Schöbel. Improving the Modulo Simplex Algorithm for Large-Scale Periodic Timetabling. *Computers and Operations Research*, 40(5):1363–1370, 2013.

13 C. Liebchen, M. Proksch, and F.H. Wagner. Performances of Algorithms for Periodic Timetable Optimization. In *Computer-aided Systems in Public Transport*, pages 151–180. Springer, Heidelberg, 2008.

14 K. Nachtigall. *Periodic Network Optimization and Fixed Interval Timetables*. PhD thesis, University of Hildesheim, 1998.

15 K. Nachtigall and J. Opitz. Solving Periodic Timetable Optimisation Problems by Modulo Simplex Calculations. In *Proc. ATMOS*, 2008.

16 J. Pätzold and A. Schöbel. A Matching Approach for Periodic Timetabling. In Marc Goerigk and Renato Werneck, editors, *16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2016)*, volume 54 of *OpenAccess Series in Informatics (OASIcs)*, pages 1–15, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/OASIcs.ATMOS.2016.1.

17 R. Rückert, M. Lemnian, C. Blendinger, S. Rechner, and M. Müller-Hannemann. PANDA: a software tool for improved train dispatching with focus on passenger flows. *Public Transport*, 9:307–324, 2017.

18 M. Schachtebeck and A. Schöbel. To wait or not to wait and who goes first? Delay Management with Priority Decisions. *Transportation Science*, 44(3):307–321, 2010. doi:10.1287/trsc.1100.0318.

19 A. Schiewe, S. Albert, J. Pätzold, P. Schiewe, A. Schöbel, and J. Schulz. LinTim: An integrated environment for mathematical public transport optimization. Documentation. Technical Report 2018-08, Preprint-Reihe, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen, 2018.

20 M. Schmidt and A. Schöbel. The complexity of integrating routing decisions in public transportation models. *Networks*, 65(3):228–243, 2015.

21 A. Schöbel. A Model for the Delay Management Problem based on Mixed-Integer Programming. *Electronic Notes in Theoretical Computer Science*, 50(1), 2001.

22 A. Schöbel. Integer Programming approaches for solving the delay management problem. In *Algorithmic Methods for Railway Optimization*, number 4359 in Lecture Notes in Computer Science, pages 145–170. Springer, 2007.

23 L. Suhl, C. Biederbick, and N. Kliewer. Design of customer-oriented Dispatching Support for railways. In S. Voß and J. Daduna, editors, *Computer-Aided Transit Scheduling*, volume 505 of *Lecture Notes in Economics and Mathematical systems*, pages 365–386. Springer, 2001.

A Figures

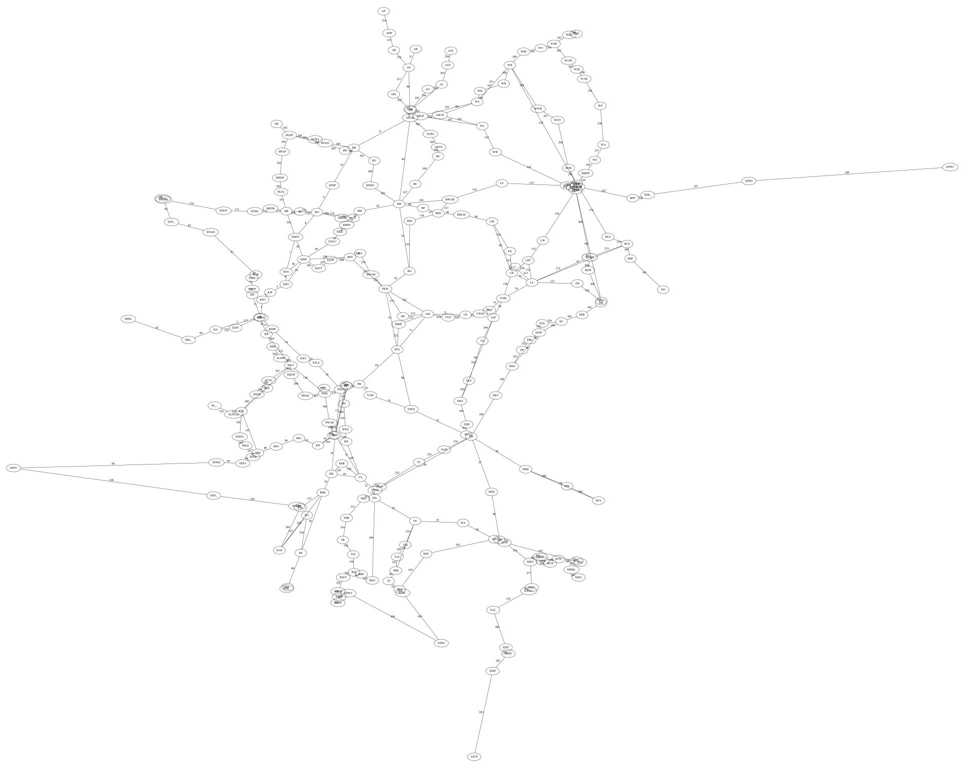


Figure 3 *bahn* dataset.

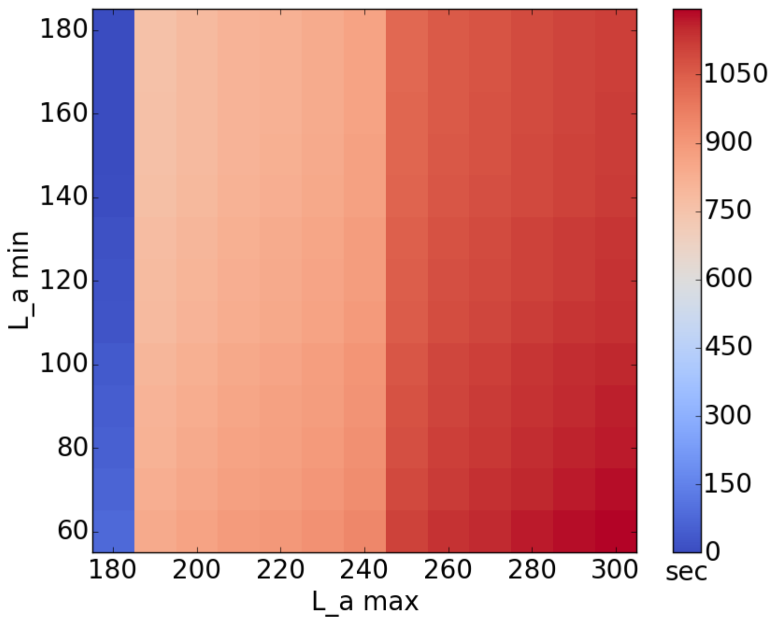


Figure 4 number of infeasible changing activities for a timetable from (DM) for different trickling intervals.

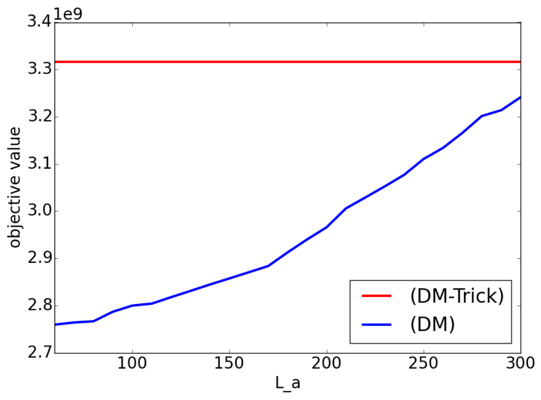


Figure 5 objective values for (DM) for different L_a.