# Stochastic Delay Prediction in Large Train Networks*

**Annabell Berger[1], Andreas Gebhardt[1],**
**Matthias Müller-Hannemann[1], and Martin Ostrowski[1]**

1 **Institut für Informatik**
**Martin-Luther-Universität Halle-Wittenberg, Germany**
`{berger,gebhardt,muellerh}@informatik.uni-halle.de`
`martin.ostrowski@student.uni-halle.de`

## Abstract

In daily operation, railway traffic always deviates from the planned schedule to a certain extent. Primary initial delays of trains may cause a whole cascade of secondary delays of other trains over the entire network. In this paper, we propose a stochastic model for delay propagation and forecasts of arrival and departure events which is applicable to all kind of public transport (not only to railway traffic). Our model is fairly realistic, it includes general waiting policies (how long do trains wait for delayed feeder trains), it uses driving time profiles (discrete distributions) on travel arcs which depend on the departure time, and it incorporates the catch-up potential of buffer times on driving sections and train stops. The model is suited for an online scenario where a massive stream of update messages on the current status of trains arrives which has to be propagated through the whole network. Efficient stochastic propagation of delays has important applications in online timetable information, in delay management and train disposition, and in stability analysis of timetables.

The proposed approach has been implemented and evaluated on the German timetable of 2011 with waiting policies of Deutsche Bahn AG. A complete stochastic delay propagation for the whole German train network and a whole day can be performed in less than 14 seconds on a PC. We tested our propagation algorithm with artificial discrete travel time distributions which can be parametrized by the size of their fluctuations. Our forecasts are compared with real data. It turns out that stochastic propagation of delays is efficient enough to be applicable in practice, but the forecast quality requires further adjustments of our artificial travel time distributions to estimates from real data.

**1998 ACM Subject Classification** G.2.2 Graph Theory (Graph algorithms; Network problems)

**Keywords and phrases** Stochastic Delay Propagation, Timetable Information, Delay Management, Train Disposition, Stability Analysis

**Digital Object Identifier** 10.4230/OASIcs.ATMOS.2011.100

## 1 Introduction

**Motivation.** Train delays occur for various reasons: Disruptions in the operations flow, accidents, malfunctioning or damaged equipment, construction work, repair work, and extreme weather conditions like snow and ice, floods, and landslides, to name just a few. Initial delays of these types are called primary delays. They usually induce a whole cascade

[*] This work was supported by the DFG Focus Program Algorithm Engineering, grant MU 1482/4-2. We wish to thank Deutsche Bahn AG for providing us timetable data for scientific use.

11th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems.
Editors: Alberto Caprara & Spyros Kontogiannis; pp. 100–111
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

of secondary delays of other trains which have to wait according to certain waiting policies between connecting trains. On a typical day of operation of German Railways, an online system has to handle millions of forecast messages about (mostly small) changes with respect to the planned schedule and the latest prediction of the current situation. Thus, a graph model representing the current schedule has to be updated at a high rate [9]. Delay cascades cannot be forecast exactly due to several stochastic influences. For example, trains can drive faster than planned or stay shorter at stations than scheduled and so catch up some of their delay. In fact, to make the schedule more robust, certain slacks are usually integrated into the planned schedule. Stochastic forecasts can be used for several purposes:

1. **Ontrip timetable information:** The arrival and departure time distribution can be used to evaluate the reliability of a planned transfer and then used in a multi-criteria setting as an additional objective.
2. **Delay management and train disposition:** Dispatchers have to decide whether a train should wait for another delayed train. These decisions are quite complex, and so it is helpful to evaluate the reliability of forecasts of arrival and departure times as a decision aid. This information can be used for explicit human decisions or in an automatic disposition system which tries to find globally optimal waiting decisions.
3. **"Stability analysis" of the planned schedule:** Stochastic simulations of delays allow for a quantitative evaluation how small delays propagate through the system. They help to study the robustness of the schedule.

**Related Work.** Efficient deterministic propagation of primary and secondary delays has been done by Müller-Hannemann and Schnee [9]. They demonstrated that even massive delay data streams can be propagated instantly, making this approach feasible for real-time multi-criteria timetable information. Goverde [6] recently presented an efficient deterministic delay propagation algorithm for periodic timetables. Train event networks are similar to project networks. In stochastic project networks (PERT-networks), the vertices are project events and arcs correspond to activities. The duration of each activity has an associated probability distribution. One is typically interested in critical paths or in the distribution function of the overall project completion time. The computation of the distribution function is computationally hard, even the evaluation at a single point is #P-complete in general [7]. Stochastic models for the propagation of delays have been studied intensively, most importantly by Carey and Kwieciński [3, 4] and Meester and Muns [8]. They propose to use approximations of delay distributions to reduce the computational effort and study the error propagation for such approximations. However, they do not model waiting policies for connecting trains. For the computation of propagated delays, the distributions are treated as if they were independent. Although it is difficult to bound the consequences of the independence assumption quantitatively, Meester and Muns argue that the effect of their approximations is small. The experimental evaluation of [8] has been conducted on a "toy network" of ten stations and four train lines. A similar approach has been taken by Büker [2]. Compared to our work, his experiments are only based on relatively small subnetworks. Goverde [5] uses a max-plus algebra approach for stability analysis of railway timetables. See also the PhD-thesis of Yuan [10] for further references and an in-depth discussion of models.

**Our Contribution.** We present in the following section a concise and realistic stochastic model for delay propagation and calculation of arrival and departure time distributions in public transport. Our model is formulated with respect to an event graph which models the train schedule and the waiting conditions between planned transfer possibilities. It includes general waiting policies (how long do trains wait for delayed feeder trains), it uses driving time profiles (discrete distributions) on travel arcs which depend on the departure time, but also on train category or track conditions. Moreover, our model incorporates the catch-up potential of buffer times on driving sections and at train stops. We believe that the resulting model is quite elegant which made it possible to implement it with a reasonable effort.

Discrete distributions of travel times on travel arcs can be chosen arbitrarily which allows to test different scenarios, in particular to stress the system to its limits. A crucial property of our approach is that it allows dynamic updates with respect to new delay messages. Given incoming messages (new delays or updates of existing delays, and current effective status messages of trains) from some external source, we immediately propagate these messages through the whole network. The event graph is a directed acyclic graph. Therefore, delay propagation can be done in topological order of events. For start up it is necessary to propagate once the initial distributions over the whole event graph. Afterwards new forecasts and effective status messages are only propagated in the forward cone of the corresponding event, i.e. in the part of the network which can be reached from it. We work with two types of distributions: one-point distributions of already realized events and arbitrary discrete distributions of events which still lie in the future.

Although stochastic delay propagation is computationally quite expensive, we managed to implement a version which is fast enough to be used in an online system. Experiments with a prototypal implementation on the whole German train network and realistic waiting rules between connecting trains require less than $14s$ to propagate all discrete distributions for a full traffic day. Simulations with several distributions of travel times on travel arcs yield interesting insights into the robustness of the planned schedule against small fluctuations. We compare our predictions with realized event times for two different types of days, a mid-week day and a weekend day and perform experiments with four different sets of waiting rules between connecting trains.

**Overview.** In the following section we describe in detail the event graph, our stochastic model, and its underlying assumptions. Afterwards we explain, how arrival and departure probabilities can be computed for all events. In Section 4, we report on experimental results with a prototypal implementation. A full version of this paper is available as a Technical Report [1].

## 2 The stochastic model

### 2.1 The timetable and its corresponding event graph

A time table $TT := (P, S, C)$ consists of a tuple of sets. Let $P$ be the set of trains, $S$ the set of stations and $C$ the set of elementary connections, that is $C := \{c = (p, s, s', t_d, t_a) \mid \text{ train } p \in P \text{ leaves station } s \text{ at time } t_d. \text{ The next stop of } p \text{ is at station } s' \text{ at time } t_a\}$.

We define with respect to the set of elementary connections $C$ sets of departure events $Dep_v$ and arrival events $Arr_v$ for each station $v \in S$. Let $Dep = \cup_{v \in S} Dep_v$ and $Arr = \cup_{v \in S} Arr_v$. Each event $dep_v := (time, train) \in Dep_v$ and $arr_v := (time, train) \in Arr_v$ represents exactly one departure or arrival event which consists of the two attributes *time* and *train*. Staying times at a station $v$ can be lower and upper bounded by minimum and maximum staying times $minstay(arr_v, dep_v), maxstay(arr_v, dep_v) \in \mathbb{Z}^+$ which have to be respected between different events in $v$. Staying times ensure the possibility to transfer from one train (the so-called *feeder train*) to the next. We denote by $G := (V, A)$ the *event graph* with $V := Dep \cup Arr$ and the arc set $A := A_{travel} \cup A_{transfer}$ consisting of the *travel arc set* $A_{travel} :=$

$$\{(dep_v, arr_w)|\text{ there exists } c \in C \text{ with } \ t_d = dep_v(time), t_a = arr_w(time),$$
$$v = s, w = s' \ \wedge \ p = dep_v(train) = arr_w(train)\}$$

and the *transfer arc set*

$$A_{transfer} := \{(arr_v, dep_v)|\ arr_v \in Arr, dep_v \in Dep, minstay(arr_v, dep_v) \leq$$
$$dep_v(time) - arr_v(time) \leq maxstay(arr_v, dep_v)\}.$$

Furthermore, we define waiting times $wait_{transfer} : A_{transfer} \mapsto \mathbb{Z}^+ \cup \{\infty\}$ where we denote by $wait_{transfer}(arr_v, dep_v)$ the number of time units which $train(dep_v)$ may depart later than the planned time $time(dep_v)$ with respect to its feeder train $train(arr_v)$. Clearly, $wait_{transfer}(arr_v,\ dep_v) = \infty$ if $train(arr_v) = train(dep_v)$, because a train cannot depart before its arrival. We define a further waiting time $wait : Dep \mapsto \mathbb{Z}^+$ with $wait(dep_v) := \max\{wait_{transfer}(arr_v, dep_v)| \quad (arr_v, dep_v) \in A_{transfer} \wedge train(arr_v) \neq train(dep_v)\}$. If some train is delayed by more than $wait(dep_v)$, then its departure time depends on no other train, irrespectively of their delays. Each travel arc $(dep_v, arr_w) \in A_{travel}$ possesses a scheduled travel time $arr_w(time) - dep_v(time)$ and a minimum possible travel time $mintt(dep_v, arr_w) \in \mathbb{Z}^+$ with $mintt(dep_v,\ arr_w) \leq arr_w(time) - dep_v(time)$. If train $train(dep_v)$ departs too late at $v$ there exists the possibility to regain some time. We define a *realization time* $t_r(event)$ for each event and call the current time point *update time* $t_{update}$. Note that *scheduled time points* (see the attributes attached to departure or arrival events) are denoted as 'time'.

## 2.2 Model assumptions

In the following, we specify and discuss our model assumptions. The general scenario is that we obtain a stream of online messages about the delay status of trains (so-called *status messages*) from the railway company, i.e., for each train, the difference between the scheduled and the realization time for departure and arrival events is measured and reported.

▶ **Assumption** 1. *With respect to status messages, a train can arrive or depart at any time after the planned arrival or departure time, respectively.*

Of course, a train shall never depart before its scheduled departure time. In reality, a train may arrive somewhat early, but then its waiting time at the station will be increased. Thus our model assumption does not make a difference for delay propagation, but simplifies the mathematical model. For compatibility to Assumption 1, we demand the following.

▶ **Assumption** 2. *With respect to our forecasts of arrival and departure time distributions, no train departs before its scheduled time or arrives at a station before its planned arrival time.*

▶ **Assumption** 3. *We assume that the distributions of arrival times of all feeder trains of a given train are stochastically independent.*

In other words, we postulate that the delay distributions of any two feeder trains are mutually independent. Note that the same independence assumption has also been used in the previous studies mentioned in the related work section above. However, we would like to emphasize one crucial point in online delay propagation: as soon as a delay of some train has been realized, the corresponding departure or arrival time distribution of this event is replaced by a one-point distribution, and this update is propagated through the network. Hence, the contribution of *realized delays* is fully reflected in our estimates of future arrival and departure time distributions. Nevertheless, our independence assumption may be violated to a certain extent, for example, because of limited track capacities for incoming trains at a station. However, this simplification enables us to keep stochastic delay propagation tractable.

▶ **Assumption** 4. *Waiting rules are defined for any pair of arriving and departing trains for which a transfer arc is defined.*

For simplicity, we do not model new transfer possibilities due to other delayed trains (although it would not change our model, only the implementation is slightly more complicated).

## 3 Departure and arrival probabilities

### 3.1 Travel time, departure and arrival random variables

Let $(\Omega, \mathcal{A}, P)$ be a discrete probability space with sample space $\Omega$, $\sigma$-algebra $\mathcal{A}$ and probability measure $P$. Furthermore, let $T \subset \mathbb{Z}^+$ be a discrete set of time points. We define with respect to a current time $t_{update}$ for each event $event \in Dep \cup Arr$ a discrete random variable $X_{event} : \Omega \mapsto \{event(time), event(time)+1, \dots\}$. We call a variable *departure random variable* if $event = dep_v$ and *arrival random variable* for $event = arr_v$ where $dep_v, arr_v \in Dep \cup Arr$. The range of $X_{event}(\Omega)$ is $\{event(time), event(time)+1, \dots\}$ by our Assumption 2. With respect to Assumption 3, we state that all arrival random variables $X_{arr_v}$ with $arr_v \in Arr$ for a single station $s \in S$ are pairwise stochastically independent with respect to probability measure $P$. This means that for all pairs $(t, t') \in \{arr_v(time), \dots\} \times \{arr'_v(time), \dots\}$ it follows that

$$P(X^{-1}_{arr_v}(\{t\}) \cap X^{-1}_{arr'_v}(\{t'\})) = P(X^{-1}_{arr_v}(\{t\})) \cdot P(X^{-1}_{arr'_v}(\{t'\})).$$

Furthermore, we distinguish between realized and not realized random variables. For all realized events we state $P(X^{-1}_{event}(t_r(event))) := 1$ (in such cases $t_r(event) \leq t_{update}$). Non-realized events are in general not 'one-point-distributed'.

We also need a random variable which describes possible travel times on each arc $(dep_v, arr_w) \in A_{travel}$. Generally, we want to model the case that a train can regain some time with a smaller travel time as the planned travel time $arr_w(time) - dep_v(time)$. Assumption 2 ensures that we may not arrive at an earlier time as $arr_w(time)$. Hence, we need for each arc $(dep_v, arc_w) \in A_{travel}$ a *sequence of discrete travel time variables* $(X^t_{(dep_v, arc_w)})_{t \in TP}$ for each possible departure time point $t \in TP := \{dep_v(time), dep_v(time)+1, \dots\}$ with

$$X^t_{(dep_v, arr_w)} : \Omega \mapsto \{mintt(dep_v, arr_w), \dots, arr_w(time) - dep_v(time) + k\}.$$

To satisfy Assumption 2, we have to distinguish random variables for different times with respect to their time distance to the scheduled times. That means that the probability for time $t$ must be zero if the distance between a forecasted time and the scheduled arrival time $arr_w(time)$ is more than $mintt(dep_v, arr_w)$. We set $P((X^t_{(dep_v, arr_w)})^{-1}(\{mintt(dep_v, arr_w), \dots, arr_w(time) - t - 1\})) := 0$ for all $t \in \{dep_v(time), \dots, d\}$ because our Assumption 2 prohibits to arrive earlier than planned. Clearly, it is necessary to model for all points in time $t$ a distinct random variable $X^t_{(dep_v, arr_w)}$. In theory, we are able to distinguish infinitely many of such random variables. In our experiments, we restrict ourselves to the case where all random variables are identical from a certain point of time $d$ onwards. We set $d := arr_w(time) - mintt(dep_v, arr_w) - 1$ and define

$$X^t_{(dep_v, arr_w)} := X^{d+1}_{(dep_v, arr_w)}$$

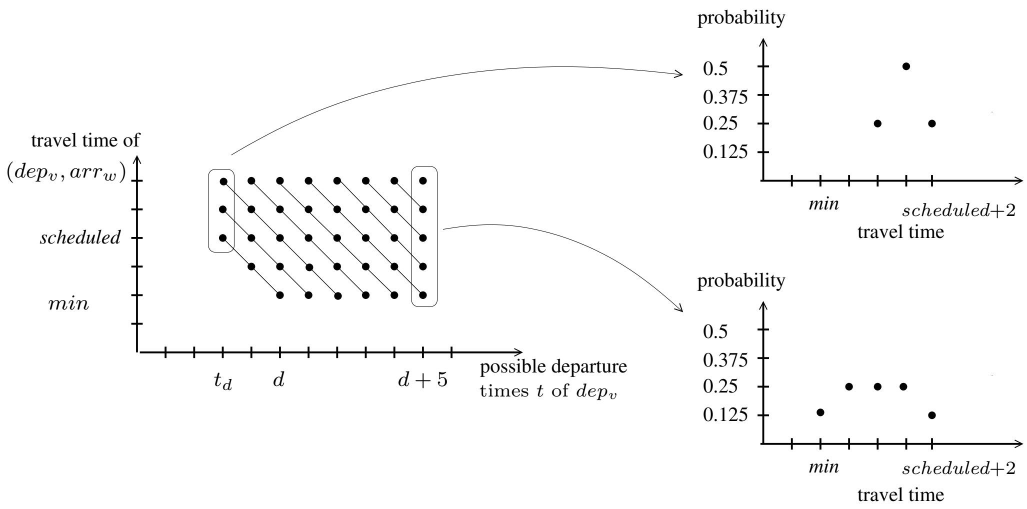

**Figure 1** Possible travel times on an arc $(dep_v, arr_w)$ depending on the actual departure time. We use the the abbreviations $t := dep_v(time)$, $d := arr_w(time) - mintt(dep_v, arr_w)$, $min := mintt(dep_v, arr_w)$ and scheduled:= $arr_w(time) - dep_v(time)$. The allowed fluctuation above the scheduled travel time is here chosen as $k = 2$. The data points connected by lines represent all travel times which lead to the same arrival time at $w$. The points in the same column $t_d$ correspond to all possible travel times for a fixed distribution $X^t_{(dep_v, arr_w)}$.

for all $t > d$. Consider Figure 1 for an example of travel time distributions.

## 3.2 Departure random variables and departure probabilities

When we reach the current time point $t_{update}$, we replace for all events with a realization time $t_r(event) \leq t_{update}$ their discrete departure or arrival random variables with the above defined 'one-point-distribution' (if the data is available). Afterwards, we can compute — following a topological ordering of the acyclic event graph — all succeeding random variables. In a next step we want to describe how one can compute the departure random variable for an event which has not yet been realized. For the determination of a departure random variable we distinguish between three cases.

1. Train $train(dep_v)$ departs at its scheduled time $dep_v(time)$.
2. Trains $train(dep_v)$ departs at $t \in \{dep_v(time) + 1, \dots, dep_v(time) + wait(dep_v)\}$.
3. Train $train(dep_v)$ departs at $t \in \{dep_v(time) + wait(dep_v) + 1, \dots\}$.

We denote the set of all arrival events of feeder trains by $F := \{arr^i_v | \; (arr^i_v, dep_v) \in A_{transfer},\; train(arr^i_v) \neq train(dep_v)\}$. Case (1) occurs if train $train(arr_v)$ arrives at $t \in \{arr_v(time), \dots, dep_v(time) - minstay(arr_v, dep_v)\}$ and for each feeder trains $arr^i_v \in F$ with $i \in \mathbb{N}_{|F|}$ one of the following holds: (a) either it arrives early enough so that the train can depart on time, i.e., it arrives in time interval $\{arr^i_v(time), \dots, dep_v(time) - minstay(arr^i_v, dep_v)\}$, or (b) it arrives so late that the departing train does not need to take care of it. This happens in the interval $\{dep_v(time) - minstay(arr^i_v, \dots, dep_v) + wait(arr^i_v, dep_v) + 1, \dots\}$. Let $l := |F| + 1$ the number of ingoing transfer arcs for departure event $dep_v$ and $arr^l_v := arr_v$. We define for all feeder trains $i \in \mathbb{N}_{l-1}$ and $t \in \{dep_v(time), \dots\}$ possible arrival intervals $I_i(t)$ depending on possible departure times of train $train(arr_v)$ with $I_i(t) := \{arr^i_v(time), \dots, t - minstay(arr^i_v, dep_v) - 1\} \cup \{dep_v(time) - minstay(arr^i_v, dep_v) + wait(arr^i_v, dep_v) + 1\}$ and $I_l(t) :=$

$\{arr_v(time), \dots, t - minstay(arr_v, dep_v) - 1\}$. Furthermore, we need slightly different index sets $J_i(t) := \{arr_v^i(time), \dots, t - minstay(arr_v^i, dep_v)\} \cup \{dep_v(time) - minstay(arr_v^i, dep_v) + wait_{transfer}(arr_v^i, dep_v) + 1, \dots\}$ and $J_l(t) := \{arr_v(time), \dots, t - minstay(arr_v, dep_v)\}$. Finally, we denote $m_i := minstay(arr_v^i, dep_v)$. For case (1) we compute the preimages of a departure random variable with

$$X_{dep_v}^{-1}(\{dep_v(time)\}) = \bigcap_{i=1}^{l} X_{arr_v^i}^{-1}(J_i(dep_v(time))).$$

By Assumption 3, and if we denote $p_{arr_v^i}(t) := P(X_{arr_v^i}^{-1}(\{t\}))$ and $p_{dep_v}(t) := P(X_{dep_v}^{-1}(\{t\}))$ we get

$$p_{dep_v}(dep_v(time)) = \prod_{i=1}^{\lambda} \sum_{\lambda \in J_i(dep_v(time))} p_{arr_v^i}(\lambda).$$

We call $p_{dep_v}$ the *departure probability* and $p_{arr_v}$ the *arrival probability*.

Case (2) occurs if train $train(arr_v)$ arrives in interval $\{arr_v(time), \dots, t - minstay(arr_v, dep_v)\}$ and at least one feeder train $train(arr_v^{i_0})$ with $arr_v^{i_0} \in F$ arrives *exactly* at time point $t - minstay(arr_v^{i_0}, dep_v)$. We define with respect to possible departure times $t \in \{dep_v(time) + 1, \dots, dep_v(time) + wait(dep_v)\}$ the *set of all 'exact' time point tuples* as

$$A_t := \{(t_1, \dots, t_l) | \ (t_1, \dots, t_l) \in (\times_{i=1}^{l} J_i(t)) \wedge \ \exists i_0 < l \text{ with } t_{i_0} = t - m_{i_0}\}.$$

For a departure random variable in case (2) we get

$$X_{dep_v}^{-1}(\{t\}) = \bigcup_{(t_1, \dots, t_l) \in A_t} \left( \bigcap_{i=1}^{l} X_{arr_v^i}^{-1}(\{t_i\}) \right).$$

This formulation is compact but we have to consider exponentially many disjoint subsets of $\Omega$ leading to a non-efficient algorithm for computing $X_{dep_v}$. Instead, we rearrange these preimages by applying the well-known 'De Morgan-rules' such that we get only polynomially many disjoint subsets of $\Omega$. Then, we get

$$X_{dep_v}^{-1}(\{t\}) = \bigcup_{j=0}^{l-1} (\bigcap_{i=1}^{j} X_{arr_v^i}^{-1}(I_i(t))) \bigcap (X_{arr_v^{j+1}}^{-1}(\{t - m_{j+1}\})) \bigcap_{i=j+2}^{l} X_{arr_v^i}^{-1}(J_i(t))) =: \bigcup_{j=0}^{l-1} S_j.$$

Using $\sigma$-additivity to compute the elementary probabilities $p_{dep_v}(t) := P(X_{dep_v}^{-1}(\{t\}))$ we have to show that for all pairs $j, j' \in \{0, \dots, k-1\}$ the sets $S_j, S_{j'}$ are disjoint. It is sufficient to prove that for an arbitrary $j_0$ the sets $S_{j_0}$ and $S_{j_0+1}$ are pairwise disjoint. Assume there exists an $\omega \in \Omega$ with $\omega \in S_{j_0} \cap S_{j_0+1}$. Then it follows that $X_{arr_v^{j_0+1}}(\omega) = t - m_{j_0+1}$ and $X_{arr_v^{j_0+1}}(\omega) \in I_{j_0+1}$. Because $t - m_{j_0+1} \notin I_{j_0+1}$ this is a contradiction. Hence, we can apply $\sigma$-additivity and use Assumption 3 that our random variables are stochastically independent. For case (2) we obtain

$$p_{dep_v}(t) = \sum_{j=0}^{l-1} \left( \prod_{i=1}^{j} \left( \sum_{\lambda \in I_i(t)} p_{arr_v^i}(\lambda) \right) \cdot p_{arr_v^{j+1}}(t - m_{j+1}) \cdot \prod_{i=j+2}^{l} \left( \sum_{\lambda \in J_i(t)} p_{arr_v^i}(\lambda) \right) \right).$$

Case (3) is much simpler because the departure time of train $train(dep_v)$ only depends on its arriving time $arr_v(time)$. That is $X_{dep_v}^{-1}(\{t\}) = X_{arr_v}^{-1}(\{t - m_l\})$ and results in $p_{dep_v}(t) = p_{arr_v}(t - m_l)$.

The case that train $train(dep_v)$ starts at station $v \in S$ is also simpler. Obviously, its departure time only depends on feeder trains. We can take all above computations but ignore the arrival event $arr_v^k$ in case (1) and case (2). For case (3) we set $p_{dep_v}(t) := 0$ for all $t \in \{dep_v(time) + wait_{transfer}(dep_v) + 1, \dots\}$.

### 3.3 Arrival random variables and arrival probabilities

Let $(dep_v, arr_w) \in A_{travel}$ be a travel arc. The arriving time on $w$ depends on the departure time at $v \in S$ and all possible travel times on this travel arc at this time. We denote the *possible travel time set* by $PTT := \{mintt(dep_v, arr_w), \dots, arr_w(time) - dep_v(time) + k\}$ with $k \in \mathbb{N}$. Formally, we get for each $t \in \{arr_v(time), \dots\}$

$$X_{arr_w}^{-1}(\{t\}) = \bigcup_{j \in PTT} (X_{dep_v}^{-1}(\{t-j\}) \bigcap (X_{(dep_v, arr_w)}^{t-j})^{-1}(\{j\}).$$

We can apply $\sigma$-additivity to probability measure $P$. We set $p_{(dep_v, arr_w)}^t(\lambda) := P(X_{(dep_v, arr_w)}^t)(\lambda)$ and get the *arrival probability* for an arrival event $arr_w$ at time $t$ as

$$p_{arr_w}(t) = \sum_{j \in PTT} p_{dep_v}(t-j) \cdot p_{(dep_v, arr_w)}^{t-j}(j).$$

## 4 Experiments

**Test instances and environment.** Our computational study is based on the German train schedule of 2011, with actual data of realized departure and arrival times for days in February and March 2011. Each day of operation has about 300,000 departure and arrival events per day. All experiments were run on a PC (Intel(R) Xeon(R), 2.93GHz, 4MB cache, 47GB main memory under ubuntu linux version 8.10). Only one core has been used by our program. Our code is written in C++ and has been compiled with g++ 4.4.3 and compile option -O3.

**Delay distributions on travel arcs.** For our simulation experiments we use two types of distributions: a uniform distribution and a kind of unimodal distribution with a peak at the scheduled travel time. Our unimodal distribution is parametrized by $k$ which controls the support size. For parameter $k$ the support has width $2k + 1$, and the distribution assigns the probabilities $\frac{1}{2^{k+1}}, \frac{1}{2^k}, \dots, \frac{1}{2^2}, 1 - \sum_{i=1}^{k} \frac{1}{2^i}, \frac{1}{2^2}, \frac{1}{2^3} \dots, \frac{1}{2^{k+1}}$ to the travel times $mintt, mintt + 1, \dots, s, s + 1, \dots, s + k$, where $s$ denotes the scheduled travel time.

We select the travel time distribution of a travel arc depending on the actual departure time $dep_v(time)$. If the departure time at departure event $dep_v$ is between the scheduled time $dep_v(time)$ and $arr_w(time) - mintt(dep_v, arr_w)$ we apply the uniform distribution. If the actual departure time is above $arr_w(time) - mintt(dep_v, arr_w)$, we always apply the unimodal distribution.

**Waiting rules.** For the waiting times $wait_{transfer}$ we use four different scenarios:

1. **rule-based:** We use the standard waiting rules from German Railways.
2. **always:** Each train has to wait for all of its feeder trains.
3. **never:** No train has to wait for another train.
4. **static:** Whenever necessary, a train has to wait for a feeder train exactly $x$ minutes. We set $x := 5$ and $wait_{transfer}(arr_v, dep_v) = 5$.

| | $k = 1$ | $k = 2$ | $k = 3$ | $k = 4$ | $k = 5$ |
|---|---|---|---|---|---|
| never | $3.72s$ | $6.23s$ | $6.07s$ | $6.98s$ | $11.32s$ |
| rule-based | $3.73s$ | $6.30s$ | $6.93s$ | $7.02s$ | $11.52s$ |
| static | $4.12s$ | $6.32s$ | $7.01s$ | $8.95s$ | $11.94s$ |
| always | $5.12s$ | $7.29s$ | $8.57s$ | $10.99s$ | $13.78s$ |

**Table 1** Running times in seconds for the different waiting strategies and travel time distributions.

**Experiment 1: Efficiency.** In our experiments we use the fluctuation parameter $k \in \{1, 2, 3, 4, 5\}$ for travel times, i.e., the maximum permitted additional travel time for each train between two stops. The exact definition can be found in Subsection 3.1. The running time for the computation of all arrival and departure distributions of a whole day takes only a few seconds, see Table 1. Hence, we are able to determine forecasts in real time. Two trends are obvious: the wider the travel time distributions (increasing fluctuation parameter $k$), the larger the running time. Likewise, when we compare the rules among each other, we observe that the more trains have to wait for each other, the larger the running time. However, in all cases the absolute running times are below $14s$.

**Experiment 2: Width of distributions over time.** Our second experiment has been guided by the following questions:

1. How precise are our forecasts, i.e. how narrow or wide are the computed distributions? The smaller the support of the distribution, the more meaningful is our forecast.
2. To which extent does the distribution width (support) depend on the chosen waiting rule? This gives us insight into the stability of a rule and also may explain the observed differences in CPU time. The wider a distribution becomes during propagation, the more work has to be done and the less stable a waiting rule will behave.
3. Do the distribution widths grow over time, and how does this depend on the chosen waiting rule? For the extreme waiting rule "always" we may expect a cascading effect, while for the other three rules the slack times within the schedule and the bounds on the maximum waiting time may have a weakening and stabilizing effect on the support widths.

In Figure 2, we investigate the widths of all supports with respect to the time horizon and all four waiting rules on Thursday, 10.03.2011. For the delay distributions on travel arcs we used the fluctuation parameter $k = 2$. For the waiting strategies "rule-based", "never" and "static", we observe that the width of the supports of event distributions stays relatively narrow over time, while for the extreme waiting policy "always" the widths of supports grow fast with an increasing time horizon. These findings also partially explain the observed running times for the different waiting rules as shown in Table 1.

**Experiment 3: Predictions vs. realized data.** This experiment investigates how well our predictions fit to realized data. In this experiment, we computed our predictions *without* using any information about actual delays. Since real operations have been conducted approximately according to the "rule based" waiting rule, we compare our predictions with this rule.

For the comparison of our predictions with realized data we use two different test days, namely a Thursday and a Sunday. In Table 2, we give an overview about the data availability for both days. This is necessary, because we did not get all realized event times from German Railways. For about 30% of the regional trains we have no information about their realized departure or arrival times. For the 10.03.2011 we have collected $289,459$ messages

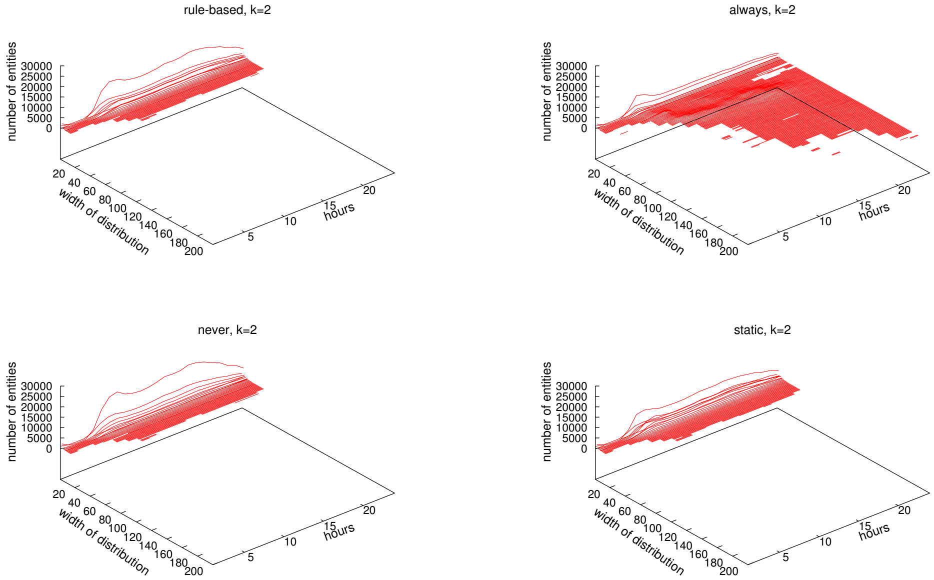

**Figure 2** The three-dimensional plots show how the predicted arrival and departure time distributions change over time, from early morning to midnight for the traffic day 10.03.2011 (24 hours). For each point in time, the z-axis gives the number of events which have a distribution with a certain support width. We compare the different waiting rules for the fluctuation parameter $k = 2$. Waiting rules: "rule based" (upper left) and "always" (upper right), "never" (lower left) and "static" (lower right).

about realized event times and for the 20.02.2011 we got $193,461$ messages. According to information from German Railways, we may assume that the rest of non-available messages were "in due time", what here means that all these trains have at most 1 minute of delay. With respect to this data situation, we determined the absolute difference between realized timestamps (of observed real world data) and the planned schedule time to measure the "strength of delays" on these days, see Figure 3.

In Figure 4, we display the absolute differences between the expectation values and the realized times for 289459 available events on Thursday 10.03.2011 and for 193461 available events on Sunday, 20.02.2011. With respect to expectation values, the difference to the realized values is less than 5 minutes in about 66% of all available events on both investigated days. However, a significant number of forecasts is wide off (by 2 hours or more in some cases). In order to interpret the results of this experiment, recall that our computation of arrival and departure time distributions is based on the pure published schedule only, it does not incorporate actual delays. Without information about actual delays these heavy tails of

| date | available event data | non-available but presumably in due time event data | non-available event data without any information |
|---|---|---|---|
| 20.02.2011 | 45 | 38 | 17 |
| 10.03.2011 | 51 | 44 | 5 |

**Table 2** Data availability with respect to all events. Percentage of available and non-available data. A fraction of non-available data can be assumed to be "in due time" (3rd column.)

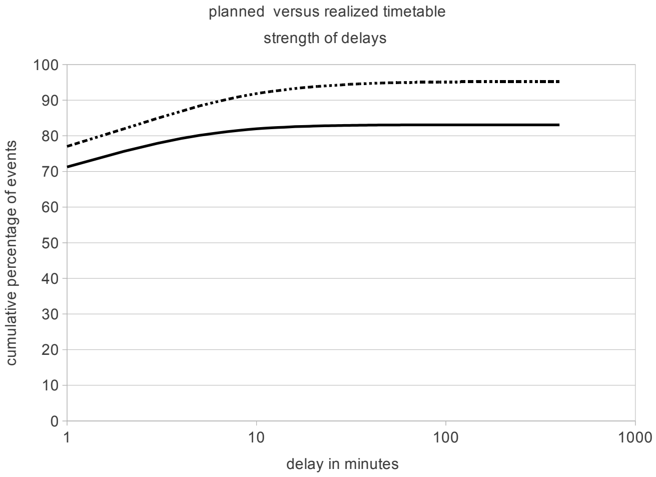

**Figure 3** The cumulative curves show the percentage fraction of events with a realized delay of at most $x$ minutes for two different days, 10.03.2011 (above, dotted line: a Thursday) and 20.02.2011 (below: a Sunday).

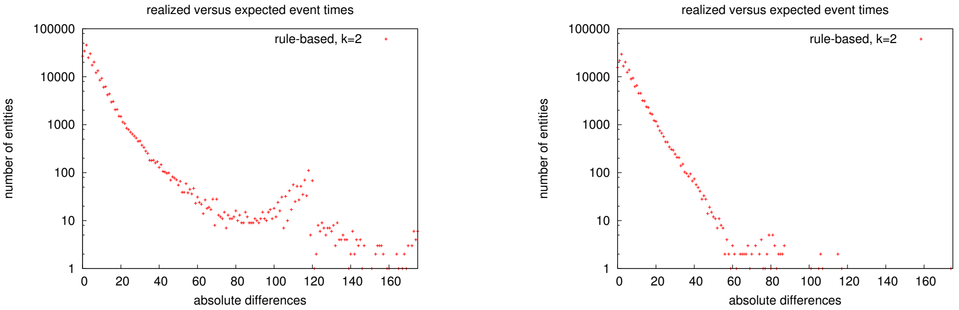

**Figure 4** Comparison of the absolute differences from the expected and realized timestamps on 10.03.2011 (left) and on 20.02.2011 (right) with fluctuation parameter $k = 2$ for travel time distributions.

large differences between expectation values for predictions and the corresponding realized values are more or less unavoidable. The travel time distributions used in our model are designed to capture small delays, they are not capable to predict large source delays (for example, that a trains is delayed by two hours because of a defect of the engine). On the positive side, small fluctuations are seemingly captured quite well.

**Experiment 4: Predictions over time.** In contrast to the previous experiment, we now incorporate all delays which occurred before 11:59 a.m. on several test days and make on this basis predictions for the next four hours. These predictions are then compared with the realized data. Our hypothesis is that the fit of our predictions should decrease the further in the future we look, which is confirmed in Figure 5. On five test days, we observe a small average increase, ranging from 4 minutes difference when looking 30 minutes ahead to less than 7 minutes 4 hours ahead. Thus, the accuracy of prediction is already quite good and degrades only slowly when we look into the forthcoming hours. We believe that this is good news for applications in real-time timetable information.

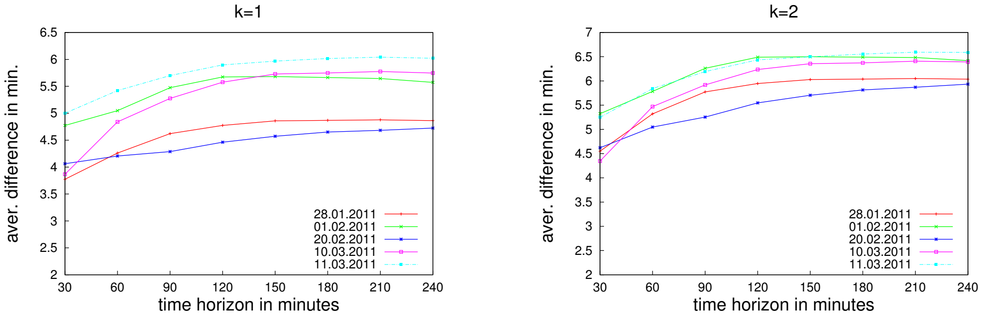

**Figure 5** Based on actual delays before 11:59 a.m., we compute the distributions of events occurring in the next four hours. We show how the average distance of the expectation values of our predicted event time distributions from the realized delay data evolves over time.

## 5 Conclusions

We have presented a stochastic model for delay propagation in large transportation networks. This model turns out to be fast enough for an online scenario with massive streams of update messages. In our experiments we worked with simple artificial distributions for travel time fluctuations (in the absence of real distributions). The next step is to replace these distributions by empirical distributions from collected statistical data over several months. We expect that empirical distributions will enable us to generate significantly tighter predictions.

## References

1 A. Berger, A. Gebhardt, M. Müller-Hannemann, and M. Ostrowski. Stochastic delay prediction in large train networks. Technical Report 2011/1, Institut für Informatik, Martin-Luther-Universität Halle-Wittenberg, 2011.

2 T. Büker. *Ausgewählte Aspekte der Verspätungsfortpflanzung in Netzen.* PhD thesis, RWTH Aachen, Germany, 2010.

3 M. Carey and A. Kwieciński. Stochastic approximation to the effects of headways in knock-on delays of trains. *Transportation Research Part B*, 28:251–267, 1994.

4 M. Carey and A. Kwieciński. Properties of expected costs and performance measures in stochastic models of scheduled transport. *European Journal of Operational Research*, 83:182–199, 1995.

5 R.M.P. Goverde. Railway timetable stability analysis using max-plus system theory. *Transportation Research Part B*, 41:179–201, 2007.

6 R.M.P. Goverde. A delay propagation algorithm for large-scale railway traffic networks. *Transportation Research Part C*, 18:269–287, 2010.

7 J. N. Hagstrom. Computational complexity of PERT problems. *Networks*, 18:139–147, 1988.

8 L. E. Meester and S. Muns. Stochastic delay propagation in railway networks and phase-type distributions. *Transportation Research Part B*, 41:218–230, 2007.

9 M. Müller-Hannemann and M. Schnee. Efficient timetable information in the presence of delays. In R. K. Ahuja, R. H. Möhring, and C. D. Zaroliagis, editors, *Robust and Online Large-Scale Optimization*, volume 5868 of *LNCS*, pages 249–272. Springer, 2009.

10 J. Yuan. *Stochastic modeling of train delays and delay propagation in stations.* PhD thesis, Technische Universiteit Delft, The Netherlands, 2006.