Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

VALIDATION AND QUANTIFICATION OF POSSIBLE MODEL EXTENSIONS FOR A RAILWAY OPERATIONS MODEL USING DELAY DATA DISAGGREGATION

Nadine Schwab[1,2], Matthias Rößler[2], Hannah Kastinger[2], Günter Schneckenreither[2,3], Matthias Wastian[2], and Niki Popper[1,2,3]

[1]Institute for Information Systems Engineering, TU Wien, Vienna, AT-9, AUSTRIA
[2]dwh Simulation Services, dwh GmbH, Vienna, AT-9, AUSTRIA
[3]Institute of Statistics and Mathematical Methods in Economics, TU Wien, Vienna, AT-9, AUSTRIA

ABSTRACT

In simulation of railway traffic, it is crucial to incorporate realistic delays. Delay data can be obtained from historical records containing scheduled and actual times of trains. Labelled or unlabelled with delay causes, these data do not consistently indicate whether a delay is secondary (e.g., caused by another delayed train) or primary (i.e., caused by external or unknown events) for a specific simulation application due to missing, inconsistent, and biased data. In a realistic simulation, only primary delays should be injected, whereas secondary delays emerge from the simulated dynamics. The authors developed an approach to differentiate between primary and secondary delays that is flexible in regard to simulated resources, performing a network analysis and using different thresholds for blocking trains. Finally, the disaggregated primary delays were inserted into a simulation model, showing that it can reproduce the actual total delays with reasonable errors.

1 INTRODUCTION

The planning of resources in railway systems is challenging due to their multitude, complexity, and the interactions of infrastructure, vehicles, and personnel. The goal of planning is to ensure efficient and reliable rail operations: Resources are valuable and sometimes rare, but it is of utmost interest that perturbations of the system caused, e.g., by delays or disruptions do not cause more harm than necessary.

In railway transportation systems, an important distinction is made with respect to delays (Rößler et al. 2021). Train-specific delays, e.g., due to problems with rolling stock, weather conditions, or signal malfunctions, are referred to as *primary delays*. Subsequent delays, e.g., due to blocked tracks or delayed resources like locomotives are called *secondary delays*. In terms of strategies for avoidance, the distinction of delays is extremely relevant. While primary delays can only be avoided, if at all, on the micro level (infrastructure, personnel,...), secondary delays may be avoided by more robust planning.

Secondary delays also pose an important limitation to classic optimization routines for planning: Highly optimized schedules are often the most susceptible to bottlenecks due to delay propagation and, therefore, non-practical. A simheuristics using an optimization metaheuristic, responsible for minimizing resource demand, and a stochastic agent-based model (ABM), responsible for in-the-loop robustness assessment was identified (Rößler et al. 2020) to be a successful strategy to explore the corresponding Pareto front. To assess the robustness, the ABM, first described by Rößler et al. (2018), takes an optimized time-table as input, samples random primary delays, and evaluates the emerging secondary delays.

It is clear that the validity of the generated primary delays is essential for a correct evaluation of the plans. To train the stochastic samplers, historic primary delay data would desperately be needed. Available data sets, however, only compare planned and actual operation times, which allow for the naive calculation of total delays. Therefore, ex-post tagging of the delays, i.e., disaggregation of the total delay data into

primary and secondary, is necessary to parameterize the model. Most importantly, the algorithm should be based on the latent network structure implicitly encoded in train schedule data.

To this point, there is little research regarding this specific disaggregation task. In Yamamura et al. (2013), authors disaggregated delay data from an urban railway line in Tokyo, in Wen et al. (2017) and Wen et al. (2020) from a Chinese high-speed railway train track between Wuhan and Guangzhou. Neither of the investigated tracks includes any junctions. The approaches presented in Manitz et al. (2017) and Palmqvist et al. (2023) investigate propagation of delays on an actual railway network, part of Germany and Sweden respectively. Both come with limitations: The prior simplifies the problem to detect a single source of primary delay, the latter aggregates the delay over the whole network. Finally, Ochiai et al. (2019) proposed a method to simplify manual delay labelling with special visual aid. However, considering that the delay data should be automatically disaggregated in their entirety with all sources of primary delay quantified, neither of the presented methods is directly applicable.

The aim of this work is to develop, implement, and test a novel heuristics for disaggregation of total delay data from the entire railway network of a country. The authors, therefore, extend the described state of the art, since the method does not only identify one but all sources of primary delays individually and quantifies them automatically. Notably, the method does not explicitly require a model of the underlying network infrastructure, since it uses the latent network implicitly encoded in the delay data.

It is furthermore shown how the disaggregated primary delay data are used for parametrization of the agent-based simulation model and explore the differences between the corresponding simulated total delays and the historic total delay data. Similarities and differences between the two will contribute to (cross-model) verification and validation (in the sense of *confirmative validation* as defined in Rehman and Pedersen (2012)) of both, the ABM and the novel heuristics.

2 AGENT-BASED MODEL

The used version of the ABM is an extension of the model described in Rößler et al. (2020) implemented in the authors' own simulation framework ABT (Brunmeir et al. 2023) utilizing C# and the .NET8 runtime. The initial model was developed in order to assess the robustness of circulation plans, but has been successfully applied to new areas of application such as robustness analysis of traction unit driver shift plans and the comparison of disruption management strategies, recently. Since prior versions of the model have already been published, only the most important mechanisms and the mentioned extensions will be stated in this work.

The model consists of three main parts:

- Agents (Resources)
- Infrastructure
- Tasks

Within the model, the agents – which represent different resources of the railway operation such as trains, locomotives, or traction unit drivers – have to execute a given schedule of tasks. Each task is assigned a specific infrastructure, which is occupied throughout the execution of the task. Elements of the infrastructure may be stations and other operational control points (OCPs) of the railway system as well as sections of tracks, together forming a network structure. Delays are injected into the system through primary delays that prolong the execution time of a task. Primary delays are typically sampled from distributions based on available historical data. The propagation of delays is caused by two main mechanisms:

- Resource Delays: If an agent is delayed within its schedule, all following tasks are delayed until it can regain the lost time.
- Infrastructure Delays: The infrastructure may have limited capacity. The execution of a task is delayed if the assigned infrastructure has reached its capacity.

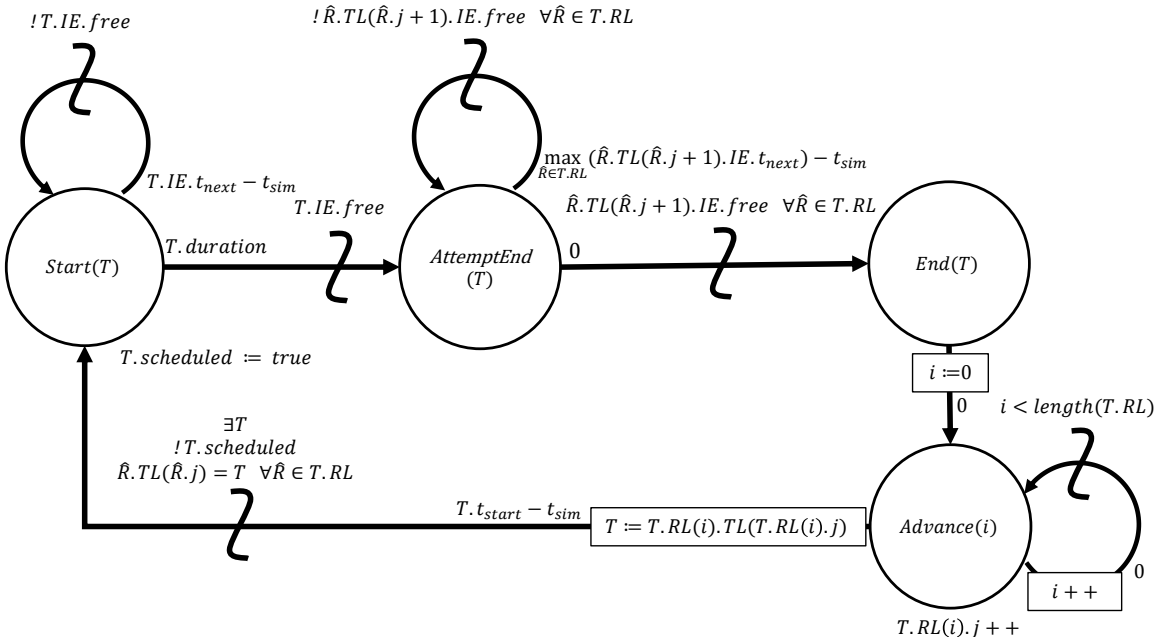


Figure 1: Event graph of the model.

The event graph describing the model is depicted in Figure 1. The ABT framework provides base classes to implement the depicted events and their execution as well as the necessary functionality to perform simulation runs, like a scheduler, a random number generator, and methods to initialize the agents and record the simulation results. In particular, the figure describes the progression of events that are triggered during the execution of a task (t_{sim} represents the current simulation time).

- When the *Start*-event of a task T is triggered, it is checked if the assigned infrastructure $T.IE$ is free ($T.IE.free$), i.e. if the capacity of the infrastructure allows the execution of T. If the infrastructure element is occupied, T may be delayed and the *Start*-event is rescheduled using the next time the infrastructure is free ($T.IE.t_{next}$) as offset.
- If $T.IE$ is free, an *AttemptEnd*-event is scheduled for the task with the duration of the task as offset ($T.duration$). When the event is triggered, it is checked, if the followup-tasks ($R.TL(R.j+1)$) can be executed, i.e., if all the assigned infrastructure elements for the followup tasks are free. There may be more than one, if the task lists of the resources differ. If one of the followup tasks is blocked, the *AttemptEnd*-event is rescheduled, otherwise an *End*-event is scheduled immediately. Through this mechanism, congestions are modeled, as an infrastructure is only released if it has somewhere to go.
- When the *End*-event is triggered, it advances the task list of each of its resources and schedules the *Start*-events of the new current tasks of each resource. The iteration through the list of resources ($T.RL$) is depicted as an *Advance*-event in the figure. The *Start*-event of a task can only be scheduled, if every resource the task is assigned to is ready, i.e., if the task in question is the current task of each resource. To prevent the double execution of a task, a flag $T.scheduled$ is set, which inhibits further scheduling of the *Start*-event for the task. The event is scheduled at time $T.t_{start}$.

The output of the simulation are the timestamps of execution within the simulation and, subsequently, the delays in relation to the planned times. The event structure of the model also enables to record the timestamps each of the above events is triggered, which enables a distinction and quantification of primary

and secondary delays as well as different secondary delay reasons, e.g., resource not available, blocked infrastructure, or congestion, within the simulation.

3 DELAY DATA

The disaggregation of total delay is based on data excerpts from a system for real-time tracking of trains for railway transport companies as well as railway infrastructure managing companies. The data excerpts were provided in the railML format (Nash et al. 2004). railML is short for railway Markup Language, an open XML-based data exchange format for data interoperability of railway applications. In general, it contains information on railway infrastructure, rolling stock, and timetables – the latter being the core content and referencing infrastructure and rolling stock entities.

The data excerpt used for the presented work contains timetables, runtimes, and composition details of all trains run in Austria on a particular day. Specifically, after some data processing, each timetable entry, i.e., each OCP each train passes or stops at, holds the information given in Table 1.

Table 1: Information given in the data concerning OCPs.

OCP_id	train_number	train_part_id	operator_class	reference_nr
OCP_type	origin	arrival	departure	scheduled_arrival
scheduled_departure	arrival_delay	departure_delay	stop_delay	

OCP type is either *pass* or *stop*, depending on whether the train passes the OCP or stops at the OCP. Operator class and reference number describe the traction unit. Origin contains information on how the data was gathered. The data can come from manual input or different automated systems. Arrival delay and departure delay are computed as the difference between scheduled and actual arrival and departure, respectively. The *stop_delay* can be calculated as

$$stop\_delay = max(departure\_delay, 0) - max(arrival\_delay, 0) \tag{1}$$

It can happen that a train arrives or departs earlier than scheduled (i.e., $arrival\_delay < 0$ or $departure\_delay < 0$). For passenger transport, this is usually only a few seconds and happens rarely, e.g., when a stop on demand is not taken. For freight transport, this may be a deliberate choice of the dispatcher, to minimize disruptions on the overall traffic, and trains may be a couple of hours ahead of schedule. In both cases, it can be assumed that delays on those trains are deliberate in order to either get back on schedule or to further minimize other disruptions and therefore, this early arrival or departure is not taken into consideration when computing the *stop_delay*. The *stop_delay* describes the change of the delay of the train in a certain stop.

In order to reduce inconsistencies and errors in the data, some preprocessing is required. For example, there are trains with two consecutive entries with the same OCP id, which sometimes occur, for example, due to additional manual entries in the data. To resolve this, one entry is chosen based on the following criteria in the given order:

- All time-related entries (*arrival*, *departure*, *scheduled_arrival*, *scheduled_departure*) are equal. Hence, the entries have different OCP types. It is checked whether the times represent a stop by comparing the scheduled and actual arrival and departure times. If either the scheduled or actual departure is more than one second later than the respective arrival the times represent a stop. If not, they represent a pass. The entry with the corresponding OCP type is chosen.
- The time entries are not equal and there are no missing values. The first entry is a *stop* and the second one a *pass* such that the time points of the passing equal the arrival times of the stop. In this case, the *pass*-entry is ignored.
- If all time values are available for one entry and their origin is "manual input", this entry is chosen. If this is the case for both, the entry that appears first in the data excerpt is chosen.
- If all time values are available for one entry, this one is chosen. If this is the case for both, the first one is chosen.

- If none of the above criteria is fulfilled, the second appearing entry is chosen.

In the data, each train is split into train parts. Here, a train does not describe a physical train, but a sequence of OCPs, i.e., a list of the OCPs the physical train passes through on its way from start to end. A train part is a part of that sequence of OCPs. If, for example, a train goes from OCP A to OCP E via the OCPs B, C and D, it may be split into one train part from A to C and one from C to E. Between two train parts (at OCP C in the above example), the formation, i.e., the constellation of traction units, can change. Hence, train parts must begin with a *stop* entry if possible manipulations of the train formation are the reason for dividing a train into train parts (which is the fact for the presented simulation validation). To ensure this, if a train part starts with a *pass*, it is merged with the previous train part.

Additionally, the following strategies are implemented:

- Entries for vehicles which are not motorised are removed.
- If the arrival times of the last entry of a train part are missing, the departure times are used.
- If the departure times of the first entry of a train part are missing, the arrival times are used.
- If the scheduled times are missing, the actual times are used.
- If the actual times are missing, the scheduled times are used.
- If the first entry of a train part is not a stop, a stop is created.
- If the scheduled times of two train parts which use the same traction units are overlapping, the assigned traction unit is being considered as erroneous and, therefore, ignored.

The disaggregation of the preprocessed delay data is based on a network analysis. Therefore, the operational control points are used as nodes of the network, while the sections in between are used as edges.

4 NETWORK ANALYSIS

The disaggregation of the delay data is performed as network analysis. It is split into three parts: delays at nodes, delays on edges, and circulation-based delays.

4.1 Node Delay

This algorithm analyses delays that appear on *nodes* of OCP type *stop*. The delays are classified into primary and secondary delays. All delays that were caused by other trains blocking the required infrastructure are considered as secondary delays. All remaining delays are classified as primary.

In the available data, it cannot be known for sure whether a delay was caused by another train. Thus, the following heuristic approach is chosen: The delay of a given train A is regarded as a secondary delay if one or more other trains can be found that were on the downstream following node or edge while train A was at the considered node. The assumption is that these trains caused train A to stay longer at its current stop.

4.1.1 Algorithm

Firstly, the following parameters are defined:

- t_{node_free}: It is assumed that after a train left a node, a time t_{node_free} needs to pass before the next train can enter the node.
- t_{edge_free}: It is assumed that after a train left an edge, a time t_{edge_free} needs to pass before the next train can enter the edge.

In both cases, only one node or edge ahead is considered, as it is assumed that blocked nodes or edges further away will not influence the current train. A train is considered to be blocking the next *node*, if it

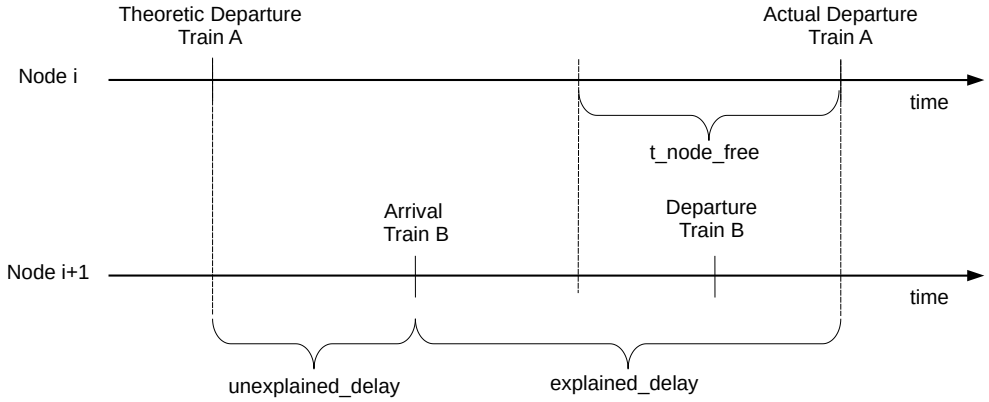


Figure 2: Depiction of the timeline, where Train A left a node delayed and Train B was found at the next node, which partially explains the delay of Train A.

arrives at that node before the actual *departure* of the considered train and leaves the same node in the time span $[departure - t_{node\_free}, departure]$. A schematic depiction of a timeline, where a train (Train A) is blocked by another train (Train B), is shown in Figure 2. This figure also contains the *theoretic_departure* of Train A. This is the time point, when Train A could have left node *i* if there were no blocked edges or nodes ahead, considering its arrival time at node *i*. It is defined as

$$theoretic\_departure := max(arrival, scheduled\_arrival) + (scheduled\_departure - scheduled\_arrival)$$

The process of checking the *edges* for possibly blocking trains is quite similar. Trains which entered the next edge before the actual *departure* of the considered train and left the edge in the time span $[departure - t_{node\_free}, departure]$ are considered to be blocking that edge.

If blocking trains can be found and it is, therefore, possible to explain the presence of a delay, it is not always the case that the whole duration of the delay can be explained. Only the part of the delay since the arrival of the blocking train can be explained, not the time before. If there are several blocking trains, it is considered that the time span from the last blocking train arriving at the blocked node or entering the edge, i.e., departing from the node at the start of the edge, can be explained. As not more than the actual delay can be explained, the *explained_delay* must be less than or equal to the *stop_delay*. The *unexplained_delay* is the remaining delay. Hence, it is defined:

$$explained\_delay := \begin{cases} min(departure - latest\_arr, stop\_delay) & \text{(node is blocked)} \\ min(departure - latest\_dep, stop\_delay) & \text{(edge is blocked)} \end{cases}$$

$$unexplained\_delay := stop\_delay - explained\_delay$$

While the delays calculated as unexplained can be classified as primary delays, the explained delays are classified as secondary delays.

4.2 Edge Delay

This algorithm handles delays that appear on *edges*, i.e., between two nodes. For these delays, the algorithm decides whether the delay can be explained or not.

As above a heuristic approach is followed: One or more other trains that were on the downstream following node or edge while the first train was on the considered edge are identified. It is assumed that these trains caused the first train to slow down (or even stop) on its current edge.

4.2.1 Data Model for Edge Delay Analysis

First, a switch from the node-centered point of view, as shown in Section 3, to an edge-centered point of view for the data is performed. In the new data, each row holds the respective information given in Table 2 for each edge.

Table 2: Information given in the data concerning edges.

start_node	end_node	arrival	departure	scheduled_arrival	scheduled_departure	edge_delay

edge_delay now can be computed as

$$edge\_delay := max(arrival\_delay, 0) - max(departure\_delay, 0) \tag{2}$$

It can also happen that $departure\_delay < 0$, i.e. when the train departs earlier than scheduled from a node, just like $arrival\_delay < 0$ is possible. In this case, an early departure should also not account to a possible edge delay. If already a *departure_delay* occurred, *edge_delay* is considered the difference between *arrival_delay* and *departure_delay* and describes, how much delay was added on this edge. Also here, only one node or edge ahead is considered.

4.2.2 Algorithm

The same parameters as above are used: t_{node_free}, t_{edge_free}. A reason for the delay, in this case, is another train which blocked the next edge or node the current train wanted to go on, while the train was on the current edge.

The downstream following *node* is blocked, if at least one train can be identified that arrived at that node before *arrival* and left it in the time span $[arrival - t_{node\_free}, arrival]$. Checking whether the *edge* following downstream is blocked is basically the same: Trains that entered the next edge before the time point where the current train left the considered edge (*arrival*) and left this edge during the time span $[arrival - t_{node\_free}, arrival]$ are considered to be blocking.

Also when it comes to considering the edges, having found one or more trains that blocked a node or an edge the considered train wanted to go on does not necessarily mean that the whole delay can be explained by these trains. Thus, the values *explained_delay* and *unexplained_delay* are defined also for edges:

$$explained\_delay := \begin{cases} min((arrival - latest\_arr), edge\_delay) & \text{(node is blocked)} \\ min((arrival - latest\_dep), edge\_delay) & \text{(edge is blocked)} \end{cases}$$

$$unexplained\_delay := edge\_delay - explained\_delay$$

4.3 Circulation Delay

This kind of delay occurs if a traction unit is delayed during the operation of a train and as a consequence the servicing of the next train by the same traction unit also becomes affected.

To analyse the switching of traction units between trains, all consecutive train parts that are operated by a distinct traction unit and the transition between said train parts are regarded. If a transition is characterized by delayed arrival of the first train part in its final destination and delayed departure of the second train part from its initial starting point, it is assumed that the delay was propagated due to the resource conflict of the designated traction unit.

If delay propagation is detected, not in all cases the whole departure delay of the second train part can be explained. Only the time span between the *scheduled_departure* of the second train part and the actual *arrival* of the first train part can be considered as *explained_delay*. If this time span is smaller than the

departure_delay of the second train part, that difference remains unexplained.

$$explained\_delay = \min(\max(arrival\_1 - scheduled\_departure\_2, 0), departure\_delay\_2) \quad (3)$$

$$unexplained\_delay = departure\_delay\_2 - explained\_delay \quad (4)$$

5 VALIDATION

For the validation, the goal is to show that the given actual data can be recreated with reasonable accuracy using the simulation. The simulation model enables different levels of complexity. Furthermore, the various levels of the network analysis allow the explanation of the different types of delays. Hence, the validation process can be split into separate steps, designing a distinct simulation experiment and using different primary delays to inject into the simulation for each step. Hence, it can be shown how the different types of delays identified by the network analysis affect the accuracy of the simulation and the necessary primary delays. The experiments build on one another and include

1. an experiment where all delays within the data are injected into the simulation as primary delays,
2. an experiment where circulation-based delays, as explained in Subsection 4.3, are deducted and the remaining primary delays are injected,
3. an experiment where additionally explained delays from blocked nodes and edges are considered. For these experiments, different values of the parameters t_{node_free} and t_{edge_free} are considered.

Ultimately, the goal is to minimize the number of primary delays that have to be injected into the model while still recreating the train runtime data, i.e., to map as many delays as possible with the simulation.

The following is applied in all validation steps:

- The agents, infrastructure, and tasks within the simulation are based on the train runtime data.
- The capacities of the sections of the tracks are computed heuristically. It is assumed that the capacities are such that no additional delays are created when operated by their planned as well as their actual schedules. The capacity of a specific section is given as the maximal number of concurrent runs on the section. Using this heuristic, it is assumed that the real capacity is underestimated.
- As a model simplification, capacities for train stations are not considered (i.e., set to infinity), because assigning a capacity to a train station is very complex and typically cannot be reduced to a single number. It would also require extensive data on the structure of all the considered train stations.
- Trains that are operated before their planned times (if actual times are before scheduled times, i.e. $departure\_delay < 0$ or $arrival\_delay < 0$), are used in the simulation using only their actual times, as the simulation can not handle negative primary delays.
- For all other trains, the scheduled times are considered as operating times, and in each task a certain primary delay (depending on the step) is added.
- The simulated times of the trains and especially the simulated times of each task of the trains, can be compared to the actual times of the train runtime data and provide a measurement for the quality of the simulation results.

5.1 Experiment without explained delays

In this experiment, the delays as calculated from the train runtime data are directly used as primary delay of the simulation. The injected primary delays for nodes and edges are bounded above and the distribution follows a function of the form

$$e^{-a \cdot x},\ a > 0.$$

The primary delays at edges have a much steeper decay, meaning smaller delays appear at edges. To prevent additional secondary circulation based delays, traction units were ignored within this experiment.

The train runtime data could be replicated without errors, i.e., all total delays in the data were also present in the simulation results.

5.2 Experiment Using Circulation-based Explained Delays

In the next step of the validation, the delays that were identified as circulation-based delays within the network analysis are deducted. The distribution of the injected primary delays at nodes and edges is highly similar to the first experiment. As the circulation does not influence delays on edges, the respective primary delays remain unchanged. This experiment was able to recreate the train runtime data without error, too.

5.3 Experiment Using All Explained Delays Without Threshold

Within this experiment, all explained delays due to blocked follow-up edges were considered on top of the circulation-based explained delays. As the network analysis only provides possible blocking trains, the explained delay is only considered if the capacity of the section was already reached by the number of identified trains. Delays explained due to blocked nodes are not deducted from the primary delays, as train stations have no capacity within the model.

The network analysis found only a small number of possible blocks in the train runtime data, when using no thresholds, i.e., other trains blocking the delayed train if it leaves the infrastructure exactly when the delayed train enters it. All of them did not reach the capacity limits of the underlying sections. Thus, effectively no additional primary delays were deducted. Therefore, the train runtime data could again be replicated exactly. It is expected that the same would have applied to a case where an actual delay would have been explained, due to the setup of the experiment.

5.4 Experiment Using All Explained Delays With Different Thresholds

In the next step of the validation, the time of a train blocking a section is varied by raising the corresponding parameters t_{node_free} and t_{edge_free} iteratively from 5s to 35s in steps of 5 seconds. In this step, a train is considered to be blocking the delayed train if it leaves the infrastructure within a certain time before the delayed train enters it. As one would expect, this raised the number of explained delays as well as the number of deductions of primary delays for the simulation. The distributions of the injected primary delays are highly similar to the previous experiments for all thresholds. Table 3 shows an overview of the results. As operational and delay data are very sensible for ÖBB, the authors decided against presenting absolute numbers, and relative values are shown instead.

For the explained delays – as identified in the network analysis – the experiment using a threshold of 0s, as presented in Section 5.3, is used as the baseline scenario and Table 3 shows the relative increase for the different thresholds. The column *expl. delays (number)* depicts the increase of the total number of edges, where a part of the delay was classified as explained. The column *expl. delays (duration)*, similarly, presents the increase of the sum of durations that are identified as explained. Additionally, these numbers are compared to the amount of explained delays that were actually used to deduct primary delays in the simulation. Again, explained delays were only considered if the capacity limit of the underlying section was reached.

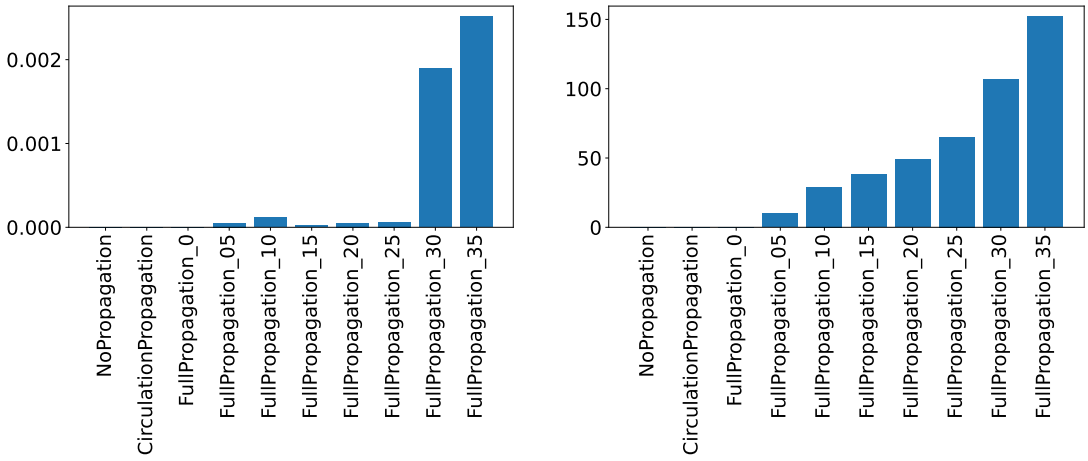
As the threshold is not included in the simulation model, a deviation of the simulation results compared to the train runtime data has to be expected, and can also be seen in Table 3. The absolute count of errors is displayed, as well as the weighted average percentage error (WAPE) for the total sum of delays. The WAPE is defined as

$$\text{WAPE} = \frac{\sum_{i=1}^{n} |A_i - F_i|}{\sum_{i=1}^{n} |A_i|}$$

where here, the A_i are the delays from the train runtime data and the F_i are the simulated delays. Furthermore, Figures 3a and 3b show the WAPE and the number of errors for all experiments and thresholds.

Table 3: Overview of results from network analysis as used within the simulation experiments.

t_{node_free} t_{edge_free}	expl. delays (number)	expl. delays (duration)	used delays (number)	used delays (duration)	errors (number)	WAPE ($\cdot 10^{-6}$)
0s	1.0	1.0	0.00%	0.00%	0	0.0
5s	2.9	3.1	10.53%	11.50%	10	51.18
10s	5.6	4.0	13.70%	13.97%	29	117.23
15s	9.1	6.9	15.25%	13.93%	38	22.74
20s	12.5	10.0	14.72%	14.49%	49	51.11
25s	17.8	14.2	15.09%	15.21%	65	63.93
30s	23.0	17.2	15.05%	18.43%	107	1 894.22
35s	29.1	21.6	16.93%	21.11%	152	2 513.37



(a) (b)

Figure 3: (a) Weighted average percentage error (in seconds) of total delays for all experiments. (b) Absolute number of occurred errors for all experiments.

6 DISCUSSION

The validation process shows that, with careful parametrization, the simulation model can indeed reproduce the train runtime data. However, in order to recreate a more realistic distribution of primary delays, certain errors within the simulation have to be accepted. As Figure 3a and Table 3 show, the error increases with the value of the parameters t_{node_free} and t_{edge_free}, showing a disproportionate jump between 25s and 30s. An explanation could be that if $t_{node\_free} = t_{edge\_free} \geq 30s$, the threshold is too high, meaning that the difference between the time span for track clearance in simulation and real-world gets too large.

There is certainly potential in the simulation model to further improve the quality of the simulation results with respect to the given data by considering the threshold from the network analysis in the simulation. If implemented correctly, it is to be expected that this leads to fewer and smaller errors within the simulation.

The heuristic approach comes with some drawbacks, e.g., single tracks cannot be considered (also due to unavailability in the data), but rather aggregated nodes and edges. Also, capacities at train stations are currently not considered. To get an idea of how many additional primary delays could be deducted if capacities of train stations would be considered within the simulation, Table 4 shows the explained delays that were identified because of nodes in relation to the explained delays due to blocked edges for each value of t_{node_free} and t_{edge_free}.

Assuming that an equal percentage of explained delays could be deducted due to reached capacities at train stations, as is currently deducted due to reached capacities at sections, it can be concluded that a deduction of 1 to 2 times the current amount of deducted seconds depending on the used threshold could be performed if capacities for train stations were used.

Table 4: Network analysis results for explained delays because of nodes relative to corresponding results for explained delays because of edges.

t_{node_free} t_{edge_free}	expl. delays (number)	expl. delays (duration)
0s	1.5	0.5
5s	3.7	1.0
10s	3.6	1.7
15s	3.3	1.4
20s	3.2	1.4
25s	2.7	1.2
30s	2.4	1.2
35s	2.2	1.1

Besides being a crucial input for the simulation, the disaggregation approach into explainable and unexplainable delays can be used for a detailed performance analysis as well. It has to be stated though that a different parametrization and slight adjustments of the algorithm shall be considered.

ACKNOWLEDGMENTS

Significant parts of the presented results were achieved within the project VIPES (FFG project number 893963). FFG is the central national funding organization and strengthens Austria's innovative power. VIPES is funded by the Federal Ministry for Climate Protection, Environment, Energy, Mobility, Innovation and Technology (BMK) as part of the call for proposals Mobilität der Zukunft. www.ffg.at. Data and data expertise were generously provided by the Austrian Federal Railways (German: Österreichische Bundesbahnen, ÖBB).

REFERENCES

Brunmeir, D., M. Bicher, N. Popper, M. Rößler, C. Urach, C. Rippinger, et al. 2023. "Four Years of Not-Using a Simulator: The Agent-Based Template". In *2023 Winter Simulation Conference (WSC)*, 255–266. IEEE https://doi.org/10.1109/WSC60868.2023.10408482.

Manitz, J., J. Harbering, M. Schmidt, T. Kneib, and A. Schöbel. 2017. "Source Estimation for Propagation Processes on Complex Networks With an Application to Delays in Public Transportation Systems". *Journal of the Royal Statistical Society Series C: Applied Statistics* 66(3):521–536.

Nash, A., D. Huerlimann, J. Schuette, and V. P. Krauss. 2004. "Railml – A Standard Data Interface for Railroad Applications". *WIT Transactions on the Built Environment* 74:233–240.

Ochiai, Y., Y. Shibata, and N. Tomii. 2019. "An Algorithm to Identify Delay Propagation Routes Based on Visualization of Association Rules". In *Proceedings of the 2nd International Railway Symposium Aachen*, 245.

Palmqvist, C.-W., I. Johansson, and H. Sipilä. 2023. "A Method to Separate Primary and Secondary Train Delays in Past and Future Timetables Using Macroscopic Simulation". *Transportation Research Interdisciplinary Perspectives* 17:100747.

Rehman, M. and S. A. Pedersen. 2012. "Validation of Simulation Models". *Journal of Experimental & Theoretical Artificial Intelligence* 24(3):351–363.

Rößler, D., J. Reisch, F. Hauck, and N. Kliewer. 2021. "Discerning Primary and Secondary Delays in Railway Networks Using Explainable AI". *Transportation Research Procedia* 52:171–178.

Rößler, M., M. Wastian, M. Landsiedl, and N. Popper. 2018. "An Agent-based Model for Robustness Testing of Freight Train Schedules". In *Proceedings of the MAS: The 17th International Conference on Modelling & Applied Simulation*, 101–105. Budapest, Hungary.

Rößler, M., M. Wastian, A. Jellen, S. Frisch, D. Weinberger, P. Hungerländer, et al. 2020. "Simulation and Optimization of Traction Unit Circulations". In *2020 Winter Simulation Conference (WSC)*, 90–101. IEEE https://doi.org/10.1109/WSC48552.2020.9383926.

Wen, C., Z. Li, P. Huang, J. Lessan, L. Fu, and C. Jiang. 2020. "Cause-specific Investigation of Primary Delays of Wuhan–Guangzhou HSR". *Transportation Letters* 12(7):451–464.

Wen, C., Z. Li, J. Lessan, L. Fu, P. Huang, and C. Jiang. 2017. "Statistical Investigation on Train Primary Delay Based on Real Records: Evidence From Wuhan–Guangzhou HSR". *International Journal of Rail Transportation* 5:1–20.

Yamamura, A., M. Koresawa, S. Adachi, and N. Tomii. 2013. "Identification of Causes of Delays in Urban Railways". *Computers in Railways* 13:403–414.

AUTHOR BIOGRAPHIES

NADINE SCHWAB is Ph.D. student at TU Wien and senior data scientist at dwh GmbH. Her scientific focus lies on machine learning, explainability and visualization. Her email address is nadine.schwab@dwh.at.

MATTHIAS RÖSSLER is researcher at dwh GmbH. He focuses on M&S in health care, logistics, and energy simulations using ABMs, ODE- and DAE-based models. He is working on his Ph.D. thesis at TU Wien. His email address is matthias.roessler@dwh.at.

HANNAH KASTINGER is a researcher at dwh GmbH. She focuses on data analysis and modeling, especially in the railway sector. Her email address is hannah.kastinger@dwh.at.

GÜNTER SCHNECKENREITHER is a PostDoc researcher at TU Wien and senior scientist at dwh GmbH. His expertise and field of interest lies in mathematical modeling and stochastic simulation. His email address is guenter.schneckenreither@tuwien.ac.at.

MATTHIAS WASTIAN is senior data scientist at dwh GmbH. His focus lies on data analysis, machine learning, as well as modeling and simulation in the fields of logistics, health care, and energy simulation. His email address is matthias.wastian@dwh.at.

NIKI POPPER is research associate at TU Wien and CSO of dwh GmbH. His research focus lies on comparison of different modeling techniques. His email address is nikolas.popper@tuwien.ac.at.